Organization (S): EDF/IMA/MMN

Data-processing handbook of Description
D0.03 booklet: Structure
D0.03.01 document

General architecture of Code_Aster

Summary:

One gives in this document an outline of the three ideas which structure in an important way the software Aster:

· The supervisor of execution,
· The manager of memory JEVEUX,
· concept of calculation elementary.

Contents

Data-processing handbook of Description
D0.03 booklet: Structure
HI-75/95/007 Ind A

---

1 Introduction

Code_Aster
is made up various "modules" which one can classify in:
· program main FORTRAN 77,
· routines FORTRAN 77 (subroutine or function),
· functions C,
· routines CAL (CRAY assembling language),

· catalogues.
The whole of the texts of these modules constitutes the source of Aster (approximately 400.000 lines).
catalogues are textual files which **parameterize** certain programs: mainly the analyzer
elementary orders and calculations (within the meaning of the finite element method).
The functions C carry out certain impossible tasks "system" in FORTRAN 77: allowance
dynamics, measurement of time,…
The routines CAL were written to optimize performances (CPU) of the method of resolution
linear systems by Combined Gradient.
If the few functions C and the routines CAL are forgotten, we see that the Aster program is
constituted of:
· a few hundreds of catalogues,
· a few thousands of routines FORTRAN 77.
The goal of this document is to help "to find itself" in this great number of modules: only for
FORTRAN, we calculated that the tree of complete call of the Code_Aster program was written
on more than 6.108 lines, which excludes to give it in appendix!
How, under these conditions, to identify the sources relating to a given functionality?
Where to insert a new functionality?
A form of answer to these 2 questions is in the general architecture of the code which was selected with
beginning of Project (07/89) and which was not called into question since.
This architecture can be summarized about in **three** ideas:
1) Code_Aster can be seen like a whole of independent orders that the user
connect with its liking,
2) these orders exchange named objects ("concepts") which are them same
composed of objects JEVEUX,
3) Code_Aster being a code of finite elements, concept of calculation "elementary" (i.e made by one
finite element) is strictly codified because it constitutes to some extent the "core" of
numerical method.

Note:

The first 2 ideas are **very general** and could be used as architecture with the many ones
software out of the field of the finite elements.

They are these three ideas that we will develop in the following paragraphs:
1) is identified about so that one calls the **Supervisor** of execution [§4],
2) is possible thanks to the manager of memory **JEVEUX** and the **Structuring** of the **Data**
[§5],
3) corresponds to the parameterization of elementary calculations and the existence of the routine
CALCULATION [§6].

Note:

· idea 2 structure not the code strictly speaking, but it makes it possible to carry out 1,
· if these three ideas strongly structure the code, they also form the "yoke" of
programming: one cannot withdraw oneself from it. The remainder of the programming is with little
close free,
· the implementation of idea 3, in addition to the fact that it structure a large volume of code
(19000 lines of catalogues and 70000 lines of FORTRAN), a certain number solidifies of
essential notions of the program:

- node, mesh, grid,
- size, option, finite element,
- fields, assembled matrices.

---

2 General architecture of routines FORTRAN

Schematically, the organization of routines FORTRAN is as follows:
program ppal
Supervisor
order 1
CALCULATION
Aster
order 2
JEVEUX
order 3
...
order N
CALCULATION
elementary calculation 1
elementary calculation 2
elementary calculation 3
...
elementary calculation p

Note:
To the 01/10/94:
a number of orders: N = 128
a number of elementary calculations: p = 3043

The Supervisor (as the routine CALCULATION) structure the code because they **affirm an independence**
between the routines which they call:
routine
0P0001
--> order 1
routine
0P0002
--> order 2
...
...
routine
TE0001
--> elementary calculation 1
routine
TE0002
--> elementary calculation 2
...
...

· the bond between an order user and number I of the OP000i routine which corresponds to him is given in the catalogue associated with this order [§ 3.1],

Data-processing handbook of Description

· the bond between an elementary calculation (for example: the calculation of geometrical rigidity for one element of hull of the type DKT) and the routine TE00031 which corresponds to him is given in catalogue associated with this finite element [§ 3.2].

Independence between the OP000i routines is very interesting. It wants to say that to include/understand the programming of order one does not need to include/understand the others; the only bonds enters the orders are the structures of Data which they exchange [§ 5.2]. Those are described in [D4].

The independence of the TE000i routines is more natural (one will see however that the same routine TE000i can be associated several close elementary calculations).

Of course, the preceding diagram does not want to say only all source FORTRAN corresponding to order I is in the OP000i routine: the programmer of an order (like that of one elementary calculation) can structure its order as it hears it: it can "cut out it" in several routines.

Schematically, one can write:

OP000i
-->
CALCULATION, JEVEUX or any other utility which can be used for several different orders.
-->
routines specific to the order 0P000i (cutting functional of OP000i)

The orsque one one seeks the source code associated with a given functionality, one must thus be posed them following questions:

· is it about a functionality specific to an order?

1) not: to see the utilities common to several orders [D7.01],

· is it about a functionality specific to an elementary calculation?

2) not: to see the utilities common to several elementary calculations [D7.02].

3
Structure of the catalogues

We distinguish two kinds of catalogues:

· catalogues of orders which parameterize the supervisor,
· catalogues of elements which parameterize:

-
routine
CALCULATION,
-
orders LIRE_MAILLAGE and AFFE_MODELE.

3.1
Catalogues of orders

Architecture is simple: to each ordering of name, commande_i corresponds a catalogue of
even name. These catalogues all are independent from/to each other.

catalogue
commande_1
catalogue
commande_2
...
catalogue
commande_n

The contents of the catalogue of an order are described in [D5.01.01].

3.2
Catalogues of finite elements

Architecture is still rather simple there. The description of the contents of these catalogues is made in

[D3.02.01].
“cata1 --> cata2” wants to say: the catalogue cata1 is pressed on the catalogue cata2. In other words, it uses
entities described in the catalogue cata2.
Phenomenon/Modeling
type_élément 1
option 1
type_élément 2
option 2
...
...
type_élément Q
option R
types of mesh
sizes
Note:
To the 01/10/94:
numbers type_element: Q = 233
a number of options: R = 159

---

4 The Supervisor of execution

4.1 General

What one calls “Supervisor” is the whole of the routines FORTRAN which belong to
library SUPERVIS.
One can logically cut it into two:
SUP1:

what is used to connect the various orders; i.e. all it
who is (in the tree of call) between the main program and them
OP000i routines (including the main program). One includes in
supervisor contents of 3 particular orders: BEGINNING,
CONTINUATION and END,
SUP2:
what allows the communication of information with OP000i:
routines GETXXX [D6.03.01].

4.2

Operation general of the Supervisor

1) opening of the 3 data bases JEVEUX [D6.02.01] (makes 3 files of direct access of them).
- bases
"LOCAL" it is a base of work reserved to the Supervisor, this base is not
safeguarded at the end of the execution
- bases
"VOLATILE" it is the base reserved for the objects of work (except Superviseur), this
base is not safeguarded at the end of the execution,
- bases
"TOTAL" it is the base of the user. It will contain at the end of the execution them
concepts corresponding to the orders carried out

2) reading of the catalogues
-
catalogues of orders
- catalogues
elements

3) reading of the command file of the user
-
reading in free format; elimination of the comments,
-
syntactic checks (using the catalogues of orders),
-
orthography of the key words,
-
types of the arguments,
-
exclusion of the key words,…
-
assignment of the default values,
-
creation of the concepts corresponding to values (DEFI_VALEUR [U4.21.10]) and to
interpreted functions (FORMULA [U4.21.11]),
-
evaluation of the numerical expressions (key word EVAL [U4.21.11 §4.1]),

-
information storage of the command file in objects JEVEUX (bases
local).
4) request for execution of the orders of the user [§ 4.3],
5) impression of the execution time of each order,
6) validation (progressively) of the concepts created by the orders: this allows
“to take again” a calculation which badly finished,
7) closing of the data bases at the end of the execution,
8) program stop.

4.3 Ask execution of the orders

The Supervisor “buckles” twice on the orders read in the command file of
the user:

1st master key:
phase of additional checks: one checks the data of the user
(what could not be checked by the supervisor),

2nd master key:
production run: truly the order is carried out.

If the user asked:
BEGINNING (PAR_LOT: “YES”,…)
(it is the default value)
the Supervisor carries out the 1st master key on all the orders before starting 2nd
pass. This makes it possible to check all the command file before starting
execution.

If not: BEGINNING (PAR_LOT: “NOT”,…)
The Supervisor carries out the 2 master keys one after the other for each order.

Note:
The Supervisor connects the orders the ones after the others. “Sentences of language " (orders) are followed without instructions of control: IF THEN ELSE, loops C,…

4.4 Orders
“Supervisory”

Note:
This paragraph can be jumped in first reading.
The preceding paragraph [§4.3] related to the execution of the “ordinary” orders.
The ordinary orders are those whose number lies between 1 and 9998.
The orders which are not ordinary are:
· the orders BEGINNING and CONTINUATION which do not have an external catalogue,
· the order END associated with the number 9999 which is charged (inter alia things) with to discharge the memory and to close the data bases,
· orders known as “supervisor”.
The Superviseur orders have a catalogue (like the ordinary orders), but their number is a negative number (key word NUMERO_SUPERVIS__ instead of NUMERO__). Routines FORTRAN associated name OPS00i.
There are today (01/10/94) 7 Superviseur orders.
The difference in behavior between an order Supervisor and an ordinary order is that the supervisor carries out a preliminary master key on the Superviseur orders. The idea being that after this preliminary master key, all occurs as if the command file only contained ordinary orders. This preliminary master key can be regarded as preprocessing of command file. The “echo” of the command file (which one finds in the file MESSAGE) represent the state of the command file after this preliminary phase.
The 7 current Superviseur orders break up into two: those which are destroyed at the end preliminary master key: INCLUDE, PROC, RETURN and MACRO_MATR_ASSE and those which are not

---

destroyed: DEFI_VALEUR, FORMULA and TO DESTROY. For these 3 last, one will thus pass three times
in the associated OPS00i routine: pass preliminary, passes from additional checks and
pass from execution.

Principal interest of the orders Supervisor (in addition to to have allowed the "include", the use of the functions
interpreted and of the named constants) is to allow the development of "macro" orders
; MACRO_MATR_ASSE is an example. At the time of the preliminary master key, the order
MACRO_MATR_ASSE dynamically generates the text of several ordinary orders then it
is destroyed. The development of such macro-orders is documented in [D5.01.02].

That is to say the command file:

C_O1
C_S1
C_O2
C_S2
C_O3

where:

C_Oi
are ordinary orders

C_Si
are Superviseur orders

C_S1
is a Superviseur order of the macro type orders which generates them
ordinary orders C_O4 and C_O5.
C_S1 is destroyed at the end of the preliminary master key

C_S2
is a Superviseur order which is not destroyed

· Séquence of the master keys if BEGINNING (PAR_LOT: "yes")

-
place preliminary (for the orders supervisor only): (pp)
-
C_S1
pp
-
C_S2
pp
-
pass from additional checks: (statement)
-
C_O1

statement
-
C_O4
statement
-
C_O5
statement
-
C_O2
statement
-
C_S2
statement
-
C_O3
statement
Data-processing handbook of Description
D0.03 booklet: Structure
HI-75/95/007 Ind A

-
pass from execution: (EP)
-
C_O1
EP
-
C_O4
EP
-
C_O5
EP
-
C_O2
EP
-
C_S2
EP
-
C_O3
EP
· Séquence of the master keys if BEGINNING (PAR_LOT: “NOT”)
-
C_O1
statement
-
C_O1
EP
-
C_S1
pp

-
C_O4
statement
-
C_O4
EP
-
C_O5
statement
-
C_O5
EP
-
C_O2
statement
-
C_O2
EP
-
C_S2
pp
-
C_S2
statement
-
C_S2
EP
-
C_O3
statement
-
C_O3
EP

5 JEVEUX and Structuring of the Data

We in this paragraph will try to release the principal functionalities of the manager of memory JEVEUX and of the use that one makes some in Aster.

5.1 Administrative JEVEUX of objects

JEVEUX is the whole of routines FORTRAN described in [D6.02.01].
These routines allow:
· to create objects,
· to save them (writing on disc),

· to destroy them,
· to release them (main memory),
· to point out them (in main memory),
· to copy them, print them,…

5.1.1 What an object JEVEUX?

· A whole of **homogeneous** information (entireties, realities, complexes,…),
· each object is named (24 characters),
· each object with accessible attributes in reading (and sometimes in writing):
-
length (for 1 vector),
-
type of the values: entirety, reality,…
- ...
· each object has a "image disc virtually",
· there exist **simple objects** (roughly speaking, they are vectors),
· there exist **collections** of objects,
-
the objects of a collection are all of the same type (but it can have lengths different),
-
the access of an object of collection is done by the name of the collection plus something which identify the object:
-
a number (numbered collection),
-
or a name (named collection).

5.1.2 Allowance
dynamics

One can create, at any moment to release (and destroy) an object JEVEUX. That makes it possible to manage
dynamically memory.
Of course, this possibility is used to allocate working areas. It is the only **mechanism of dynamic allocation authorized** in Aster because it manages the **whole** of the place memory available:
(one understands by memory available the place available in the “Area” requested from the execution less the volume of the achievable code minus the zones managed by system UNICOS).

5.1.3 Memory
virtual

When an object A is released, JEVEUX regards it as “déchargeable” (on disc). If news
requests are made on other objects and that the place in main memory has suddenly missed, object A will be written on disc and its place will be recovered.
JEVEUX thus makes it possible to reach (at different moments) more memory than does not contain any really the “Area” of main memory allocated with the execution.
It thus acts like a system of “virtual memory”.

5.1.4 Writing and reading on disc

· When one saves explicitly an object (routine JESAUV), this one is written on disc,
· when the execution of Aster finishes, one automatically saves all the objects of the base
total which was not it yet,
· when an object in main memory is pointed out (routine JEVEUO), this one is read on
disc if it were discharged and recopied in main memory.

JEVEUX thus makes it possible to be freed from all the binary readings and writings on disc.

5.2

Structuring of the Data
The orders of Aster exchange objects named by the user (8 characters) that one
call concepts.
Example:
steel = DEFI_MATERIAU (ELAS: (E: 300.000 NAKED: 0.3));
chmat = AFFE_MATERIAU (MATER: steel…);
The concept "steel" created by order DEFI_MATERIAU is an argument of entry of
order AFFE_MATERIAU.
A concept is in fact a named Structure of Data (SD in language programmer).
A structure of data is anything else only one **whole** of objects JEVEUX.
One can then "handle" in FORTRAN of the structures of data complex: the passage of
SD in argument is done by its name (character string).
This largely improves the definition of the interfaces of the routines: instead of transmitting
multitudes of vectors in arguments, one transmits some structures of data.
The regrouping of a whole of objects JEVEUX in a structure of data is done by
simple known conventions of names of the whole of the programmers.
A Structure of Data **is typified**. When one carries out (for example) the order:
= LIRE_MAILLAGE netted ();
this one must create a SD of the type grid and of name "netted". At the end of the execution of
order, it must exist on the "TOTAL" basis certain numbers of objects JEVEUX of which
the unit forms the SD netted:
".DIME NETTED"
".CONNEX NETTED"
".NOMNOE NETTED"
".NOMMAI NETTED"
...
The first 8 characters of the objects are those coming from the user. The other characters (which
the objects from/to each other are used to distinguish) are fixed by the programmers. The description of
contained objects .DIME, .CONNEX,… form what one calls the description of the SD of the type
grid (cf [D4]).
Important remark:
Only information necessary to the good unfolding of an order is:
· values that the user provided behind the key words of the order:
entireties, realities,…
· SD (already created by preceding orders) given in argument.
There is no information under-terraine (COMMONS, files,…) between the orders. It is the respect
of this rule which ensures the real independence of the orders between them.
Data-processing handbook of Description
D0.03 booklet: Structure
HI-75/95/007 Ind A

Code_Aster ®
Version

The only exceptions to this rule are:

· the SD catalogues [D4.01.01] which is accessible everywhere (but it is never modified),
· certain writings (or readings) in files. In this case, the name of the order is
always of the form: IMPR_XXX or (LIRE_XXX).

The "format" of the file can then be seen like the description of a SD, for example:

- grid
Aster (LIRE_MAILLAGE),
- function
Aster (LIRE_FONCTION),
-
results to visualize by I-DEAS (IMPR_RESU (FORMAT: IDEAS…)).

6 Calculations
elementary

We saw with [§2] that various elementary calculations were numerous in Aster.
This significant number of elementary types of calculations results:

· of the great number of finite elements in the codes computer of structures:
-
isoparametric 2D in Thermics, Mechanics and Accoustics,
-
isoparametric 3D in Thermics, Mechanics and Accoustics,
-
elements of structures: beams, hulls,…
- elements
incompressible,
-
elements of fluid interaction/structure,
- ...
· and of the great number of options of possible calculation:
-
mechanical or thermal rigidity,
- mass,
damping,

-
geometrical rigidity or centrifuges,
-
constraints, deformations, flow,
-
surface, voluminal or linéïques forces,
-
change of gravity, thermal dilation,…

In Aster today (01/10/94), one a:

· 233 types of finite elements (approximately 19000 lines of catalogues),
· 159 options of elementary calculation,

what entraine more than 3200 theoretically possible elementary calculations (undoubtedly much more). Only 3043 of these elementary calculations are actually programmed (approximately 70000 lines of FORTRAN).

These 3043 elementary calculations are made in 310 TE000i routines; indeed, several calculations elementary can be implemented in only one TE000i. For example, it is rather easy of to parameterize the programming of all the isoparametric elements 2D.

The large volume of the code concerned with elementary calculations justifies an effort of parameterization of these calculations. The objectives of this parameterization are:

· to simplify to the maximum the programming of TE000i: data of an elementary calculation "arrive" in the routine in the form wanted by the programmer (and described in catalogue element [D3.02.01 §7]),
· to have a single routine (CALCULATION) managing all elementary calculations: constraints, rigidity, thermal "mass",…. What avoids multiplying the "loops" on the elements, them controls and error messages: the "function" CALCULATION accounts for 3500 nevertheless lines…
· to impose types of Structures of Data commun runs on all the results of calculations elementary: the CHAM_ELEM (fields by elements) and the RESU_ELEM (matrices and elementary vectors).

The assembly of the matrices and the elementary vectors can then be made in two
routines (ASMATR and ASSVEC).
The mechanisms of this parameterization are explained in [D3.02.01].
The documents [D5.04.01] and [D5.04.02] describe the manner of introducing new calculations
elementary.

Intentionally white left page.

Organization (S): EDF/MTI/MMN

Data-processing handbook of Description
D1.01 booklet: Environment of development and tools
D1.01.02 document

Environment of development on platform
autonomous: run_aster

Summary:

This document describes the environment of development on autonomous platform (SOLARIS, HPUX, IRIX,
linux or Windows NT4). It comprises in particular a note of use of the procedure run_aster and description of the files of associated parameters.

Data-processing Descriptive handbook
D1.01 booklet: Environment of development and Tools
HI-75/01/005/A

Code_Aster ®
Version
6.0

Titrate:
Environment of development on autonomous platform: run_aster Date:
05/02/02

1
General and pre presentation required

1.1 Objectives

The constitution of a local version and the overload of Code_Aster, in addition to the traditional procedures of compilation of the source code (C and FORTRAN) and of edition of the bonds, requires some operations particular like the constitution of the catalogues compiled associated the process control language and the description of the finite elements. Procedures of validation by the execution of a series of tests of references constitute a point of passage essential to the validation of a development, with less to ensure itself of nonthe regression.
The concern of ensuring the portability on various platforms and various operating systems has conduit to write a whole of procedures presenting a minimum of adherence system. The choice directed itself towards the language of script tcl, which ensures us a broad diffusion and makes it possible to touch with the time the world Unix and Windows. The call of procedures not being really convivial, it seemed useful to coat their calls, without to develop a complex interface. We thus have chosen a simple means to call in a generic way various operations of construction and of use of an achievable room by using a single ordering and a file of parameter to words keys easily éditable.
This document describes the various accessible operations, the syntax of the file of associated parameter, like all the preliminary actions of organization of the sources and bookshops.

1.2 Pre necessary

The target machine must make it possible to reach directly or by assembly of the type NFS the sources, with bookshops and with the files of tests, it is not envisaged a distant access of type rsh, CCP under Unix. These various files must be deposited under a fixed tree structure organized (under Unix) of following way:

sources FORTRAN
bibfor/algeline

bibfor/algorith

bibfor/assembled

bibfor/calculel

bibfor/cataelem

bibfor/elements

bibfor/jeveux

bibfor/modelled

bibfor/postrele

bibfor/prepost

bibfor/soustruct

bibfor/stbtrias

bibfor/supervis

bibfor/utilifor

bibfor/utilitai

fermetur
sources C
bibc/algeline

bibc/include

bibc/supervis

bibc/utilitai
sources CATALOGUES
catapy/order

catalo/compelem

catalo/options

catalo/typelem
python sources
bibpyt/Accas

bibpyt/Cata

bibpyt/Lecture_Cata_Ele

bibpyt/Supervisory
sources of the tests
tests
Data-processing Descriptive handbook
D1.01 booklet: Environment of development and Tools
HI-75/01/005/A

The preliminary installation of software TCL/TK is necessary, all scripts of construction resting on this language. In addition it is essential to have proceeded to the installation of version 2.1 of python. The machine must have a compiler FORTRAN (FORTRAN 77 for Code_Aster) and of a compiler C.

The linkage editing requires bookshops MED version 2.0 and HDF version 5.0 which will have to be built au préalable using the procedure of provided installation.

bookshop MED
libmed.a
bookshops HDF
libhdf5.a
bookshop python

libpython2.1.a
TclTk bookshops
libtcl8.3.a libtk8.3.a libX11.a
bookshop BLAS *
/usr/lib64/libcomplib.sgimath.so
detection of the signals *
lfpe

**** only under SGI IRIX64***

1.3
Description of the result of the procedure of installation

The procedure of installation in addition to the source files described above, deposits scripts of construction and some repertories under the repertory indicated at the time of the installation. The file of
configuration and script is adapted according to the provided information (compilers used, bookshops,…).

scripts of launching
tcl/run_aster.tcl
tcl/aster.tcl
tcl/run
tcl/run_aster
tcl/conf/init_conf
tcl/make_aster
tcl/make_exec
tcl/make_cata
tcl/make_etude
tcl/make_lib
tcl/make_test
repertories of development
dvp/
(used by scripts)
dvp1/
dvp2/

The procedure run_aster consists of three command files tcl (aster.tcl,
run_aster.tcl and run) and of one (or several) file (S) of parameters (for example make_exec, make_cata, make_etude…).
aster.tcl contains the various procedures tcl associated with each action.
run_aster.tcl contains the procedure of analysis of the file of parameter and a generic procedure launching the calls to the various actions.
run makes it possible to coat the call with the preceding procedures directly since Shell script (Unix or

Windows) without passing by a Tcl window (of wish type).
Script run_aster makes it possible to position the environment python.
In addition, it is essential to provide a file of configuration describing the whole of adherences related to the object computer (platforms SOLARIS, HP, IRIX, linux, or NT). It file (tcl/conf/init_conf) makes it possible to indicate the various tools (compiler, data link, bookshops, etc), their access path and all options specific to the platform.

The execution of make_aster makes it possible to build a complete version of the code in nodebug mode or
debug and led to the creation of the following tree structure:

---

modules nodebug objects
obj/
obj_f/
modules debug objects
dbg/
dbg_f/
bookshops nodebug
lib_obj/lib_aster.lib
lib_obj/ferm.lib
lib_obj/asterm.obj
bookshops debug
lib_dbg/lib_aster.lib
lib_dbg/ferm.lib
lib_dbg/asterm.obj
achievable nodebug
asteru.exe

achievable debug
asterd.exe
catalogues compiled
elements
order
cata_ele.pickled

2 Use of the procedure

2.1 General

The script of launching is coated in an achievable file (run_aster under Unix, run_aster.bat under Windows) calling Shell script tcl directly. This script must be launched on object computer, possibly after connection in rlogin (under Unix).
% run_aster will mon_fichier_para

The file of parameters will mon_fichier_para is analyzed overall:

- ***all that follows the character % is ignored,***
- ***the key words all are in small letters, their order is indifferent, but it is the last occurrence in the file which is taken into account,***
- ***the key words are located on the left character: (the white are ignored around),***
- ***it has two types of key words:***
  - those being used to activate a task: make_aster, make_exec, make_cata, make_lib, make_test, make_etude,… they carry out the various tasks in one fixed order (key word action),
  - those being used to inform the files and the various parameters (key word parameter),
- ***any line not containing a key word is ignored,***
- ***all the file names must be indicated in absolute.***

The repertory or provided file names, are it always in absolute (for example /home/utilisa/mon_fichier_de_donnees.dat).

2.2 Description of the file of parameters

*The symbol: separate the key word from the associated value. The key word action are all of the form make_action or maj_action, they yes takes only the value or not. Key words parameters depend on the key word **action**, they are described for each order in the paragraph [§2.4].*
*Syntax is common to the whole of the key words **action**, except for the key word make_etude.*
In this case, to inform the various result or data files associated with a study, the key word is consisted of the name of the logical unit followed of at least one of the letters D or R indicating that the file is in data ou/et in result.
Example:
fort.20 DR.: /home/user/toto.mail

The key word rep_tcl is obligatory in any file of configuration, it makes it possible to charge them various procedures appearing in the file aster.tcl.
Data-processing Descriptive handbook
D1.01 booklet: Environment of development and Tools
HI-75/01/005/A

2.3
Description of the file of configuration

This file makes it possible to define the various compilers, their localization, the parameters used, them external bookshops and the bookshops system. It is a file with key words which follows the syntax of the files
parameters (one uses the same analyzer). There is no checking during the analysis of this file
of configuration, no key word is obligatory, the default value is put at ""(chain reduced to
a white).

Key word
Associated parameter

f77
file name
Compiler FORTRAN 77
DC
file name
Compiler C
lib
file name
Order filing of the bookshops
link
file name
Data link
python
File name
Achievable python (interpreter of orders python)
rep_incl
name of repertory

Contains the includes C; the file names must
to have the extension .h
opcc
character string
Described the options of compilation C
opcc_dbg
character string
Described the options of compilation C in debug mode
opf77
character string
Described the options of compilation FORTRAN
opf77_dbg character string
Described the options of compilation FORTRAN in mode
debug
path_hdf
name of repertory
Contains bookshops HDF
path_med
name of repertory
Contains bookshops MED
op_link
character string
Described the options of linkage editing
lib_sys
character string
List bookshops systems
lib_hdf
character string
List bookshops HDF
lib_med
character string
List bookshops MED

The file of configuration associated with the platform with installation (init_conf) belonged to supply and is adapted at the time of the installation.

2.4
Description of the orders available

One describes the whole of the possible actions below, while specifying what is awaited behind each key word. The key words must imperatively appear in the file of parameters, they can if required to be assigned to the value " "(chain reduced to a white character).

In the tables below, one boldfaces the parameters which are likely to be

modified with each execution of script. In theory, the other parameters (type of platform, localization of the sources/catalogues/bookshops Aster of reference, of scripts tcl, the files of configuration) are unchanged from one execution to another and can thus be indicated once for all.
Data-processing Descriptive handbook
D1.01 booklet: Environment of development and Tools
HI-75/01/005/A

Code_Aster ®
Version
6.0

Titrate:
Environment of development on autonomous platform: run_aster Date:
05/02/02
Author (S):
Key ***J.P. LEFEBVRE***
:
D1.01.02-B Page
: 6/12

2.4.1 Initial construction of a version of Code_Aster: make_aster

This procedure makes it possible to build the bookshops, achievable modules and the catalogues compiled reference. It is recommended to build the optimized version and the instrumented version for the déboggage.

Key word
Associated parameter

make_aster yes

rep_tcl
name of repertory
Repertory containing the provided files tcl
fic_conf
file name
File of configuration specific to the platform
used
platform
IRIX, HPUX, SOLARIS, Plate-forme targets (used for compilation
P_LINUX or PPRO_NT
conditional)

IRIX: platform SGI
HPUX: platform HP
SOLARIS: platform SUN
P_LINUX: linux on platform x86
PPRO_NT: Windows on platform x86
debug
debug or nodebug
debug: the sources are compiled in debug mode
(- G)
nodebug: the sources are compiled with the level
of optimization specified in the file of configuration

2.4.2 Construction of achievable - a key word make_exec

This procedure makes it possible to constitute an achievable module, by possibly overloading them bookshops of reference by "personal" sources FORTRAN or C (added routines, or modification of the routines of reference).

Key word
Associated parameter

make_exec yes

rep_tcl
name of repertory
Repertory containing the provided files tcl
fic_conf
file name
File of configuration specific to the platform
used
platform
IRIX, HPUX, SOLARIS, Plate-forme targets (used for compilation
P_LINUX or PPRO_NT
conditional)
IRIX: platform SGI
HPUX: platform HP
SOLARIS: platform SUN
P_LINUX: linux on platform x86
PPRO_NT: Windows on platform x86
debug
debug or nodebug
Debug: the sources are compiled in debug mode (-
G)
nodebug: the sources are compiled with some

level of optimization
rep_ref
name of repertory
Repertory containing the bookshops and the environment
of execution of reference (obtained by make_aster)
source
list names of
Repertories containing the files FORTRAN and/or C
repertories
"personal" with which one wants to overload
version of reference. The file names must
to have the extension .f or .c. Various names of
repertories must separated by the character |.
rep_obj
name of repertory
Repertory containing of the modules user objects,
the files must have the extension .o (Unix) or
.obj (Windows)
rep_lib
name of repertory
Repertory containing of the bookshops user, them
files must have the extension .lib
exec
file name
Name of the produced achievable file

The key words rep_obj and rep_lib are optional.

2.4.3 Construction of the catalogues compiled - key word make_cata

This procedure makes it possible to build the catalogues of orders and elements compiled to leave catalogues of reference of Code_Aster and files catalogues sources (in complement or in overload).
The operations of compilation of catalogue is based on the presence of a chart used at the time of management of configuration on the machine of reference, it is thus essential to preserve this chart when it exists and to add it in new the catalogues.

Key word
Associated parameter

make_cata yes

rep_trav
name of repertory
repertory used during the execution of Code_Aster
(to compile the catalogues); this repertory is
normally destroyed at the end of the procedure
platform
IRIX, HPUX, SOLARIS, platform targets (used for compilation
P_LINUX or conditional PPRO_NT)
IRIX: platform SGI
HPUX: platform HP
SOLARIS: platform SUN
P_LINUX: linux on platform x86
PPRO_NT: Windows on platform x86
fic_conf
file name
file of configuration specific to the platform used
rep_tcl
name of repertory
repertory containing the provided files tcl
rep_catalo
name of repertory
repertory containing the tree structure of the sources of
catalogues of reference
catalo
list names of
file directory catalogues of elements
repertories
"personal" with which one wants to overload them
catalogues of reference.

For the catalogues of order, the names of
files must have the extension .capy and include
chart #& MODIF or #& ADDITION.
For the catalogues of elements, the names of the files
must have the extension .cata and include the chart
%& MODIF or %& ADDITION. Various names of
repertories must separated by the character |.
unigest
file name
File used during the management of configuration and
which one exploits lines CATSUPPR to destroy
catalogues of corresponding elements [D1.02.01]
exec
file name
name of the achievable aster used to compile them
catalogues (it must exist)
rep_coco
name of repertory
catalogue compiled orders produces
elco
file name
catalogue compiled elements produces

Note:

The compilation of the catalogues of elements is an incremental operation which rests on
file cata_ele_pickled (built by make_aster).

2.4.4 Construction of a bookshop - key word make_lib

This procedure makes it possible to build a personal bookshop starting from source files (FORTRAN or C).
Key word
Associated parameter

make_lib
yes
debug
debug or nodebug
debug: the source are compiled in debug mode (- G)
nodebug: the source are compiled with some
level of optimization
platform
IRIX, HPUX, SOLARIS, Plate-forme targets (used for compilation
P_LINUX or conditional PPRO_NT)
IRIX: platform SGI
HPUX: platform HP
SOLARIS: platform SUN
P_LINUX: linux on platform x86
PPRO_NT: Windows on platform x86
fic_conf
file name
File of configuration specific to the platform
used
rep_tcl
name of repertory
Repertory containing the provided files tcl
source
list names of
Repertory containing the files FORTRAN and/or C
repertories
“personal” with which one wants to overload
version of reference. The file names must
to have the extension .f or .c (separated by 1 “L”)
rep_lib
name of repertory
Repertory recipient of the produced bookshop
lib_aster
name of the bookshop
Name of the produced bookshop deposited under rep_lib

2.4.5 Passage of a list of tests - key word make_test

This procedure makes it possible to launch a series of tests. An Aster test consists of a certain number files (of which has minimum a command file and a file of parameters) whose names are constituted starting from the name of the test and a suffix. The principal suffixes are as follows:

Suffix
Logical unit
Type of file
.com m fort.1
orders Aster
.mail fort.20
grid with the format Aster
.mgib fort.19
grid with the Gibi format
.msup fort.19
grid with the Ideas format
will.para -
parameters of execution
.mess fort.6
file MESSAGE
.resu fort.8
file RESULT
.erre fort.9
file ERROR
.27 fort.27
associated data file for
this test with the logical unit 27

The file containing the list of the tests must contain a name of test per line, all that follows the character % is ignored.

Key word
Associated parameter

make_test yes

rep_tcl
name of repertory
Repertory containing the provided files tcl
rep_trav
name of repertory
Repertory used during the execution of Code_Aster (for
to compile the catalogues); this repertory is normally

destroyed at the end of the procedure
liste_test file name
List tests with launching
rep_test
name of repertory
Repertory of the files of tests
rep_ref
name of repertory
Repertory of reference of the sources
rep_py
name of repertory
Sources
python overloading the environment
of execution
exec
file name
Name of achievable used
rep_coco
name of repertory
Catalogue compiled orders used
elco
file name
Catalogue compiled elements used

2.4.6 Passage of a study - key word make_etude

This procedure makes it possible to launch a study. An Aster study consists of a certain number of

files for which it is necessary to associate a logical unit and an indicator D or R to specify if the file or the repertory is in data and/or results. The parameters of execution passed overall behind the key word para (the list perhaps obtained while launching the achievable one with the parameter - H or - help).
If the value of the key word make_etude is positioned with not, the repertory of execution is prepared with there catalogues compiled, the environment python and data files in order to be able to launch the debugger. During the execution of the procedure, the line of ordering of launching of the achievable one Code_Aster is posted with all the parameters necessary.

Key word
Associated parameter

make_etude
yes or not
yes: the execution of Code_aster will be launched afterwards
recopy environment and data in
repertory of work.
not: the repertory of work is prepared in order to be able
to launch an execution with the debugger.
rep_tcl
name of repertory
Repertory containing the provided files tcl
rep_trav
name of repertory
Repertory used during the execution of Code_Aster (for
to compile the catalogues); this repertory is normally
destroyed at the end of the procedure
exec
file name
Name of achievable used
rep_coco
name of repertory
The catalogue of compiled orders contains used
elco
file name
Catalogue compiled elements used
base
name of repertory
Repertory of reception of the base Aster (given and
results)

d_base
name of repertory
Repertory of reception of the base Aster (given only)
r_base
name of repertory
Repertory of reception of the base Aster (results only)
d_ensi
name of repertory
File directory to the Ensight format in data
r_ensi
name of repertory
File directory to the Ensight format in results
para
character string
List parameters of execution

The various logical units are indicated in the following form:
fort.ij DR.: nom_de_fichier
where ij indicates the number of associated unit, at least one of the indicators D or R must be present.

Among the parameters of execution, it is essential to provide the size of the zone managed memory dynamically during the execution, the unit used is the méga word of entirety.
- memjeveux 16 thus correspond to 64 méga bytes on a platform 32 bits.

2.5
Files of parameters delivered at the time of the installation

Several files of parameters directly usable are delivered under the repertory tcl at the time of the installation, they were automatically adapted to the configuration of the user and can be immediately last in parameter of the procedure run_aster.

make_aster
allows to build a complete version in debug mode or not of bookshops, of achievable and the catalogues compiled.
make_exec
allows to build achievable starting from the source files FORTRAN and C (extensions .f or .c) deposited under the repertories dvp1/and dvp2/, it result is deposited under dvp/asteru.exe
make_cata
allows to build the catalogues of orders and elements compiled with

to leave the source files catalogues (extension .cata and .capy) deposited under the repertory dvp/, the results are deposited under dvp/order and dvp/elements
make_etude
allows to carry out an execution of Code_Aster with the files

asteru.exe, order and elements with the data deposited under study

---

make_test
allows to launch the execution of deposited under the /etude repertory with files asteru.exe, order and elements. The list of the tests used is deposited in the file study/liste_ct.
make_lib
allows to build a bookshop starting from sources located under /dvp, it result is deposited under /dvp/lib

These files can be recopied (like examples) then modified.

3 Example of use

3.1 File of configuration on platform x86 under linux

% Fichier of configuration associates has aster.tcl
%
% PPRO_linux version with supervisor python

% ===================
%
f77: /usr/bin/g77
DC: /usr/bin/gcc
lib: /usr/bin/ar - rv
link: /usr/bin/g77

python: /usr/local/bin/python2.1

rep_incl: /home/aster/STA6.2/include
path_hdf: /home/aster/MED2.0/hdf/lib
path_med: /home/aster/MED2.0/lib

opcc: - C - I /usr/local/include/python2.1
opcc_dbg: - C - G - I /usr/local/include/python2.1

opf77: - C - O3
opf77_dbg: - C - G

op_link: - Xlinker - export-dynamic - lieee - ldl - lpthread - lutil
lib_sys: - L/usr/local/lib/python2.1/config - lpython2.1 - L/usr/local/lib - ltk8.3 - ltcl8.3 lX11
lib_hdf: /home/aster/MED2.0/hdf/lib/libhdf5.a - lm - lz
lib_med: /home/aster/MED2.0/lib/libmed.a

3.2 File of parameters used to build achievable under linux

% parameters obligatory:
% ==========================
rep_tcl: /home/aster/STA6.2/tcl

% construction of achievable overloading bookshops:
% ==
fic_conf: /home/aster/STA6.2/tcl/conf/init_conf
make_exec: yes
source: /home/aster/STA6.2/dvp
rep_ref: /home/aster/STA6.2
exec: /home/aster/STA6.2/dvp/asteru.exe
debug: nodebug
platform: P_LINUX
Data-processing Descriptive handbook
D1.01 booklet: Environment of development and Tools
HI-75/01/005/A

3.3 File of parameters used to carry out an execution under linux

obligatory parameters:
==========================
rep_tcl: /home/aster/STA6.2/tcl

construction of achievable overloading bookshops:
===
fic_conf: /home/aster/STA6.2/tcl/conf/init_conf

make_etude: yes
rep_ref: /home/aster/STA6.2
rep_trav: /tmp/trav_aster
rep_coco: /home/aster/STA6.2/commande
elco: /home/aster/STA6.2/elements
exec: /home/aster/STA6.2/dvp/asteru.exe

para: - memjeveux 16 - reference mark none
fort.1 D: /home/aster/STA6.2/tests/ttnl100a.comm
fort.19 D: /home/aster/STA6.2/tests/ttnl100a.mgib
fort.6 R: /home/aster/STA6.2/tests/ttnl100a.mess
fort.8 R: /home/aster/STA6.2/tests/ttnl100a.resu
fort.9 R: /home/aster/STA6.2/tests/ttnl100a.erre
r_base: /home/aster/BASE
platform: P_LINUX

3.4 Example of call

% run_aster make_exec

% run_aster make_etude
Data-processing Descriptive handbook
D1.01 booklet: Environment of development and Tools
HI-75/01/005/A

Code_Aster ®
Version
6.0

Titrate:
Environment of development on autonomous platform: run_aster Date:
05/02/02
Author (S):
Key ***J.P. LEFEBVRE***
:
D1.01.02-B Page
: 12/12

Intentionally white left page.
Data-processing Descriptive handbook

Code_Aster ®
Version
5.0
Titrate:
Launching of an execution and re-use of the tools of asterix
Date:
02/04/01
Author (S):
J.P. LEFEBVRE*, C. MASSERET*
Key:
D1.03.01-C Page:
1/52
Organization (S): EDF/MTI/MMN, SAMTECH
Data-processing handbook of Description
D1.03 booklet:
D1.03.01 document
Launching of an execution on a waiter
centralized Aster and re-use of the tools
of asterix by other applications:
lib_exec_aster (4.0) and lib_asterix (4.0)

Summary:
This document describes the principles of the libraries lib_exec_aster and lib_asterix as well as the interfaces
functions placed at the disposal. These libraries make it possible to subject executions of Code_Aster on a waiter centralized Aster. The library lib_exec_aster makes it possible to subject an execution without
to require graphic environment. The library lib_asterix makes it possible to do it in one X11R5-Motif1.2 environment. It also makes it possible to use certain tools developed within the framework of
the interface X11-Reason of Code_Aster: asterix.
Paragraphs 1 to 4 of this note specify the limits of supply of the libraries.

Date:
02/04/01
Author (S):
***J.P. LEFEBVRE**, C. MASSERET*
Key:
D1.03.01-C Page:
2/52

Contents

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

The general principles of the tools of asterix as well as the operational requirements are described in the document
"Functional Description of the Interface of Aster on workstation" [bib1].
The delivered libraries are mainly used to subject in batch an Aster execution on a waiter
centralized Aster. Indeed the launching of Aster refers to scripts, with a specific configuration of
poured and with a manager of batch given. That thus does not make it possible to launch an execution for one
version known as "local" of Aster which is managed completely differently. Currently centralized waiters taken
in account are claster, clcraya and cldragon (knowing that the Aster access to clcraya is removed and that machine cldragon does not exist any more).

1
Principal tools proposed in the libraries
1.1 Library
lib_exec_aster

This library contains all the basic utilities to subject an execution batch of Aster to a waiter centralized Aster. All the utilities suggested are independent of a graphic environment.

1.1.1 Management of profiles of studies of the asterix type

A profile of study is a manner of describing the data files and results of an Aster execution thus that parameters of the execution (version Aster, memory, maximum time…). These files are created and recognized by asterix, but they are also used as support to characterize an execution. As they are stored with the ASCII format, it is simple to create them independently of asterix by respecting syntax given.

The provided tools make it possible to create and handle the profiles of study asterix.

1.1.2 Launching of an execution Aster

The tender of an Aster execution is done compared to a profile of study of the asterix type. Files allowing supervise the implementations are created on the object computer (Aster waiter) and on the machine of tender (local machine).

1.1.3 The management of the communications *remote* with or without ASURE

The communications between the machine of the user and the waiter of Aster execution are done by the intermediary orders UNIX remote (CCP, rsh). lib_exec_aster places at the disposal a certain number of tools to coat these orders. It is in particular possible to use them while crossing firebreak EDF ASURE.

1.2 Library

lib_asterix

This library takes again the tools of lib_exec_aster in a graphic context, and adds some a number of utilities developed for asterix.

1.2.1 A box of selection of file multi-machine: bsf

This box replaces the File Selection Box Reason. It makes it possible to sail on various machines by mechanisms Unix remote (remote Shell and remote Copy).

It implies that each user has a file $HOME/.rhosts

It makes it possible to publish a selected file. For that of the editors are pre defined.

The handled files are it in the form machine@user: nom_de_fichier.

1.2.2 Launching of an execution Aster

It is possible to subject the passage of an Aster study starting from a profile of study and of a profile of execution and
of some parameters.
Compared to lib_exec_aster, the subjected jobs are dealt with by the follower of job asjob.

1.2.3 Follow-up of jobs on the object computer

The follow-up of the jobs comprises mechanisms of management (interrogation, interruption, suppression,…) and one
graphic application (asjob) to reach these functionalities interactivement. Tenders of a study
Aster by the provided procedures are interfaced with this mechanism of follow-up.
This mechanism uses a repertory on the object computer ($HOME/flashor, by defect) and a repertory on workstation ($HOME/flash_Aster, by defect). One can modify these repertories of destination.

1.2.4 Execution of an interruptible order system

If one defined a Interrompre button according to recommendations' further given, the function sysint () allows to carry out an order system which can be stopped by the user by a simple click on
button To stop.

2 Delivery

On the machine claster, in /aster/interface/lib/, there is a repertory by supported environment.
Each repertory contains the shareable libraries, the files of resources and an example of use
of lib_exec_aster only and lib_asterix with XFaceMaker [§17].
The delivery includes/understands 4 shareable libraries and two files of heading:

File name	Description
lib/lib_asterix.sZ	shareable library lib_asterix
lib/lib_exec_aster.sZ	shareable library lib_exec_aster
lib/lib_util_exec_aster.sZ	library of utilities lib_exec_aster without graphic behavior (when one uses lib_exec_aster without lib_asterix)
lib/lib_util_exec_aster_X.sZ	library of utilities lib_exec_aster with graphic behavior (when one uses lib_asterix).
include/lib_asterix.h	File of heading for lib_asterix
include/lib_exec_aster.h	File of heading for lib_exec_aster

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

The shareable extension of the libraries depends on the operating system.

Example of use of lib_exec_aster, in the repertory ex_lib_exec_aster/:

File name	**Description**
A_LIRE.txt	Description of the stages to compile the example.
Makefile	File of make of the example.
asexec	Script of launching of the example.
asterix/include/	Files header of lib_exec_aster and lib_asterix (definition of the types, aggregate variables interfaces functions provided).
asterix/lib/	Shareable libraries compressed with gzip.
bin/asexecB	Achievable of the example created by the order make.
gunzip	Utility of decompression of the format gzip (.gz).
obj/	Repertory of the files objects.
src/asexec.c	Main program of the example.
zz.pret	File profile of study asterix of example.

Example of use of lib_asterix, in the repertory ex_lib_asterix/:

File name	**Description**
A_LIRE.txt	Description of the stages to compile the example.

Makefile
File of make of the example.
asjob
Script of launching of asjob.
asterix/bin/asjobB.gz
Achievable of the follow-up of the jobs compressed.
asterix/include/
Files header of lib_exec_aster and lib_asterix
(definition of the types, aggregate variables interfaces functions
provided).
asterix/lib/
Shareable libraries compressed with gzip.
asterix/rdb/
File of resources X for asjob and lib_asterix.
bin/exempleB
Achievable of the example created by the order make.
obj/
Repertory of the files objects.
example
Script of launching of the example.
src/exemple.fm
XFaceMaker file of the interface.
src/exemple.c
File C generated by XFaceMaker starting from the file
exemple.fm
src/exemple_pp.c
Main program of the interface.
xfm3.5/include/
Files header for the use of XFM.
xfm3.5/lib/libFm_c.a
Library XFM for the edition of the bonds of the application.
Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

2.1
On Sparc Sun under SOLARIS 2.x

Repertory of delivery: /aster/interface/lib/sun_solaris
The shareable libraries have as an extension .so.

2.2
On HP 9000 pennies HP-UX 10.x

Repertory of delivery: /aster/interface/lib/hp
The shareable libraries have as an extension .sl.

2.3
On SGI under IRIX 6.5

Repertory of delivery: /aster/interface/lib/sgi
The shareable libraries have as an extension .so.

3 Anomalies,
evolutions

All the noted anomalies and the desired evolutions must follow the circuit of *Code_Aster* (anomaly software, anomaly documentation, software evolution) via the interface asterix.

4
Use of the libraries

The libraries are in the form of shareable bookshops UNIX. They and are in the repertory rep_asterix/lib/.
They are delivered only in shareable version to allow a fast evolution of the mode of tender of *Code_Aster* without requiring intervention on the applications using it.

4.1 Use
of
lib_exec_aster

One will use lib_exec_aster without lib_asterix to be independent of the graphic environment X.
In this case there, it is necessary to associate the utilities without graphic behavior to him: lib_util_exec_aster.
The edition of the bonds of the application must thus have a reference of the type:
- l_util_exec_aster - l_exec_aster

4.2 Use
of
lib_asterix

To take account of the evolution of the library lib_asterix, and possibly of the resources X which him are associated, it is recommended to update the resources of the application in the following way (example on SUN):
sed - E “1, $ s/ZZZZB*/nomApplication*/” lib_asterix.rdb >/tmp/ressources.rdb
xrdb - merge /tmp/ressources.rdb
rm /tmp/ressources.rdb

The library lib_asterix was developed with the generator of XFaceMaker interface. As it is one shareable library, it does not integrate all the resources necessary in statics. To be able to carry out the edition of the bonds, the application using lib_asterix must also refer to the library libFm_c.a which is also delivered. lib_asterix refers to lib_exec_aster, and to have it graphic behavior of certain utilities of lib_exec_aster, it is necessary to refer to lib_util_exec_aster_X.

The edition of the bonds of the application must thus have a reference of the type:
- lFm_c - l_util_exec_aster_X - l_exec_aster - l_asterix.

4.3
Under Sun SOLARIS

In the script of launching of the interface it is necessary to enrich variable LD_LIBRARY_PATH.
Example in C-Shell:
setenv LD_LIBRARY_PATH $ {LD_LIBRARY_PATH}: rep_asterix/lib
rep_asterix/interfaceB

For the compilation of the files including the file lib_exec_aster.h or lib_asterix.h, it is necessary to transmit with the compiler the directive - D_SOLARIS.

4.4 Under
HP-UX

In the script of launching of the interface it is necessary to enrich variable SHLIB_PATH.
Example in C-Shell:
setenv SHLIB_PATH $ {SHLIB_PATH}: rep_asterix/lib
rep_asterix/interfaceB

For the compilation of the files including the file lib_exec_aster.h or lib_asterix.h, it is necessary to transmit with the compiler the directive - D_HP.

4.5 On

SGI
In the script of launching of the interface it is necessary to enrich variable LD_LIBRARYN32_PATH.
Example in C-Shell:
setenv LD_LIBRARYN32_PATH $ {LD_LIBRARYN32_PATH}: rep_asterix/lib
rep_asterix/interfaceB
For the compilation of the files including the file lib_exec_aster.h or lib_asterix.h, it is necessary to transmit
with the compiler the directive - D_SGI.

---

5 Aggregate variables

5.1 lib_exec_aster

lib_exec_aster uses a certain number of aggregate variables. Certain variables are interesting for applications other than asterix. Those will be presented. However the majority are used for management intern of the tools of asterix, therefore it will have to be made sure that there is not conflict with the aggregate variables of
applications which want to use the tools described in this document (starting from the file lib_exec_aster.h).

Name	**Type**	**Function**
Hostname	tank *	Name of the machine from where the interface is launched
LOGINUSER	tank *	To use Unix which launched the interface

TMPDIR_INTERFACE
tank *
Repertory of work
FLASH_STATION
tank *
Name of the repertory on station for the follow-up of the jobs on the machine
of execution
HOMEHOST
tank *
$HOME of LOGINUSER on hostname
MACHINE_EXEC
tank *
Object computer of *Code_Aster*
OIAUser
tank *
To use on the object computer of *Code_Aster*
RSH
tank *
Order to carry out an order remote from
machine hostname (rsh or remsh)

5.2
lib_asterix

lib_asterix also uses a certain number of aggregate variables. They are declared in the file lib_asterix.h.

Name
Type
Function
CMDxedit
tank *
Order launching of the editor xedit until the 2.2
CMDvi
tank *
Order launching of editor VI until the 2.2
CMDsedit
tank *
Order launching of the editor sedit until the 2.2
CMDemacs
tank *
Order launching of the emacs editor until the 2.2
CMDtextedit
tank *
Order launching of the editor textedit until the 2.2
Data-processing handbook of Description

6 Services of lib_exec_aster

6.1 Initialization of the library

Interface:

```
/*-----------------------------------------------------------------------
Initialistation of lib_exec_aster
- treatment of the arguments not X-Reason
-----------------------------------------------------------------------
*/
void Init_lib_exec_aster
(
int argc,/* - > Nb of argument of the line of order *
tank ** argv,/* - > list of the arguments *
String to tmpdir/* - > Prefix of the repertory for the files
temporary *
);
```

Interpretation of the arguments passed to the interface to detect the following parameters:

Parameter	**Function**
- chatterer	Allows to return the echo of the Shell orders launched starting from the function sysint (). This echo is sent on stdout in a context not X and one fenestrate if it is in the context lib_asterix.
- noexec	Allows not to carry out the Shell orders launched starting from the function sysint ().
- host nom_machine	

Allows to use another name of machine that the name returned by the order
uname (is useful when the name of the machine is not the same one as that declared for
authentification EDF)
- rep_flash *reference mark*
Repertory in which the files of message are written on the local machine
(cf [§7.5.4] for a description of these files).
- rep_flash_exec reference mark
Repertory in which the files of message are written on the machine
of execution (cf [] for a description of these files).
- rep_tmp to tmpdir
Repertory of work on the local machine.
parameters ASURE
Cf [§6.4.1].
This function of initialization carries out the creation of the repertory of work whose radical is to tmpdir and to which one
add _LOGINUSER. Attention if one wants to launch several simultaneous sessions of the same application with
the same one to use, it is necessary to give a single radical (based on the date or the number of process for example) for
that there is not confusion on the names of the job stream files. The complete name of this repertory of work is
contents in variable TMPDIR_INTERFACE. The analysis of the file $HOME/.rhosts is carried out in this
phase.

6.2 To leave
the application

To properly release the resources allocated by lib_exec_aster, in particular files and the repertory

of work, the Quitter_lib_exec_aster function () is available.
Interface:

```
/*----------------------------------------------------------------------
Exit of lib_exec_aster with removal of the temporary files.
----------------------------------------------------------------------
*/
void Quitter_lib_exec_aster
(
bool SupprRep/* - > Suppression of the temporary repertory *
);
```

6.3

Local file/File remote

The use of the bsf implies that the user can choose a local file (to even use and even machine) or distant (to use or machine different). To take account of the organizations with a disc file server for the user data, the reference to a machine and one to use local are masked. There is then convention following:

· *******: to use which launched the application,

· *******: machine from where the application was launched.

As the treatment for a local file and a distant file is different, lib_exec_aster proposes to encapsulate the detection of a local file by the following functions:

Functions which answer TRUE if the file is considered in room and FORGERY if not:

```
/*----------------------------------------------------------------------
To determine if a file is local or remote.
If the couple to use/machine is not detects local, but that one of the 2
values is generic (*******) one renvoit the value reele in the place.
----------------------------------------------------------------------
*/
bool Local
(
String usr,/* <-> to use *
String mac/* <-> machine *
); /* < - TRUE so local *
/*----------------------------------------------------------------------
To determine if a file is local or remote
If the couple to use/machine is not detects local, but that one of the 2
values is generic (*******) one renvoit the value reele in the place.
----------------------------------------------------------------------
*/
bool FichierLocalNFC
(
String fic/* - > Name file in form user@machine: nom_fichier *
); /* < - TRUE if local file *
```

Data-processing handbook of Description

D1.03 booklet:
HI-75/01/005/A

Function to test if a file is regarded as room compared to another reference that to use it and machine of launching of the application (to test in particular if a file is local in the context of one profile):

```
/*------------------------------------------------------------------------
To determine if a file is local compared to a reference.
------------------------------------------------------------------------
*/
bool LocalRef
(
String usr,/* <-> to use *
String mac,/* <-> machine *
String usrRef,/* - > to use of reference *
String macRef/* - > machine of reference *
); /* < - TRUE so local A a reference *
```

Function to transform the generic name into real name:

```
/*------------------------------------------------------------------------
If to use it or the machine have the generic name it east replaces
by the real name.
------------------------------------------------------------------------
*/
void NomLocal
(
String usr,/* <-> to use *
String mac/* <-> machine *
);
```

Function to transform the real name into generic name:

```
/*------------------------------------------------------------------------
If to use it or the machine have the name real it east replaces
by the generic name.
------------------------------------------------------------------------
*/
```

void AliasLocal
(
String usr,/* <-> to use *
String mac/* <-> machine *
);
6.4
protected CCP and rsh via ASURE
lib_exec_aster takes account of the constraints of ASURE. For that it is necessary to pass to him from the parameters on
line of order and to envisage a file of declaration of the topology of the machines which are defined in file $HOME/.rhosts. cf [bib4] and [bib5].
6.4.1 Parameters ASURE for lib_exec_aster
Parameters ASURE for lib_asterix:
Parameter
Function
- user_asure to use
Count entering on footbridge ASURE (external - > EDF).
- auth_asure auth
Authenticating for entering account ASURE (external - > EDF): word of
pass dynamic.
- cf_asure CFA
Name of the cut fire (footbridge ASURE) clasure.edf.fr by defect
- display_asure disp
display in relayage on ASURE (is used only in the context
lib_asterix).
- user_asure_sortant user_s
Outgoing account on the passerel E ASURE (EDF - > external).
- dns_domain field
Field DNS of the user.
Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

Code_Aster ®
Version
5.0
Titrate:
Launching of an execution and re-use of the tools of asterix
Date:
02/04/01
Author (S):
J.P. LEFEBVRE, *C. MASSERET*
Key:

6.4.2 Workspace file ASURE for lib_exec_aster

If one of these parameters is detected on the line of order, lib_exec_aster seeks the file $HOME/.topo_asure on the object computer of the application. It indicates if a machine is reached via ASURE or locally. Format of this file: machine local asure |

Example:
claster asure
clcraya asure
Hawaii room
Tahiti room

Information and the file of topology of the machines can be prepared with the asasure application.
If lib_exec_aster is used with one of the parameters relating to ASURE, for any CCP or all rsh of library, lib_exec_aster determines if the implied machine is used via ASURE (entering or outgoing).
If it is the case the order is transformed with format ASURE [bib4] [bib5] starting from transmitted information.
The other applications can profit from the same mechanism using the functions described below.

6.4.3 ENTERING ASURE (external - > EDF)

```
/*-----------------------------------------------------------------------
Construction of the part of the order rsh identifying the account and
the distant machine by taking account of Asure. (ENTERING)
------------------------------------------------------------------------
*/
String carsh
(
String usr,/* - > to use distant *
String mac/* - > distant machine *
); /* < - distant identification with or without Asure *
```

If lib_exec_aster is not launched with parameters ASURE or if it is launched with ASURE and that mac is local, carsh re-examines the chain: “mac - L usr”. If lib_exec_aster is launched with ASURE and that
mac is accessible via ASURE, carsh returns the chain:
“cf_asure - L user_asure-auth_asure_code: usr@mac”.

```
/*-----------------------------------------------------------------------
Construction of the part of order CCP identifying the account and
the distant machine by taking account of Asure. (ENTERING)
------------------------------------------------------------------------
*/
String carcp
(
String usr,/* - > to use distant *
String mac/* - > distant machine *
); /* < - distant identification with or without Asure *
```

If lib_exec_aster is not launched with parameters ASURE or if it is launched with ASURE and that
mac is local, carcp re-examines the chain: “usr@mac”. If lib_exec_aster is launched with ASURE and that mac
is accessible via ASURE, carcp returns the chain:
“user_asure-auth_asure@cf_asure_code: usr@mac”.

6.4.4 OUTGOING ASURE (EDF - > external)

For reasons practise (batch) the password ASURE EDF - > external is not a password
dynamics (it must serve during the night and the weekend). It must be described in the file $HOME/. authAsure on
object computer. It has the following format: user_sortant_ASURE mot_de_passe_associe. With each
use of this file to calculate the password recognized by ASURE (with MD5) its Unix rights are
positioned with - rw------- to maintain the confidentiality of the fixed password which must obligatorily be
writing in light.

```
/*-----------------------------------------------------------------------
Construction of the part of the order rsh identifying the account and
the distant machine by taking account of Asure. (OUTGOING)
-----------------------------------------------------------------------
*/
String carshS
(
String usr,/* - > to use distant *
String mac,/* - > distant machine *
String macA/* - > machine ASURE *
); /* < - distant identification with or without Asure *
```

If lib_exec_aster is not launched with parameters ASURE or if it is launched with ASURE and that
mac is local, carshS re-examines the chain: “mac - L usr”. If lib_exec_aster is launched with ASURE and

that mac is accessible via ASURE, carshS returns the chain:
"cf_asure - L user_asure_sortant- `/aster/adm/tool/asmd5f
user_asure_sortant `: usr@mac ".

```
/*------------------------------------------------------------------------
Construction of the part of order CCP identifying the account and
the distant machine by taking account of Asure. (OUTGOING)
--------------------------------------------------------------------------
*/
String carcpS
(
String usr,/* - > to use distant *
String mac,/* - > distant machine *
String macA/* - > machine ASURE *
); /* < - distant identification with or without Asure *
```

If lib_exec_aster is not launched with parameters ASURE or if it is launched with ASURE and that mac is local, carcpS re-examines the chain: "usr@mac". If lib_exec_aster is launched with ASURE and that
mac is accessible via ASURE, carcpS returns the chain:
"user_asure_sortant- `/aster/adm/tool/asmd5f user_asure_sortant `@cf_asure: usr@mac".

6.4.5 DISPLAY
ASURE

The function cadisp () returns the display on which lib_exec_aster for the local machines is launched and display ASURE for the accessible machines via ASURE. This information has direction only when one is in a context X.

```
String cadisp
(
String mac/* - > distant machine *
); /* < - display has to use *
```

6.5

Management of profile of study or execution asterix

6.5.1 Structure of data for a profile of study or execution

```
typedef struct t_desc_fic TDescFic;
struct t_desc_fic {
standard tank [16]; /* Type of the file
*/
tank name [1024]; /* name of the file
*/
tank to use [32]; /* to use of the file
*/
tank machine [32]; /* machine of the file
*/
int ul; /* Logical Unit of file 0 if there does not need any
*/
bool data; /* Flag given OUI/NON
*/
bool result; /* Flag result OUI/NON
*/
bool aEditer; /* Flag to publish OUI/NON
*/
TDescFic *suiv; /* Pointeur on the following
*/
};
/* Pointeur on a descriptor of file *
typedef TDescFic *PTDescFic;
/* Allocation of a descriptor of file *
#define TDescFicAlloc (TDescFic *) malloc (sizeof (TDescFic))
```

Since version 3.5 of lib_asterix, a new Flag is managed to take into account files
compressed with the utility of the public domain gzip (GNU). Not to modify the structure of data, it
new state is detected on the extension of the file name. If this name comprises the extension ".gz", it is considered
as compressed.

6.5.1.1 Creation of a chained list profile starting from a file

Interface:

```
/*-----------------------------------------------------------------------
To open a profile on any machine to transform it
in list chainee.
-----------------------------------------------------------------------
*/
String OuvrirProfil
(
```

PTDescFic *rac,/* < - Root of the list *
String mac,/* - > machine or is the file *
String usr,/* - > to use to reach the file *
String reference mark,/* - > repertory of the file *
String fic/* - > name of the file *
); /* < - error message *

To open a profile of study or execution on any machine and to transform it into chained list.
This function can return an error message if it detects an anomaly. In the contrary case it return a null string ("\ 0").

6.5.1.2 Safeguards of a list chained profile on a file

Interface:

/*---
To record a profile. Transform a list chainee
in file on any machine.
----
*/
String EnrProfil
(
PTDescFic *rac,/* - > Root of the list *
String nfc/* - > File name composes of the profile
in the form user@machine: nom_de_fichier *
); /* < - error message *

This function can return an error message if it detects an anomaly. In the contrary case it return a null string ("\ 0").

6.5.1.3 Management of a chained list profile

Allowance of a structure of the descriptor type and initialization with reference of the pointer:

/*---

Allocate a structure of the descriptor type of file, initializes it
with the parameters, and a pointer returns passes on this
structure.
----
*/
PTDescFic ConsDescFic
(
String typ,/* - > Standard of the file *
String name,/* - > name of the file with the path *
String usr,/* - > to use to reach the file *
String mac,/* - > machine on which is the file *
int ul,/* - > Logical Unit FORTRAN of the file *
bool fd,/* - > flag given *
bool Fr,/* - > flag result *
bool Fe/* - > flag editable *
); /* < - Pointer on the new structure *
Modification of a structure descriptor starting from its pointer (only the nonnull parameters overload):
/*--
Modife a structure of the descriptor type of file,
initializes with the parameters passes. Surgarge only with
nonnull parameters.
----
*/
void ModifDescFic
(
PTDescFic Pt,/* - > Pointer on the structure *
String typ,/* - > Standard of the file *
String name,/* - > name of the file *
String usr,/* - > to use for accerder with the file *
String mac,/* - > machine of the file *
int ul,/* - > Logical Unit FORTRAN *
bool fd,/* - > flag given *
bool Fr,/* - > flag result *
bool Fe/* - > flag editable *
) ;

Insertion at the head of list of a structure descriptor:

```
/*-----------------------------------------------------------------------
To at the head insert a file in the list chainee of a profile of study.
-----------------------------------------------------------------------
*/
void InsDebPEtude
(
PTDescFic *rac,/* - > Root of the list *
PTDescFic pdes/* - > Structure has to add at the head *
);
```

Insertion at the end of the list of a structure descriptor:

```
/*-----------------------------------------------------------------------
To insert a file in the list chainee of a profile of study in tail.
-----------------------------------------------------------------------
*/
void InsFinPEtude
(
PTDescFic *rac,/* - > Root of the list *
PTDescFic pdes/* - > Structure has to add in tail *
);
```

Insertion with a given position of a descriptor (if the position is greater than the length of the list insertion
in tail, if the position is lower than 1 insertion at the head):

```
/*-----------------------------------------------------------------------
To insert a file in the list chainee of a profile of study
has a given position.
If the position is supperieure with the length of the list
Insertion in tail.
If the position is lower A 1 insertion at the head.
-----------------------------------------------------------------------
*/
void InsPosPEtude
(
PTDescFic *rac,/* - > Root of the list *
PTDescFic pdes,/* - > Structure has to add in pos *
```

int pos/* - > Position of insertion *
);
Removal of a descriptor to a given position:
/*--
Removal of a file in a profile of study
----
*/
void SupPosPEtude
(
PTDescFic *rac,/* - > Root of the list *
int pos/* - > Position of insertion *
);
Release of the memory taken by a chained list profile:
/*--
To release the place memory occupied by a profile of study (List chainee)
----
*/
void FreeProfil
(
PTDescFic *rac/* - > Root of the list *
);

6.5.2 Standardization of a profile

With the concept of local file generic, it is delicate to build Shell orders which will be carried out on another machine. As one should not either remove the references to the generic names, the function DupNormaProfil will make it possible to duplicate a profile by standardizing the generic names. After use of this temporary profile for the construction of the orders, it can be released.

Interface:

```
/*-----------------------------------------------------------------------
Create a new structure profile while remplacant
to use and the generic machine by LOGINUSER and hostname
-----------------------------------------------------------------------
*/
PTDescFic DupNormaProfil
(
PTDescFic *rac/* - > Root of the list *
); /* < - Root on the profile standardizes *
```

DupNormaProfil takes as reference LOGINUSER and hostname, which is not appropriate for a profile opened on another machine. There is thus also DupNormaProfilRef to carry out the same operation compared to another reference.

```
/*-----------------------------------------------------------------------
Create a new structure profile while remplacant
to use and the generic machine by usrRef and macRef
-----------------------------------------------------------------------
*/
PTDescFic DupNormaProfilRef
(
PTDescFic *rac,/* - > Root of the list *
String usrRef,/* - > Machine of reference *
String macRef/* - > to use of reference *
); /* < - Root on the profile standardizes *
```

6.6

The function sysint ()

The function sysint () is available with lib_exec_aster and lib_asterix. According to the context it has one different behavior. Within the framework of a nongraphic execution, it makes it possible to centralize them
orders systems carried out, and to take into account the parameters (- talkative and - noexec).

Interface:

```
/*-----------------------------------------------------------------------
Launch a Shell order.
-----------------------------------------------------------------------
*/
int sysint
(
String cmdstring/* - > order has soummettre as Bourne Shell *
);
```

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

6.7
Tender of a study Aster

The tender of an execution of *Code_Aster* is done in general starting from a profile of study and a profile of execution. According to interfaces' suggested, the profiles must be in the form of a chained list where each
element is a structure of description of file, or indicated by a file name.

All the functions described here have a different behavior if they are called only via lib_exec_aster or lib_asterix. If they are called via lib_asterix, there is coupling with the application of follow-up of the job: asjob.

6.7.1 Creation/tender of a script of execution

This function integrates the standardization of the profiles (DupNormaProfil). It builds the orders for to carry out *Code_Aster* according to the profiles given. If the tender of the built job is required, it is envoy on the object computer (machE) via using it userE.

From version 3.2 of lib_asterix the call evolved/moved much. The object computer and to use it of execution beings passed in parameters must. Are also added the parameters ficunic, origin (for the tracing of the uses of *Code_Aster*) and suivi_int (for the interactive follow-up of the execution). For
to avoid any confusion, the name of this function changed (ConsJob () - > AsConsJob ()).

Interface:

```
/*-----------------------------------------------------------------------
Construction of a job has to subject to the object computer for launching
Aster
-----------------------------------------------------------------------
*/
String AsConsJob
(
PTDescFic *RacE,/* - > Root of the list profile of Study *
PTDescFic *RacX,/* - > Root of the list profile of Execution *
String worm,/* - > Version Aster STA2 NEW2…. *
int tps,/* - > Time in S for the job *
int tpg,/* - > Time in S for the management of the job *
```

int mem,/* - > Memory for the job in Mmots *
int cpt,/* - > Account of charge *
String classifies,/* - > Classe job batch nbatch wbatch *
String nomjobp,/* - > Name of the job *
bool aq,/* - > Safeguard aq of the data files *
bool exec,/* - > Flag execution or not *
String userE,/* - > to use on the object computer *
String machE,/* - > object computer *
String ficunic,/* - > file script by defect in the form
user@machine: repertory/fic_script
or "" if the file is already defined
in the profile of study *
String origin,/* - > application of origin *
bool suivi_int/* - > followed interactive of the execution *
); /* < - Error message *
This function can return an error message if it detects an anomaly. In the contrary case it
return a null string ("\ 0").
If the pointer on the profile of execution is different from NULL and that it contains a file of the unic type, then it
file is updated. If there does not exist, it is created with its name by defect: REP_ETUDE/nom_etude.unic and
addition with the profile of study.

If there is no profile of study and a profile of execution, the same test is carried out. If there is a file unic it is put
up to date, if not it is created with its name by defect: REP_EXEC/nom_exec.unic and addition with the profile of execution.
If the flag exec is FALSE the file unic is replaced or created but is not subjected to the object computer.

If the flag exec is TRUE, it in is subjected to the object computer, in connection with the follow-up of the jobs.
Safeguard AQ is the recopy of the command file (comm), of the file of messages (mess) and of the file results (resu) by adding to their respective names the date in the form: _jj-mm-aa_hh.mm.ss+.
Since version 3.6, a new interface is added to take into account accounts of charge alphanumeric (as on SSU) and to turn over to the user the identification number of the subjected job.

```
/*------------------------------------------------------------------------
Construction of a job has to subject has Al object computer for launching
Aster.
To take account of the SSU the account of charge is chains of bus.
instead of an entirety. The num of the subjected job east turns over.
--------------------------------------------------------------------------
*/
String AsConsJobNum
(
PTDescFic *RacE,/* - > Root of the list profile of Study *
PTDescFic *RacX,/* - > Root of the list profile of Execution *
String worm,/* - > Version Aster STA2 NEW2…. *
int tps,/* - > Time in S for the job *
int tpg,/* - > Time in S for the management of the job *
int mem,/* - > Memory for the job in Mmots *
String scpt,/* - > Account of charge *
String classifies,/* - > Classe job batch nbatch wbatch *
String nomjobp,/* - > Name of the job *
bool aq,/* - > Safeguard aq of the data files *
bool exec,/* - > Flag execution or not *
String userE,/* - > to use on the object computer *
String machE,/* - > object computer *
String ficunic,/* - > file unicos by defect in the form
user@machine: repertory/fic_unicos
or "" if the file is already defined
in the profile of study *
String origin,/* - > application of origin *
bool suivi_int,/* - > followed interactive of the execution *
int numId/* < - number of the subjected job *
); /* < - Error message *
```

6.7.2 Tender of a script of execution

Recopy a script of execution on the object computer (machE) and submits it to the manager of batch by the intermediary of using userE. Also transmits information to the follow-up of the jobs.
From version 3.2 of lib_asterix, the call evolved/moved much. The object computer and to use it of execution beings passed in parameters must. The parameter origin (for the tracing is also added uses of *Code_Aster*). To avoid any confusion the name of this function changed (SubaFicUnic () - > AsSubaFicUnic ()).

Interface:

```
/*------------------------------------------------------------------------
Recopy of a Unicos script on the object computer (MACHINE_EXEC) and
tender with qsub with using it OIAUser. Transmit information to
follow-up of the jobs.
------------------------------------------------------------------------
*/
void AsSubaFicUnic
(
String err,/* < - error message *
String umac,/* - > machine or is script *
String uusr,/* - > to use to recopy script *
String urep,/* - > repertory or is script *
String ufic,/* - > name of the file script *
String nomjobp,/* - > name has to give to the job *
String usrExec,/* - > to use to carry out the job *
String macExec,/* - > object computer *
String origin/* - > application which soummet the job *
);
```

Since version 3.6, a new interface is added to take into account accounts of charge alphanumeric (as on SSU) and to turn over to the user the identification number of the subjected job.

```
/*------------------------------------------------------------------------
Recopy of a script on the object computer (MACHINE_EXEC) and
tender with the manager of batch. Transmit information to
follow-up of the jobs. The num of identification of the subjected job east turns over.
------------------------------------------------------------------------
*/
```

```
void AsSubaFicUnicNum
(
String err,/* < - error message *
String umac,/* - > machine or is script *
String uusr,/* - > to use to recopy script *
String urep,/* - > repertory or is script *
String ufic,/* - > name of the file script *
String nomjobp,/* - > name has to give to the job *
String usrExec,/* - > to use to carry out the job *
String macExec,/* - > object computer *
String origin,/* - > application which subjects the job *
String cpt,/* - > account of charge
("0" to have the account by defect) *
String fictail,/* - > file of message for the tail *
int numNQS/* < - identification number of the job *
);
```

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

6.8 Tender starting from a file profile

This function was more especially fulfilled to subject an execution without coupling with the follow-up of jobs, starting from a file of profile of study of the asterix type. The file of profile must contain a reference to one file of valid parameters (version Aster, time max,…).

Interface:

```
/*------------------------------------------------------------------------
Execution without environment X of a study Aster starting from a profile
of study of the type asterix indicates by a file name. If the file name
give is into relative, the file is research from the current directory.
The file profile must contain a reference has a file parameter.
------------------------------------------------------------------------*/
String AsExecFicProfil (
String nfic,/* - > name of the file profile of the asterix type *
String napp,/* - > name of the application which soummet the job *
String flocal,/* - > repertory flashor locally
optional (one can put (String) 0) *
String fexec,/* - > repertory flashor on object computer
optional (one can put (String) 0) *
int *njob/* < - number of the subjected job (if =0, there is a problem) *
); /* < - information or error message has desallouer
with free () if one wants to recover the place memory
= 0 if there is no detectee error nor of message
has to transmit has the user *
```

6.9 Utilities in bulk

6.9.1 Version of lib_exec_aster

To know the version of the bookshop used.
Interface:
String Version_lib_exec_aster (void);

6.9.2 basename

To ensure the same functionality as the order system Unix éponyme: to remove the way in a name of file (between the last character “/” and the end of the chain).
Interface:

```
/*----------------------------------------------------------------------
To provide the function of the order Unix eponyme.
----------------------------------------------------------------------
*/
void basename
(
String path,/* - > way has to analyze *
String bases/* < - extracted name *
);
```

6.9.3 dirname

To ensure the same functionality as the order system Unix éponyme: to extract the way in a name of file (between the beginning the chain and last character “/”).
Interface:

```
/*----------------------------------------------------------------------
To provide the function of the order Unix eponyme.
----------------------------------------------------------------------
*/
void dirname
(
String path,/* - > way has to analyze *
```

String to dir/* < - extracted way *
);

6.9.4 File without its extension: QuelBase

Return a file name without its extension with its complete way (removes the characters between the last "." and end of the chain).
Interface:
/*---
To return the name of a file without its extension.
----
*/
String QuelBase
(
String fic/* - > file name *
); /* < - name without extension *

6.9.5 Radical of a file: radical

Return a file name without its extension and its way (extracted the characters between the last "/" and it the last ". ").
Interface:
/*---
Return the "radical" of a file name =
all characters ranging between the last "/", if it
exist, and the last ". ", if there exists.
Take account of the complete names: user@machine: fic
----
*/
Radical String
(
String exp/* - > chains from which it is necessary to extract a radical *
); /* < - radical of the chains *

6.9.6 Extension of a file: QuelExt

Return the extension of a file (extracted the characters between the last "." and end of the chain).
Interface:
/*---
To return the extension of a file.
----
*/
String QuelExt
(
String fic/* - > name of the file *
); /* < - extension *

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

6.9.7 To compose a file name to format CCP: compNFC

To create a made up file name to format CCP: user@machine: nom_de_fichier. If names credits to use it and the machine are used, they are changed into real names.

Interface:

```
/*------------------------------------------------------------------------
Create a file name composes in the form:
user@machine: nom_du_fichier_avec_path
------------------------------------------------------------------------
*/
void compNFC
(
String nfc,
/* < - Name of the file composes *
String usr,
/* - > To use *
String mac,
/* - > Machine *
String reference mark,
/* - > Path *
String fic
/* - > File name *
);
```

6.9.8 To break up a file name to format CCP: decompNFC

To break up a file name to format CCP: user@machine: nom_de_fichier.

Interface:

```
/*------------------------------------------------------------------------
Break up a file name composes of the form:
user@machine: nom_du_fichier_avec_path
------------------------------------------------------------------------
*/
void decompNFC
```

(
String nfc,/* - > Nom of the file composes *
String usr,/* < - To use *
String mac,/* < - Machine *
String ficsel/* < - complete File name *
);

6.9.9 To join a name of repertory and a file name: Ficsel

To compose a file name de starting from its way and a name (by putting one "/" and only one enters them two).

Interface:

/*---
To give a complete name of file starting from a repertory and
of a file name (to put one/at the medium - and only)
----
*/
String Ficsel
(
String reference mark,/* - > name of repertory *
String fic/* - > file name *
); /* < - name of repertory/file name *

6.9.10 "Standardization" of a character string: strlennor

To remove the parasitic and "invisible" characters that one can recover in a field éditable Motif. Suppression of the white characters "", tabulation" \ you and carriage return "\. Also return the length chain thus "standardized".

Interface:

/*---
Return Nb of characters of chains without counting:

- white,
- tabulations,
- end of line,

The chains last is renvoyee by removing the characters
preceding.
Allows to know if a field is “apparement empty”.
----
*/
int strlennor
(
String word/* <-> chains has to standardize *
); /* < - length of the chains normalisee *

6.9.11 Length of a “standardized” chain: strlennor2

Return the length which the chain “standardized” with the same criteria as for strlennor would have but without
to modify the chain. Is used to know if a chain contains “meaning” characters.
Interface:
/*---
Return Nb of characters of chains without counting:

- white,
- tabulations,
- end of line,

The chains last is not modifiee

----
*/
int strlennor2
(
String word/* - > chains has to analyze *
); /* < - length of the chains normalisee *

6.9.12 Contents of a file in a chain: filestr

To put the contents of a file in a character string. The file must be local. The chain is
allocated dynamically, it is thus advised désallouer (by a free ()) when one needs some more.
Interface:
/*---
To see a file to put it in character strings
(allouee for the circumstance).
----
*/
String filestr
(
String EDF/* - > Nom of the file has to read *
); /* < - chains alouee by malloc containing the file *

6.9.13 Copy file or repertory between machines

```
/*------------------------------------------------------------------------
Copy of a file or a repertory whatever the machine
of departure and the machine of destination.
------------------------------------------------------------------------
*/
String CopierConf
(
String Susr,/* - > To use of the source file *
String Smac,/* - > Machine of the source file *
String srep,/* - > Repretoire of the source file *
String sfic,/* - > Source file *
String Dusr,/* - > To use of the file destination *
String Dmac,/* - > Machine of the file destination *
String drep,/* - > Repertory of the file destination *
String dfic,/* - > File destination *
bool conf/* - > Flag of confirmation or not of crushing
of an existing file *
); /* < - Mesage of error *
```

7
Services of lib_asterix

7.1
Initialization of the library

Interface:

```
/*------------------------------------------------------------------------
Initialistation of the lib_asterix and in particular of the bsf
- treatment of the arguments not X-reason of lib_asterix
```

----
*/
void Init_lib_asterix
(
int argc,/* - > Nb of argument of the line of order *
tank ** argv,/* - > list of the arguments *
String to tmpdir,/* - > Prefix of the repertory for the files
temporary *
Widget topw/* - > ToplevelShell *
);
This function calls upon Init_lib_exec_aster (). It adds just initializations specific to the environment X-Reason like the loading of the police forces and pixmap necessary to the bsf.
This initialization can be done only when Widget toplevel is known.

7.2 To leave
the application

To properly release the resources allocated by lib_asterix, in particular files and the repertory of work, the Quitter_lib_asterix function () is available.

Interface:
/*---
Exit of the lib_asterix with removal of the temporary files.
----
*/
void Quitter_lib_asterix
(
bool SupprRep/* - > Suppression of the temporary repertory *
);
This function calls upon Quitter_lib_exec_aster ().

7.3
Box of selection of file: bsf

It is possible to have several bsf in an application. In general there will be a bsf to carry out them selections (in modal dialogue) and one or more bsf not blocking (dialogues about it nonmodal). Each bsf created is independent and has its own structures. However the mechanism of remanence of repertories used is common to all the bsf of the application.

From version 3.0 of the lib_asterix the concept of edition of a file extended to the execution from one order Unix, definite by the user, with like parameter the selected file. The Editer button is identical but it is surmounted by a Commande menu… allowing for choice of an order among those defined. A Préférence button… makes it possible to add, modify or remove an order (within the limit of 20).

These orders are preserved in the file preference $HOME/.prefbsf. With the initialization of the bsf it file is read to update the corresponding structures.

It is possible to refer to the Unix orders of the bsf by creating lists of PushButton giving it name of the orders. These lists can beings integrated in of PullDownMenu or OptionMenu. These lists are updated at each call of the window of preferences of the bsf. (Example: choice of different types of editors in the panel of option of asterix.)

From version 3.2 a new mechanism on the remanent repertories is installed. Two types of repertories are now taken into account in remanence: “fixed” repertories and “volatile” repertories. The fixed repertories are positioned at the head small Répertoire…, and they are in video reverse (white on bottom black). They do not change a position when they are called upon. The volatile repertories correspond to remanence of the preceding versions. With each time one uses a repertory it is preserved. If there existed already it passes in first position, if not it is inserted in first position and the other repertories are shifted downwards. If there is no more place (15 + $HOME) the oldest repertory disappears. To define the repertories fixed it is necessary to use the window of definition of the preferences of the bsf (Préférences button…). The choice of one machine reveals the list of the associated repertories. By the usual mechanisms (To add, Modify, To remove) one can update the list. So that the modifications are taken into account it is necessary to validate (with the button of the same name) for each machine. It is possible to rearrange the list with the buttons " and " who shift the selection of a position. The button To record safeguard in the file $HOME/.prefbsf them validated modifications and lists it repertories of the current machine if the application does not manage these information. The fixed and volatile repertories can beings mixed in the list, but when they are integrated in the Répertoire menus… of the bsf the fixed repertories are gathered at the head.

By defect the remanent orders Unix and repertories are preserved in the file preferably bsf ($HOME/.prefbsf) for all the applications using lib_asterix. However lib_asterix allows with the application to store these information in its own file of preferences through some utilities (this solution is adopted by asterix).

For the paramètrage impression to see it [bib1].

Data-processing handbook of Description

7.3.1 Creation and call of a bsf in modal dialogue

Interfaces:

```
/*-----------------------------------------------------------------------
Creation of different the widgets dregs has one limps of selection
of file modal.
Initialization for the bsf but also for lib_asterix:
- initialization of the cursors
- initialization of the bills of character
- creation of the dialogue of impression
- creation of the dialogue of alarm
- creation of the dialogue of information
-----------------------------------------------------------------------
*/
Widget CreateBSF
(
Widget appSh/* - > ApplicationShell *
);
```

Create the structures of data for a *bsf* and returns allocated Widget.

```
/*-----------------------------------------------------------------------
Post the bsf wBSF in modal dialogue with initialization of
its parameters and returns the same parameters modify.
-----------------------------------------------------------------------
*/
bool ShowBSFmodal
(
Widget wBSF,/* - > BSF has to post into modal *
String tit,/* - > Title of Shell *
String mac,/* <-> Machine *
```

String usr,/* <-> To use *
String reference mark,/* <-> Repertory *
String fic,/* <-> File *
String nEdi,/* - > Name of the editor *
String nAct,/* - > Name of the action of the fsb *
String wire,/* <-> Filter on the files *
String aff,/* - > View of the ls *
String tsel/* - > Title of the Selection *
); /* < - TRUE if a file or reptoire east selects
*/
tit: titrate of Widget Shell of the bsf
mac: selected machine
usr: to use selected
reference mark: selected repertory (if vacuum = $HOME)
fic: selected file
nEdi: name of the associated editor (among the Unix orders defined in the file .prefbsf if order is not definite it is the first ordering of the list which is used)
nAct: name of the action of the bsf (button OK)
wire: filter Unix for the ls, the find or the grep (= regular expression)
aff: view of the ls (among Date - classified more recent with oldest or Name - by order alphabetical)
tsel: label of the field selected file

7.3.2 Creation and call of a bsf in nonmodal dialogue

Interfaces:
/*---
Creation of different the widgets dregs has one limps of selection
of file nonmodal (Autonomous)

Initialization for the bsf but also for lib_asterix:
- initialization of the cursors
- initialization of the bills of character
- creation of the dialogue of impression
- creation of the dialogue of alarm
- creation of the dialogue of information
----
*/
Widget CreateBSFA
(
Widget appSh/* - > ApplicationShell *
);
/*---
Post the bsf wBSF dialogues about it nonmodal with initialization
of its parameters.
----
*/
bool ShowBSFnm
(
Widget wBSF,/* - > BSF has to post into modal *
String tit,/* - > Title of Shell *
String mac,/* - > Machine *
String usr,/* - > To use *
String reference mark,/* - > Repertory *
String fic,/* - > File *
String nEdi,/* - > Name of the editor *
String nAct,/* - > Name of the action of the fsb *
String wire,/* - > Filter on the files *
String aff,/* - > View of the ls *
String tsel/* - > Title of the Selection *
);
It should be noted here that, the dialogue not being modal, the last selection carried out by the posted bsf cannot
to be turned over. This information is nevertheless preserved in the structures of the bsf and one can them to recover for réafficher the bsf with its last context thanks to the GetRemBSF function.
Interface:
/*---
Return the remanence of the BSF whose Widget Shell is wBSF.
----
*/
void GetRemBSF
(
Widget wBSF,/* - > Widget Shell of the BSF *

String usr,/* < - To use selects *
String mac,/* < - Machine selectionnee *
String reference mark,/* < - Repertory selects *
String fic,/* < - File selects *
String wire/* < - Filter *
);

7.3.3 Orders of launching of the editors (until 2.2)

To the version the 2.2 orders of launching of the editors are by defect:
tank CMDxedit [1024] = “xedit - fn Courier12”;
tank CMDvi [1024] = “xterm - Sb - E VI”;
tank CMDemacs [1024] = “emacs”;
tank CMDtextedit [1024] = “textedit”;
tank CMDsedit [1024] = “asedit”;
The intialisation is carried out in lib_asterix.h, but it is obviously possible to change them
orders constantly.

7.3.4 The orders Unix of the bsf and associated lists of PushButton

It is possible to define only 20 orders to the maximum.

7.3.4.1 Management of the PB lists

An application cannot contain which 24 lists of this type. For each bsf created one needs 2 lists: for the bsf
principal and for the bsf of copy.
Behavior of this list: it is updated after each call to the window of the preferences of the bsf. If
it is used in a context OptionMenu, the activateCallback of PushButton of the list updates
structures of OptionMenu consequently. If one gives Widget-id of PushButton, his label
is updated by the activateCallback of PushButton of the list. After an update, for each list,
if the order previously selected still exists in the list it remains the selected order,
if not it is the first ordering of the list which becomes the current order.

7.3.4.2 Creation of a list

```
/*-----------------------------------------------------------------------
Creation of a list of orders Unix related to the mechanism of the BSF
to be uses in the application.
Return an identifier to refer the structure creates.
-----------------------------------------------------------------------
*/
int InitPBprefBSF
(
Widget wpm,/* widget relative *
Widget wom,/* widget OM *
Widget wedit
);
```

7.3.4.3 Selection of an order Unix of OptionMenu starting from its name

```
/*-----------------------------------------------------------------------
ActivateCallback of the order ncmd of the structure id of orders Unix
bound to mechanism BSF.
-----------------------------------------------------------------------
*/
void activePBcmdUnix
(
int id,/* - > Identifying of the PB list *
Widget wom,/* - > OptionMenu associates *
String ncmd/* - > Name of the associated order Unix *
);
```

7.3.4.4 Recovery of the name of an order Unix selected with OptionMenu

```
/*-----------------------------------------------------------------------
```

Return the name of the Unix order selects in OptionMenu of one
list PB associated with the mechanism with the BSF.
----
*/
String historiqueOMcmdUnix
(
int id,/* - > identifying of the list of PB *
Widget wom/* - > OptionMenu concerns *
); /* < - Name of the order Unix selectionnee *

7.3.4.5 Call of the window of the preferences of the bsf

/*--
To post, in modal dialogue, the preferences of the bsf.
Return TRUE if the modifications were enregistrees in
$HOME/.prefbsf
and the modifications reportees in OptionsMenus and PullDownMenu
dregs has this mechanism.
----
*/
bool ShowPrefBSF (void);

7.3.5 Conservation of the orders Unix and the remanent repertories by the application.

Information concerned owes beings put in a file with key words of the form:
“key word: value”
If the application does not have this type of file of preferences it is possible to create some for the occasion.

7.3.5.1 To prevent lib_asterix

It should be indicated to lib_asterix that one deals with the safeguard not to take into account it file $HOME/.prefbsf with initialization and especially not to crush it at the time of the safeguard. This statement must be made after the initialization of lib_asterix: Init_lib_asterix ().

/*--
So that the application can declare that it manages the repertories
residual and the orders Unix in its file of preferences.
----
*/
void AsGestionPrefBSF (void);

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

7.3.5.2 Reading of information

/*--
Reading of the orders Unix for the bsf in a file of preferences.
Update of the corresponding structures.
Appelee when one meets the key word 'Definition of the Unix orders
:'.
The file is read until Fine key word the “definition of the Unix orders: ”.
----
*/
void AsLireCmdUnix
(
SPIN *fl/* - > identifying of the file of preferences *
);
/*--
Reading of the residual repertories for the bsf in a file
preferences.
Update of the corresponding structures.
Appelee when one meets the key word “residual Repertories: ”.
The file is read until Fine key word the “residual repertories: ”.
----
*/
void AsLireRepRem
(
SPIN *fl/* - > identifying of the file of preferences *
);

7.3.5.3 Safeguards information

/*--
Writing of the residual repertories of the bsf in a file
preferences.
----
*/

```
void AsEcrireCmdUnix
(
SPIN *fl/* - > identifying of the file of preferences *
);
/*----------------------------------------------------------------------
Writing of the residual repertories of the bsf in a file
preferences.
----------------------------------------------------------------------
*/
void AsEcrireRepRem
(
SPIN *fl/* - > identifying of the file of preferences *
);
```

7.4
To carry out an interruptible order system: sysint ()

The tools of asterix use all this function with each time an order system is potentially
blocking (ls in the presence of assembly NFS, CCP, rsh) or a long order (find or grep recursive).
To use it apart from the bsf and of the follow-up of the jobs it is necessary to define in the interface a
Interrompre button.

7.4.1 Button
To stop

This button appears only the execution time of sysint (). No pressing another button is
possible as long as this one is visible (events X are consumed). There can be several buttons
To stop in the application. At a moment given only one button Interrompre is active. In general when one
post a window having this type of button, one declares this button current (DefWint). When one closes
this
fenestrate it is necessary to make this button inactive so that another Interrompre button takes over

(UndefWint).
The event of interruption is detected starting from an event of the KeyRelease type in Widget
To stop.

7.4.1.1 To define the button to stop running DefWint ()

Can be done during the creation of the button for the principal window of the application or at the time of the popupCallback
for a window which is not always posted.
Interface:

```
/*------------------------------------------------------------------------
To define the button to stop running.
------------------------------------------------------------------------
*/
void DefWint
(
Widget wid/* - > Widget of the new button To stop running *
);
```

7.4.1.2 To make inactive a button to stop UndefWint ()

To make when one masks a window containing a Interrompre button returned running by DefWint (at the time of
popdownCallback).
Interface:

```
/*------------------------------------------------------------------------
To cancel the button to stop running and revnir with the precedent.
------------------------------------------------------------------------
*/
void UndefWint
(
Widget wid/* - > Interrompre button has to make inactive (to depilate) *
);
```

7.4.1.3 Example of declaration button To stop in the principal window application

In the interface, for the creation of the button, it is necessary to call the following functions:

DefWint (Widget_du_bouton); /* to announce that the button to stop running
*/
/* is this one *

Example of definition of a button To stop with XFM:

xfmCreateCallback =
function None DefWint (Widget);
DefWint (coil);
width = 102
height = 35
highlightThickness = 0
labelString = To stop

7.4.2 function sysint ()

Carry out an order system as Bourne Shell with possibility of stopping it. If the parameter - chatterer has summer transmitted to the application, the order carried out is sent in a window of text with elevator. If no Interrompre button was definite it is the order system () which is used. If not it order is carried out after a vfork. As long as the order is not finished and that one did not click in the button To stop, the process father awaits the termination of the process wire. All actions X on the buttons others that Interrompre is consumed (not to have to manage the coherence of the actions which would be “bufferized”).

To version 2.3 the interruption was detected using the aggregate variable to stop updated in the activateCallback of the button. This one is not used any more, but if it exists, it does not harm the good
operation of the mechanism.

This function takes account of two arguments transmitted to the achievable one:

· - talkative which makes it possible to return the echo in a window of the Shell orders carried out via function sysint (),

· - noexec which makes it possible not to carry out the Shell orders carried out via the function sysint ().

Interface:

```
/*------------------------------------------------------------------------
Launch a Shell order with possibility of stopping it.
For that need has to know Widget stop key
urgently.
------------------------------------------------------------------------
*/
int sysint
(
```

String cmdstring/* - > order has soummettre as Bourne Shell *
);
Return 0 if the tender of the order were well placed and 1 if not. (This code of return is not the code of return of the order it even as for the function system ().)

7.4.3 Detection of the errors: errshellRSH ()

To detect the errors of an order system subjected since a program C it is necessary to analyze them messages sent in stderr. When, in an order system, one redirects stderr in a file,
the function errshellRSH () makes it possible to analyze this file to determine if the order were badly placed. If
the file in which one redirected stderr is not empty the function returns TRUE and the first line of the file.
This function detects in more the message on a password out-of-date on the CRAY since Unicos 8.0 (it does not have there
not yet of equivalent on claster). If a password is out-of-date, the function proposes to change it (via one rlogin) if one does not use lib_asterix within the framework of ASURE. If one uses this mechanism with ASURE, it
is not possible to directly carry out a rlogin, it is necessary to carry out a connection with a authentifior.
In this case the user is warned by a message in a window popup.

Interface:

```
/*---------------------------------------------------------------------
Return TRUE if the file tmperr is nonempty and false if not.
If this file is nonempty its first line is recopiee in
the variable err.
to test if the password is out-of-date
and to send a connection to change the password if one utilse not
ASURE.
---------------------------------------------------------------------
```

*/
bool errshellRSH
(
String err,/* < - contents of the first line of the file tmperr *
String mac,/* - > machine destination of the rsh *
String usr,/* - > to use destionation of the rsh *
String ficerr/* - > name of the file in which stdout was redirects *
);
Return TRUE if ficerr is nonempty and FALSE if not.

7.5 Follow-up

jobs on the object computer
This solution to see evolving/moving the execution of the jobs on the object computer requires a repertory on
station which will contain the list of the jobs managed by the follow-up and the files of result which are associated
(FLASH_STATION).
To version 3.1 this mechanism was integrated in the application. A job subjected by another application was not taken into account.
Since version 3.2 it is an autonomous application (asjob) which can be used by several applications
at the same time. To preserve the interest of this functionality, it is desirable that all the applications
use the same repertory flashor on station ($HOME/flash_Aster).
For a use with tender of passages of *Code_Aster* only initialization and posting of the window
of follow-up of jobs are to be used. Functions of construction of scripts and tenders for *Code_Aster*
integrate the use of the mechanism of follow-up. For the other uses it is necessary, moreover, to announce each time
that a new job is to be dealt with and to return the information awaited by the follow-up of the jobs in
files in FLASH_STATION. The communication between the applications which subject jobs and the application
of follow-up is ensured by the mechanisms of waiter X (positioning of atoms and emission of events X).

7.5.1 Initialization, creation of the window of follow-up

With the following function:

```
/*-----------------------------------------------------------------------
Creation of the window of follow-up of the jobs. and Initializations
inherent this functionality has, with access to the parameters of
connection.
Creation of the repertory flashor on station.
-----------------------------------------------------------------------
*/
bool AsCreateSuivJobTempo
(
String rep_flash_station,/* - > Repertory flashor on station
with way in absolute or by report/ratio
$HOME has.
*/
bool makes an attempt,/* - > Flag of waiting or not of the end of
connection.
*/
int nbMaxEssaiConnex,/* - > max. Nb of tests of connection.
*/
int nbMaxEssaiPhase,/* - > max. Nb of tests by phase.
*/
int delaiEssai/* - > withdrawal period (in S) between each
phase.
*/
); /* < - TRUE if connection is established.
*/
```

This interface giving access to the parameters of the attempt at connection is usable starting from the version
3.6. To preserve compatibility, or if one does not want to modify the default settings, the old one interface is preserved. The default values of the new parameters are:

· nbMaxEssaiConnex=10,
· nbMaxEssaiPhase=5,
· delaiEssai=1.

```
/*-----------------------------------------------------------------------
Creation of the window of follow-up of the jobs and initializations
inherent this functionality has.
-----------------------------------------------------------------------
*/
void AsCreateSuivJob
```

(
String rep_flash_station,/* - > Repertory flashor on station
*/
bool makes an attempt/* - > Attente or not of the end of
the initialization of asjob
*/
);
The application tries to communicate with an application of the asjob type having the same repertory flashor on
station and using same DISPLAY X. If there is no application corresponding to these criteria, an application
asjob is launched ("asjob - rep_flash rep_flash_station"). If the application is used with
possibilities CCP/rsh of ASURE the adequate parameters are transmitted to asjob.
Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

If the repertory does not exist there is created with mkdir (thus on only one level). The asterix application uses it
repertory $HOME/flash_Aster. To preserve the universal aspect of this follow-up of jobs on the waiter Aster, it
is advised to use this repertory for all the applications calling upon the services of asjob.
Not to lengthen the initialization of the application, it is possible not to await the end of the launching of asjob
and the complete initialization of the communication enters the two applications by positioning the parameter
make an attempt on FORGERY. In this case the communication is really established only when a job is subjected or that
the window of follow-up is asked for. To avoid having several asjob applications at the same time, a mechanism
"debouncing" is set up. It is not possible to launch, in the same application, twice asjob with less

from 6 seconds of interval.

7.5.2 To post the window of follow-up

If no asjob application with the good characteristics is detected, it is launched. If not fenestrates it follow-up is posted (die-inconifiée).

With the following function:

```
/*------------------------------------------------------------------------
Posting of the window of follow-up of the jobs.
------------------------------------------------------------------------
*/
void AsShowSuivJob (void);
```

7.5.3 To add a job to be managed by the follow-up

If no asjob application with the good characteristics is detected it is launched.

This functionality is to be used only when one is useful oneself of the follow-up for jobs not Aster or not subjected with
the AsSubaFicUnic function ().

From version 3.2 of lib_asterix the parameter origin is added (for the follow-up of the uses of *Code_Aster*). To avoid any confusion the name of this function changed (AddJob () - > AsAddJob ()).

Interface:

```
/*------------------------------------------------------------------------
To insert a job in end of the list chainee of the jobs.
To increment Nb of job not finished.
------------------------------------------------------------------------
*/
String AsAddJob
(
int num,/* - > num. NQS of the job *
String name,/* - > name of the job *
String usr,/* - > to use of execution *
String mac,/* - > object computer *
String fic,/* - > file name for the tail (followed interactive of
job) *
int tpi,/* - > initial time requires *
String dates,/* - > date tender jj/mm/aa *
String hour,/* - > hour tender hh: mm: ss *
String origin/* - > application of origin *
); /* < - error message to the tender *
```

This function can return an error message if it detects an anomaly. In the contrary case it return a null string ("\ 0").

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

Code_Aster ®

7.5.4 The files which the job must send on the flashor station

Only when one uses the follow-up for jobs not Aster. The follow-up of the jobs detects the end of a job is in
questioning the manager of batch on the object computer, that is to say by detecting the existence of the file
FLASH_STATION/nomjob.dn°id. When the object computer answers that the job does not turn any more, the follow-up
jobs will seek, to update its list, the following files:

· FLASH_STATION/nomjob.dn°id which must contain the diagnosis of the job,
· FLASH_STATION/nomjob.mn°id which must contain the messages of the job.

To follow in an interactive way unfolding of the job, script must return, as soon as this information is to him
known, the repertory of work in which the file of message in the file arrives:

· FLASH_STATION/nomjob.tn°id which must contain the name of the temporary repertory.

7.5.5 Safeguard information of the Follow-up of the Jobs

Information of the Follow-up of the Jobs is stored in main memory in the shape of a chained list.
contained list is safeguarded in the FLASH_STATION/.flashjob file when the application is left
of follow-up of the jobs and with each addition or suppression of job in the list.

7.5.6 To close the asjob application starting from an other application.

Normally when one leaves the application which communicates with asjob, asjob remains active. One can nevertheless
to send a message to asjob so that the application finishes and thus to couple the end of its application with
end of asjob.

Interface:

void AsQuitSuivJob (void);

7.6

Tender of a study Aster

The jobs created by lib_asterix use the follow-up of the jobs. That implies to have initialized the follow-up of the front jobs
the first tender: AsCreateSuivJob () [§7.5].

All the functions of tender of an Aster execution of lib_exec_aster [§6.7] are appealable in context lib_asterix. They have just a behavior different (coupling with asjob, Interrompre button, ...).

7.7

Fenestrate confirmation

The window of Motif confirmation is not ideal to use for two reasons:

· it is not possible to change the color of the buttons (they are Gadget),

· the only means of selecting a button as being the action by defect is to do it with the creation of the window (it is thus necessary to create Widget with each use).

To cure that, the library lib_asterix proposes appealable Widget in two different ways according to whether it is wanted that the positive or negative action is the action by defect.

Interface with the Oui button by defect:

```
/*-----------------------------------------------------------------------
To post, in modal dialogue, one limps of alarm with the Oui button
select by defect.
-----------------------------------------------------------------------
*/
bool ShowAlerteOui
(
String titrates,/* - > title of Shell *
String msg/* - > message has to post *
); /* < - TRUE if confirmation with the button Yes *
```

Interface with the Non button by defect:

```
/*-----------------------------------------------------------------------
To post, in modal dialogue, one limps of alarm with the Non button
select by defect.
-----------------------------------------------------------------------
*/
```

bool ShowAlerteNon
(
String titrates,/* - > title of Shell *
String msg/* - > message has to post *
); /* < - TRUE if confirmation with the button Yes *
These functions return TRUE if the selected action is Oui and FALSE if the selected action is Non.

7.8 Edition

Provision them utility "to publish" files on any machine. Orders used
are defined in the file $HOME/.prefbsf or the file preferably of the application. In general one
do not recover an error message when that occurs badly (when the order is not found by
example) because the order is launched in "background".
If the order is not defined it is the first ordering of the list which is used.

7.8.1 To publish a file

Interface:
/*--
Send the format control CMD… for an editor gives
on any machine.
In gal not of return of error bus launching in background.
----
*/
String To publish
(
String EDI,/* - > Name editor among: sedit xedit VI emacs textedit *
String mac,/* - > Machine of the file has to publish *
String usr,/* - > User of the file has to publish *
String reference mark,/* - > Repertoire of the file has to publish
If vacuum it is $HOME *

String fic/* - > Nom of the file has to publish *
); /* < - Error message *

7.8.2 To publish all the files “éditables” profiles

To publish all the files having the “flag” E (éditable) in a profile of study and a profile of execution (and of
more general manner in one or two profiles). A session editor is launched on the first three machines
found in (S) the profile (S) (to avoid the multiplication of the windows). This function has direction only with one
multifile editor (sedit, emacs, VI). If one uses an editor not existing on all the machines,
format control on this machine is not taken into account (it does not have there an error message because
order is in “background”).

Interface:

```
/*-----------------------------------------------------------------------
Launch a session of editor for all the files has to publish
of a profile of study and a profile of execution. A session
by machine with with more the 3 sessions.
No the return of error bus launching in background.
-----------------------------------------------------------------------
*/
void ToutEditer
(
PTDescFic *RacE,/* - > Root of the list Study *
PTDescFic *RacX,/* - > Root of the Exec list *
String EDI/* - > Nom Editor uses among:
sedit xedit VI emacs textedit *
);
```


7.9

Display window of text

Provision of a window posting, in modal dialogue, a text in a window with elevator and with the possibility of printing the text starting from the box of impression of the bsf.

Creation of the window:

```
/*-----------------------------------------------------------------------
Creation of a window to visualize text
-----------------------------------------------------------------------
*/
void CreateVisuInfo
(
Widget appShell/* - > Shell Application *
);
```

Posting of the window:

```
/*-----------------------------------------------------------------------
To post, in modal dialogue, a text in a srcolled window with
possibility of impression.
-----------------------------------------------------------------------
*/
void ShowVisuTexte
(
String titrates,/* - > Titre of the window *
String Linf,/* - > text of the label to the top of the text *
String txt/* - > text has to post *
);
```

7.10 Some utilities in bulk

7.10.1 Version of lib_asterix

To know the version of the bookshop used.

Interface:

```
String Version_lib_asterix (void);
```

7.10.2 Name of item the current of an option-finely: historiqueOM

Return the name of Widget of item the current of an option-finely.

Interface:

```
/*-----------------------------------------------------------------------
Return the name of the Widget last clicks in an option-finely
-----------------------------------------------------------------------
*/
String historiqueOM
(
Widget wom/* - > Widget of the option-finely *
); /* < - Name of Widget selects in the option-finely *
```

Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

7.10.3 To choose item option-finely: activeClick

To select one in an option-finely item and to activate the corresponding action.

Interface:

```
/*------------------------------------------------------------------------
To update an option-finely has item gives.
------------------------------------------------------------------------
*/
void activeClick
(
Widget wom,/* - > Widget of the option-finely *
Widget witem/* - > Widget of item has to activate *
);
```

7.10.4 To print a character string: ImprString

Writing the character string in a temporary file and posts the box of impression of file of the bsf.

Interface:

```
/*------------------------------------------------------------------------
To print character strings by using the window of choice
destination (printing) and format of the BSF.
------------------------------------------------------------------------
*/
void ImprString
(
String str/* - > chains has to print *
);
```

7.10.5 MAJ of a XmNlabelString resource

With creation and release of the XmString chain essential to the operation.

```
/*------------------------------------------------------------------------
To properly affect a labelString starting from String
(with release of the memory of XmString).
------------------------------------------------------------------------
*/
```

void SetNlabelString
(
Widget W,/* - > Widget having a resource labelString *
String str/* - > Chaine has to put in the labelString *
);

7.10.6 Recovery of a XmNlabelString resource

/*---
To properly recover a labelString of Widget
(with release of the memory of XmString).
----
*/
void GetNlabelString
(
Widget W,/* - > Widget having a resource labelString *
String str/* <-> labelString recovers *
);

7.10.7 Posting of the cursor shows in a window

/*---
Post the Montre cursor (immediately)
----
*/
void tempoM
(
Widget wid/* - > Widget for which it is necessary to apply the cursor Montre *
);

7.10.8 Return to a normal cursor

void UnDefCur

```
(
Widget wid/* - > Identifying of the window *
)
/*******************************************************************/
/* Positionne the cursor by defect of the window *
/*******************************************************************/
```

8 Example
of use
8.1 Example
for
lib_exec_aster

Example of use of lib_exec_aster, with an application which subjects a profile of study starting from a name
of file given on the line of order (- ready).

```
/*################################################################################
#
Example of use of lib_exec_aster to subject an execution
on a waiter Aster starting from a file profile of study asterix.
################################################################################
#*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <lib_exec_aster.h>
static void left (int status);
static void use (tank *nc);
hand (int argc, tank ** argv)
{
int status=0;
tank *nfp=0;
tank *nc=0;
tank *nomod=0;
char* err;
int numID;
printf ("Version of the library lib_exec_aster: %s \ N \ N",
Version_lib_exec_aster ());
```

```
/*-----------------------------------------------------------------------
--
Initialization lib_exec_aster
-----------------------------------------------------------------------
-*/
Init_lib_exec_aster (argc, argv, “/tmp/asexec”);
/*-----------------------------------------------------------------------
--
Specific treatment of arguments A the application
-----------------------------------------------------------------------
-*/
nc=AsGetNomAppli ();
while (*argv) {
yew (strcmp (*argv, “- ready”) == 0) {
argv++;
yew (*argv) nfp=strdup (*argv);
else {
fprintf (stderr, “err: %s: It misses the name of the file profile
of study \ N ",
nc);
exit (1);
}
}
else yew ((strcmp (*argv, “- help”) == 0) || (strcmp (*argv,” - H ") == 0) ||
(strcmp (*argv, “- assistance”) == 0)) {
use (nc);
exit (0);
}
argv++;
}
yew (! nfp) {
fprintf (stderr, “err: %s: It misses the name of the file profile
of study \ N ", nc);
use (nc);
```

```
exit (1);
}
/*-----------------------------------------------------------------------
--
Tender execution
-----------------------------------------------------------------------
-*/
err=AsExecFicProfil (nfp, “EXEC_ASTER”, (String) 0, (String) 0, &numID);
yew (numID == 0) {
yew (err) {
fprintf (stdout, “%s: err %s \ N”, nc, err);
free (err);
}
else {
fprintf (stdout, “%s: err??? \ N”, nc);
}
exit (1);
}
else {
fprintf (stdout, “%s: %s \ N”, nc, err); free (err);
fprintf (stdout, “%s: Num. Job: %d \ N”, nc, numID);
fprintf (stdout, “%s: Local Flashor: %s \ N”, nc, AsGetFlashStation ());
```

```
fprintf (stdout, “%s: Flashor exec: %s \ N”, nc, AsGetFlashExec ());
}
```

```
/*----------------------------------------------------------------------------
--
Exit
----------------------------------------------------------------------------
-*/
exit (status);
}
/*----------------------------------------------------------------------------
--
Exit of the application, release of the resources
----------------------------------------------------------------------------
-*/
static void left (int status)
{
Quitter_lib_exec_aster (TRUE);
exit (status);
}
/*----------------------------------------------------------------------------
--
Posting of the parameters recognized by the application
----------------------------------------------------------------------------
-*/
static void use (tank *nc)
{
fprintf (stderr, "\ nusage: %s - ready fichier_profil_d_etude [parameters
lib_exec_aster] \ N ", nc);
}
```

8.2 Example
for
lib_asterix

Principal sample program of an application of which the interface part is carried out with XFM. Use of two bsf (modal and nonmodal), display window of a text and follow-up of the jobs.

```
/*###################################################################################
#
Example of use of lib_asterix
####################################################################################
#*/
#include <stdio.h>
#include <Fm.h>
#include <lib_asterix.h>
Widget FmCreateexemple ();
Widget wBSFmodale; /* Widget of the modal BSF *
Widget wBSFnm; /* Widget of the nonmodal BSF *
```



```
Widget GetwBSF ()
/*****************************************************************/
/* to communicate Widget of the modal BSF created in *
/* the hand of the interface of the example *
/*****************************************************************/
{
return (wBSFmodale);
}
Widget GetwBSFnm ()
/*****************************************************************/
/* to communicate Widget of the nonmodal BSF created in *
/* the hand of the interface of the example *
/*****************************************************************/
{
return (wBSFnm);
```

```
}
void exit (int status)
{
/* Envoyer a message has asjob to finish the application *
AsQuitSuivJob ();
Quitter_lib_asterix (TRUE);
exit (status);
}
hand (argc, argv)
int argc;
tank ** argv;
{
Widget toplevel;
Widget appShell;
printf ("Version of the library lib_asterix
: %s \ N ", Version_lib_asterix ());
/* Initilisation without creation of widget *
toplevel = FmInitialize ("zz", "zz", NULL, 0, &argc, argv);
/*-----------------------------------------------------------------------
--
Initialization lib_asterix
------------------------------------------------------------------------
-*/
Init_lib_asterix (argc, argv, "/tmp/Exemple_lib_asterix", toplevel);
/*-----------------------------------------------------------------------
--
Specific treatment of arguments A the application
------------------------------------------------------------------------
-*/
while (*argv) {
yew ((strcmp (*argv, "- help") == 0) || (strcmp (*argv," - H ") == 0) ||
(strcmp (*argv, "- assistance") == 0)) {
exit (0);
}
argv++;
```

}

```
/* Creation of the Shell application of the IA *
appShell=FmCreateexemple ("ex_lib_asterix", toplevel, NULL, 0);
/* Creation of the follow-up of job *
AsCreateSuivJob ("flash_Aster", FORGERY);
/* Creation of the BSF *
wBSFmodale=CreateBSF (appShell);
wBSFnm=CreateBSFA (appShell);
/* Creation of a display window of text *
CreateVisuInfo (appShell);
/* Main Vent Loop *
FmLoop ();
}
```

9
Differences in use with the former versions
9.1
With version 1.1
To be able to change the bookshop lib_asterix without needing recompiler the application (what is one of required goals) two points requiring of the modifications compared to the use of version 1.1 of

lib_asterix was changed:
· aggregate variable VERSION_LIB_ASTERIX does not exist any more, it is replaced by the function Version_lib_asterix (),
· it is not necessary any more to precede inclusion by the file lib_asterix.h by #define INIT_DEF
in the file containing the main program of the application.

9.2

With version 2.2

Extension of the concept of edition to the concept of execution of an order Unix with like parameter the file
selected. These Unix orders can be redefined by the user and are preserved in the file
$HOME/.prefbsf. This fact aggregate variables CMD…. are not used any more for the call to the editors.

9.3

With version 3.0

Setting in conformity with system UNICOS 8.0 on CRAY.

9.4

With version 3.1

Version 3.2 of lib_asterix is incompatible with the preceding one. The installation of the new version of dynamic library is not enough, recompiler is needed the application and to modify it by making some choices.
The interfaces of the functions provided by lib_asterix are with format ANSI. It is necessary to compile with the option -
D_NO_PROTO_LIB_ASTERIX not to have the declaration of the parameters.
Modification of the call to construction and the tender of the jobs (addition of parameters). To avoid the errors,
the functions of which the number of arguments changed changed name.

List changes:
· ConsJob ()

- > AsConsJob ()
· SubaFicUnic ()
- > AsSubaFicunic ()
· AddJob ()
- > AsAddJob ()
· CreateSuivJob ()
- > AsCreateSuivJob ()
· ShowSuivJob ()
- > AsShowSuivJob ()
Possibility of interactive follow-up of the jobs.
The follow-up of the jobs is an autonomous application being able to communicate with several applications. It is
advised to change repertory FLASH_STATION to use that of asterix: $HOME/flash_Aster.
Possibility of preserving the repertories of the bsf in a file of preferences. These repertories, like orders Unix of the bsf, can be taken into account by the application or lib_asterix.
More aggregate variable to stop in the mechanism sysint ().
Taking into account of protected orders CCP/rsh through ASURE.
The choices to be made are:
· the choice of a signature of the application for the emission of the jobs,
· the rallying or not with a repertory using flashor on station commun runs with the applications lib_asterix,
· the management of the orders Unix and the remanent repertories of the bsf is made by the application or by
lib_asterix (in the file $HOME/.prefbsf),
· the adaptation or not of orders CCP/rsh to ASURE.

9.5

With version 3.2

The interfaces of two functions are modified. Parameters IN are transformed into IN-OUT.
That should not pose problem with the use because there is the same number of parameters with the same one
type and it are of String.
Modified functions:
· Local ()
· FichierLocalRef ().

9.6

Since version 4.0

Before this version there was only one library (lib_asterix) to ensure the official tender executions on a centralized machine Aster and to distribute the graphic utilities developed for asterix. It thus had obligatorily to be used in a context X-Reason. Starting from version 4.0, lib_asterix was cut out to be able obligatorily to subject executions without being in a graphic context with creation of the library lib_exec_aster.

Data-processing handbook of Description
D1.03 booklet:

This cutting requires a linkage editing for all the applications using the preceding versions of lib_asterix.

From this version it is henceforth possible:

· To subject Aster executions without obligatorily having an environment X-Reason (one does not have any more
need for the application of follow-up of the jobs - asjob-).

· To change the repertories of destination of the files on the local machine and that of execution thus that the repertory of work starting from the arguments on the line of order (these parameters are thus modifiable by the user or the system administrator and are not reserved any more for the developer of the application).

Other additions:

· use
with
- help, - H or - assistance on the line of order

· possibility of stopping asjob since its application,

· tender of a study starting from a file name profile asterix (AsExecFicProfil ()).

To my knowledge that relates to the following applications:

· anthemix (if that still exists),

· mekelec,

· circus,

· cab (coupling BOSS Quatro-Aster).

10 Bibliography

[1]
A.M. DONORE, C.MASSERET - Access to *Code_Aster* - general Conditions of access to *Code_Aster* [U2.00.00].

[2]
Interface graphic access to *Code_Aster*. Necessary configurations hardware and software. Document [U2.02.01].

[3]

To install the graphic interface of access to *Code_Aster* on its workstation. Document [U2.02.02].
[4]
Y.DHERBECOURT - Extension of ASURE to the services rsh/CCP: Functional specification HI-57/95/014.
[5]
Installation and use of CCP/rsh with lib_asterix [U2.02.03].
Data-processing handbook of Description
D1.03 booklet:
HI-75/01/005/A

Data-processing handbook of Description
D1.05 booklet: -
D1.05.01 document

To measure performances (CPU) on AlphaServer
or on Linux

Summary:

There are tools making it possible to trace the times CPU used (profiling) in Code_Aster.

On AlphaServer, these tools do not require a recompiling of the Aster sources. One uses for that the tool
atom. The disadvantage of this tool (specific alphaserver) is that the instrumentation of achievable involves
overcosts of execution which can be very important (up to 10 times the original cost). Under these conditions, it is
difficult to be sure relevance of measurement.

On Linux, one uses the traditional method: one recompile all sources with the option "- pg" and one use
the tool gprof. The overcost of the instrumentation is negligible.

---

1 On
Alphaserver

1.1
To prepare the profiling

One works from the achievable one that one intends to use to launch his Aster study:

· Exécutable native: To recopy the achievable aster on a local repertory which belongs to you on
waiter (the achievable native is in /aster/v7/NEW7/ on the waiter and names asterd
or asteru in debug mode or not).
· Private Exécutable: To prepare your overload like usually with ASTK or run_aster and
build your achievable.

It is then necessary to modify the achievable one using the tool atom.

On your repertory containing achievable Aster which you want to profile:
atom - tool hiprof votre_executable

The program will create new achievable named <votre_executable.hiprof>

1.2
To make the profiling

For ASTK or run_aster it is necessary to use the new achievable one in overload. It is necessary imperatively
to modify the script of launching of Aster bus during the execution in profiling, the file <votre_executable.hiout> will be created in the temporary repertory of calculation. It is thus needed to copy in the adequate repertory.
For ASTK:

1) Prepare the study (file, overloads of the catalogues, new achievable "profiled", bases, time, memory and options various).
2) Add script btc in RESULT in the mitre OVERLOADS.
3) Launch calculation. Calculation will not be carried out (one limps of dialogue informs you) but script (btc) will be created.
4) Modify script btc by publishing it and by adding the following line at the end:
CP votre_executable.hiout /chez_vous/votre_executable.hiout

Take guard! For any modification in the profile of execution (in particular time and memory), it is essential to recreate the btc and to modify it.
After execution, one finds oneself with two files:
votre_executable.hiout
votre_executable.hiprof

These two files must be in the same repertory. One carries out gprof then while redirecting standard exit:
gprof votre_executable votre_executable.hiout > ResultatProfil

You have from now on a file <ResultatProfil> which is the result of the analysis.

For the possible options, to make a man gprof. Some useful options:

gprof - has
Avoid the posting of the static functions, in particular the calls systems which weigh down it file

*gprof - has - F jeveuo_*

Limit posting with the function designee

Caution:

For a routine FORTRAN, imperatively add a _ (underscore) at the end of the name of routine and remove the extension .f

---

1.3
To strip the results of the profiling

By defect, the file is heavy. It is possible to limit the posting of the infos while playing with the options of gprof. "Times systems" are indicated in the form of a number of instructions used.

One will detail a little, while starting with the end of the file:
****
Index by function name

[401] PyArg_Parse [591] cftabl_ [1000] proc_at_0x1213acb50
[212] PyArg_ParseTuple [84] cftyli_ [660] proc_at_0x1213ad470
[1137] PyArg_ParseTupleAnd [310] cgmacy_ [453] proc_at_0x1213ad560
[1605] PyBuffer_FromObject [79] charme_ [680] proc_at_0x1213aeac0
[1256] PyCFunction_Fini [476] chlici_ [1221] proc_at_0x1213aedc0
[531] PyCFunction_New [190] chloet_ [217] proc_at_0x1213b18e0
[1549] PyCObject_AsVoidPtr [226] chmano_ [629] proc_at_0x1213b1e00Y

Each function called during the execution is located by a number between hook.

Just with the top:
****
granularity: instructions; units: inst' S; total: 201924201580.70 inst' S

<A> <B> <C> <D> <E> <F> <G>
49.6 100384307222 100384307222 161 623505013 623596299 tldlr8_ [16]
31.0 163144941823 62760634601 506 124032874 124101882 rldlr8_ [17]

This table summarizes the most frequent calls.
COLUMN <A>: percentage of the number of instructions carried out by this function compared to total of the execution.
COLUMN <B>: a number of instructions cumulated by this function and those which precede.
COLUMN <C>: a number of instructions for this function.
COLUMN <D>: a many calls have this function
COLUMN <E>: relationship between the column <B> and the column <D> (an average number of instructions by
call of the function)
COLUMN <F>: numbers means of instruction by call of the function and of its
descendants.
COLUMN <G>: name of the function and its reference number (between hooks).

In this example, the function tldlr8 took 49.4% of the total of calculation while being called 161 times.
****

Lastly, at the beginning of the file, we have the tree of complete call. It will be sorted by order of call (one
start with the hand and one goes down) or by a function (see the options of gprof).
****
Let us take the example of tldlr8:

<A> <B> <C> <D> <E> <F>

100263313681.76 14679301.29 161/161 tldlgg_ [15]
[16] 49.7 100263313681.76 14679301.29 161 tldlr8_ [16]
3129121.03 6207534.02 4485/30537 __upcUpcall [352]
35974.59 2749927.50 522/195235 jelibe_ [65]
192341.36 1770419.18 1005/775659 jeveuo_ [56]
47302.73 140745.02 161/202579 jedema_ [102]
18938.92 126525.05 322/63148 jeexin_ [196]
27722.26 85430.33 94/49118 jeecra_ [154]
17033.41 67779.29 94/13206 jecreo_ [257]
45068.75 84.88 1044/1075446 jexnum_ [163]
13618.68 2023.63 161/202581 jemarq_ [205]
1710.66 0.00 161/3481 infniv_ [853]

One locates the instruction of the tree of call by the number between hooks on the left. Here, the number [16]
indicate the function tldlr8_ (as indicated in the end of the file for example).
It is the function-reference (the node of the tree).
The lines with the top are the appealing ones of this function (they are the function-parents), those in below are the functions called (they are the function-children).
Each function has two principal digits: the number of instructions carried out in itself (“final” instruction of FORTRAN) and numbers it instructions carried out in the functions children.

Function-parent
Function-parent
...
Function-reference
Function-child
Function-child
Function-child
Function-child
...

For the function-reference:

COLUMN <A>: number of location of the function-reference.
COLUMN <B>: figure 49.7 is the percentage of the number of instructions carried out by this function-reference compared to the total of the execution (idem preceding table)

COLUMN <C>: a number of instructions for the function-reference itself.
COLUMN <D>: a number of instructions for the function-children of the function-reference.
COLUMN <E>: a number of times or the function was called
COLUMN <F>: name of the function-reference

For the function-parents and the function-children:
COLUMN <A>: vacuum
COLUMN <B>: vacuum
COLUMN <C>: a number of instructions for the function itself.
COLUMN <D>: a number of instructions for the descendants of the function
COLUMN <E>: give two digits a/b whose direction varies according to the type of function (relative or
child compared to the function reference):
· For the function-parents (above the function reference) a/b:
<a> is the number of times where the function-reference was called by this function-parent by report/ratio with the total number <b> of calls of the function-reference.
· For the function-children (below the function reference) a/b:
<a> is the number of times where the function-child was called by the function-reference by report/ratio with the total number <b> of calls of the function-child.
COLUMN <F>: name of the function

Note:

· If the number of instructions for the descendants of a function is worth zero, it is that the function considered no other calls any. One is "with the end" of the tree, it has only calls there
Basic FORTRAN in the function. (it is the case of infniv for example)
· For a given function-reference, if one makes the sum of the <a> in the column <E> of functions parents, one obtains the total number of calls of the function reference.
· For a given function-reference, if one makes the sum of the columns <C> and <D> of its function-children, one obtains the figure of the column <D> of the function-reference.

Analyze example

In the example presented, the function tldlr8 is expensive since with it-only, it represents close to half of the total number of instructions of the execution. It is also seen that they are its clean instructions which take time and not the call to his/her function-children (the relationship between the two
reached 1000). As only the function tldlgg calls tldlr8, it is necessary to look at the tree of call for this function. It is seen whereas it is the algorithm of contact/friction (fropgd) which is more glouton (the 2/3 of the calls to tldlgg are made by the algorithm of contact).
Data-processing handbook of Description
D1.05 booklet: -
HT-66/05/003/A

---

2 On Linux

2.1 Instrumentation with f77 - pg (or DC - pg)

On the Linux machine/Rock'n'rolls clpaster (cluster of PC of department AMA), the problem of integral recompiling of the sources is less crucial than on the alphaserver: recompiler can entirely Aster in less than 30 minutes "elapse".

To carry out this recompiling with Astk, it is necessary:
· "to overload" all the sources (F77 and C). To save time, one can concaténer them F77 sources in "packages" (300 routines for example).
· to modify the file "config.txt" to add the option" - pg " on the 5 following lines:
OPTL | f90 |? | - v - pg
OPTC_D | DC |? | - C - G - pg - DP_LINUX
OPTC_O | DC |? | - C - pg - DP_LINUX
OPTF_D | f90 |? | - C - G - pg - I/opt/mpich2- 1.0.1/include
OPTF_O | f90 |? | - C - O2 - pg - I/opt/mpich2- 1.0.1/include

The config.txt file thus modified makes it possible to instrument the code in mode "debug" and "nodebug".
The mode "nodebug" is a priori preferable to measure "the true" performances of the code. In revenge, the mode "debug" is necessary if one wants to know the most consuming lines.

I unfortunately observed an unexplainable problem in mode "debug": the result of the profiling indicated bonds of incoming call routines which did not exist! One can however hope that this anomaly entirely does not invalidate the remainder of measurement.

As example, I profiled the test ssnv506c and I obtained the following total results:

- *in nodebug mode without instrumentation: 138s*
- *in nodebug mode with instrumentation: 139s*
- *in debug mode without instrumentation : 218s*
- *in debug mode with instrumentation : 228s*

It is noted that the instrumentation has a negligible cost CPU.

2.2
Execution of the Code instrumented with Astk

Once this made instrumentation, it should be carried out the study that one wants "to profile" with the achievable one that one has just produced. The problem is that the execution of the study produces a file (called gmon.out) in the temporary repertory of execution. This file is thus lost at the end of the execution if one does not take precautions.

To preserve the invaluable file gmon.out, it is necessary to use Astk in interactive and to click the button "to launch pre" (instead of traditional "the throw run"). This option of Astk makes it possible to prepare environment of execution. One places oneself then in the prepared repertory and one "launches" Aster manually. It is about the same "easy way" as for the use of a debugguor.

2.3
Analysis of the results

Once the study carried out and the file "gmon.out" recovered, one can analyze this file with order:

gprof mon_executable gmon.out > listing

The interpretation of the file obtained (listing) is the same one as that described with [§1.3]. Excellent document describing all the process of profiling is that written by Jay Fenlason and Richard Stallman:
"Gnu gprof The GNU to profile". One easily finds it on the Web.

Note:

Even if one recompile all sources of Aster, "depth" of the analysis of the performances stop with the libraries which one uses with the edition of the bonds and which were not compiled with "pg". It is for example the case of routines BLAS. The time spent in these libraries cannot be attached to the routines of Aster which call them. This defect can be important, by example, if one wants to measure the performances of solveurs MUMPS or MULT_FRONT because one most of spent time is in routines BLAS.

Organization (S): EDF/IMA/MMN
Data-processing handbook of Description
D2.01 booklet: Overload of a version in exploitation
D2.01.01 document
Rules concerning the extraction and
restitution of the sources of Aster
Summary:
One gives here the rules which the developers of Aster must comply with when they recover a source official (extraction) and when they propose a modification of a source official (restitution). These rules
points also approached in the plan of development [A2.01] specify.

---

1 Introduction

One explained in [D1.01.01] how a developer could "overload" Code_Aster with sources personnel. The sources about which one speaks here are:
- *of routines FORTRAN, or C or CAL (CRAY Assembling Language),*
- *of the catalogues,*
- *of the case-tests.*

The sources official of the Code are accessible in reading on the CRAY [D1.02.01 §2].
A traditional development of Aster results in general in:
- *modification of certain sources,*
- *the writing of new sources.*

The modification of sources supposes their preliminary extraction (recopy) sources official.
Paragraph 2
relate to the rules of extraction.

The finished development, it can "be restored": one puts up to date the sources then official.
Paragraph 3
relate to the rules of restitution.

---

2 Rules
of extraction

R1
The extraction of a source is the recopy of the official version [D1.02.01 §2] of this source.
(To avoid the copies of copies).

R2
When a source must be modified for a future restitution, "to note" this source [D1.02.01] (order asno of asterix). This notation gives rights of precedence for the restitution when a source is modified simultaneously by several developers: "noted" first is that which has the right to restore.

R3
When a source must be modified and restored: not to touch the "line" MODIF which contains the date of the last modification of the source. This date makes it possible to check at the time of restitution that the source of origin (before modification) is well last the source official.
This prevents that a modification "does not crush" a modification prédédente.
This rule is capital: an intentional violation would be very badly interpreted… and sanctioned.

R4
Not to modify (for the pleasure or by taste) the presentation of the source: the indentation, the name variables,… except if is to make it in conformity with the rules presentation or of programming [D2.02]; because this artificially increases the volume of the "delta of evolution" that one preserves: the relevant modifications do not jump any more to the eyes.

R5

If the source extracts was already noted by another developer:
· to contact it to make him confirm its intention to restore this source,
· to try to negotiate with him, the order of the restitutions,
· if there remains noted "in front of" you, not to launch out head lowered in a modification of great width of this source, because it will be necessary to start again the modifications in source official resulting from its evolution (cf R3).

R6

Not to note sources unnecessarily: you can géner the other developers (cf R5).
In particular, to note a source only one intends to restore in 6 months!

R7

When one noted a source by error, to indicate it [D1.02.01] (order asdeno).

3

Rules of restitution

R1

To use the automatic mechanism of restitution of asterix/agla: order asrest [D1.02.01].
This mechanism imposes that a restitution is autonomous: the official code modified by restitution must be able to carry out the case-tests of nonregression.

R2

A "asrest" erases another of them: there is no pile of restitution for a developer given. A developer cannot thus make more asrest per week for a version data of code (NEW2 or NEW3).

R3

A "restitution" (sources modified + sources added) will not be effective (modification sources official) that after the procedure of update of the code made by the administrator [D1.02.02 §1.2].

R4
To destroy the sources of a restitution only after having received (and read) the announcing "email" update of the code.
R5
Any restitution must be accompanied by a file of history (standard histor of asterix) in which the developer must document sobrement but with precision its restitution:
· new key words (vocabulary),
· new functionalities: a little mechanics and numerical analysis,
· numbers of the corrected anomalies: AL94-XX,
· numbers of the treated evolutions: EL95-XX,
· evolutions or additions of the cases tests.
Not to tire itself to list the names of the modified units of source: the system does it automatically.
R6
"to present" its restitution in meeting of EDA Monday. To bring a transparency of its file history.
R7
To think of the validation of its restitution: addition/modification of case-tests.
R8
To destroy a source (or to change it library) to see [D1.02.01 §3].
R9
When a new unit of source is added. This unit must contain a line "ADDITION" whose syntax depends on the type of the unit [D1.02.01§1].
C ADDITION nom_de_bibliothèque
for FORTRAN
%& ADDITION nom_de_catalogist (ORDER, TYPELEM,…)
for CATALOGUE
% ADDITION
for case test
% TITRATES…
Data-processing handbook of Description
D2.01 booklet: Overload of a version in exploitation
HI-75/95/009 Ind A

Author (S):
J. PELLET, P. MIALON

Organization (S): EDF/IMA/MMN

Data-processing handbook of Description

D2.02 booklet: Rules of development

D2.02.01 document

Rules of programming FORTRAN

Summary:

This document presents the rules retained by the Team of Development of Aster (EDA) for the writing of routines of the code in FORTRAN 77.

One can distinguish two kinds of rules:

· the rules resulting from book "FORTRAN 77 Guides for the writing of portable programs" (F. FICHEUX-VAPNE and coll). For these rules, we sometimes modified the statement in transforming a council into rule,

· rules resulting from the experience gained during the first years of the project. These last rules are sometimes dictated by strategic technical choices: for example, management of memory (JEVEUX) exemption often of the use of the COMMON. These last rules thus have a range less general than the first.

The compliance with these rules has two principal objectives:

· to ensure a good portability of the code,

· to ensure a good legibility (and thus maintainability) of the source text.

1
Contents

---

2 Introduction

The goal of this document is to present the whole of the rules retained by the Team of Development of Aster (EDA) for the writing of routines FORTRAN of the code.

The compliance with these rules has two objectives:

· to ensure a good portability of the code,

· to ensure a good legibility (and thus maintainability) of the sources.

It is obvious that it is not enough to comply with these rules to achieve the second goal. This one requires also rules of presentation [D9.03.01] and especially of the efforts on behalf of each programmer for to render comprehensible itself.

It is quite as clear that these rules relate to only the aspects of the language used and that other rules must be applied concerning the programming (emission of the messages of error, use of JEVEUX, use of the structures of data, etc) or the development (presentation, documentation, validation, etc). These rules are presented in the Handbook of Data-processing description and the Handbook of Administration.

Immediately let us note the **imperative** character of these rules. It is not a question of virtuous councils. Each rule is written so that one can say without ambiguity if it is respected or not: there is nothing of qualitative. The developers of the Aster code must respect them. We will see that the first stated rule (most important) is the respect of standard ANSI. The current tool for compilation of Aster (cft77 on CRAY) allows to check its application easily. Other rules are checked by the AGLA [D2.01.02] us will systematically indicate between brackets the emitted code-return when it is nonnull (2 or 4). The code return (2) allows the administrator sources of Aster (ASA) to control the departures from the rules. The code return (4) prohibited restitution of the sources (cf [§5]). For the rules whose automatic checking is less easy, the "sanction" will be done a posteriori: the procedure of evolution of the sources indeed makes it possible to find easily identity of a possible negligent developer.

One cannot speak about rules of programming FORTRAN at DER without speaking about book "FORTRAN77 Guide for the writing of portable programs " [bib1] carried out under the direction of F. FICHEUX-VAPNE. This book us was used as a basis for this document: practically, for the presentation of the rules, we preserved it plan: lexical elements,…, input-outputs. We retained:

· 18 councils of portability,
· 22 methodological councils.
The respect of the standard, which we instituted in rule n° 1 replaces 33 councils of the book.
The councils of the book were set up in rules, sometimes by modifying their statement: “to use… with prudence” -->
“not to use…”.
With these rules, we added some rules which are clean for us and which result from the gained experience
by the Team of Development during the first five years of the Project.
In this document, we tried to explain (at least partially) the reasons of the choice of these rules.
This is not always easy to make. For that, we return to [bib1] for the rules coming from this book,
and we make references in certain paragraphs of explanation for the rules which are clean for us.
Let us finish this introduction by saying that, contrary to a spread idea, FORTRAN is not one
“obvious” language. Certain elements of the language are “survivals” of old versions of the language. These
elements are not included/understood often any more “young” programmers. The curious reader will be able to read with profit it
deliver “FORTRAN 77” of H. KATZAN [bib3] for including/understanding well what is the FORTRAN77 of standard ANSI.

---

3 Rules reserves

3.1 Presentation

The rules are numbered. For the rules resulting from [bib1] we preserved same classification as
that of the book, which makes it possible to refer to it more easily, in particular to include/understand it why of this
regulate. The additional rules that we were given are noted Have.

Have:
ième rule Aster,
Pi:
ième council of portability of [bib1],
Semi:
ième council of methodological of [bib1].
In general, each rule is followed clarification of the exceptions to this rule (when they exist).
recall:
the code return of the AGLA (asverif) is written between brackets when it is nonnull: (2) or (4).
Some terms of Code_Aster
JEVEUX:
manager of memory of Code_Aster,
SUPERVISOR:
"main program" which connects (and supervises) the different ones
orders of the software,
UTILICRA:
library containing the nonportable routines (in their version
"CRAY"),
BIBC:
library containing some utilities writes out of C,
BIBCAL:
library containing some routines written in "assembler" CRAY
(CAL): optimization of the combined gradient.
3.2 Normalizes
ANSI
A-1
To respect standard FORTRAN77 ANSI. (4)
[bib2]
It is obviously the most important rule.
Data-processing handbook of Description
D2.02 booklet: Rules of development
HI-75/95/009 Ind A

Code_Aster ®
Version
3.0
Titrate:
Rules of programming FORTRAN
Date:
28/04/98
Author (S):
J. PELLET, P. MIALON

Key:
D2.02.01-A
Page:
5/18

3.3
Exceptions to the standard

The exceptions to the respect of standard ANSI are:

A-2
To use declarations REAL*8 and COMPLEX*16 (cf [§4.2]).

A-3
It is allowed to make ZK8 (I) = ZK8 (J) if ZK8 is a table of character strings.

A-4
It is allowed to use the functions except standard:
· IAND, IOR on type INTEGER,
· DIMAG and DCONJG and DCMPLX for type COMPLEX*16 (cf [§4.3]).

A-5
The tiny ones and certain special characters are allowed only for certain units of program (cf [§4.1]) (2).

A-6
It is interdict to use the FORMAT of impression of the binary variables except standard Z, B, O, etc… except for routine JJIMPO (2) making the "dump" memory.

A-7
It is allowed JEVEUX to put in equivalence variables of the character type and of type not-character (2). (cf also A-13)

A-40
It is allowed to pass in argument an expression of the character type of the style "chain" //arg (1: N) where arg is an argument unknown length.

Example:
SUBROUTINE AAAAAA (C)
CHARACTER* (*) C
CALL UTMESS (…, "LOUSE" //C (1: 4),….)

A-41
It is interdict to not use the constant data in binary format for the types floating. Exception routine DEFVEM (2).

A-44
It is safe interdict with JEVEUX (2) to put in equivalence a vector length 1 with a variable declared in COMMON.

3.4 Elements
lexical

P.I.6
To register the comments in lines comments comprising in first column C other than any other character (4).

A-49

The white lines are accepted.
P.I.8
Not to insert comments between lines continuations (0), nor before the headings
FUNCTION, SUBROUTINE, nor after END (4).
M.I.2
Not to use key words as identifiers: IF, END, CALL, GOTO,…
M.I.3
Not to put white inside the identifiers, of the key words (except for GO TO,
END IF, and ELSE IF) and of the constants except those of character type.
M.I.5
Not to cut at the end of the line the key words, the identifiers and the constants.
A-29
To use it & as character of continuation of the charts continuations.
A-30
To cut the lines to be continued so that the instruction is syntactically
incorrect without the charts continuations. For example by finishing the lines to be followed by: “,”
“/” “+” “-” “/” etc…
A-39
To give to each unit of program a name having between 5 and 6 characters.

3.5
Objects of the language

P.II.3
Not to put white inside a literal constant of arithmetic type or
logic.
P.II.6
To compare only arithmetic expressions in the same way standard.

P.II.8
Not to use the operators of comparison .LT., .LE., .GT and .GE on chains of characters that to compare between them chains only made up of figures or of the chains only made up of letters.
P.II.9
To use only intrinsic functions LGE, LGT, LLE and LLT to compare according to the order of the ASCII code of the chains made up at the same time of letters, figures and of special characters.
A-2
(Recall) To use declarations REAL*8 and COMPLEX*16.
A-8
Not to use the declarations REAL, COMPLEX and DOUBLE PRECISION DOUBLE PRECISION (4).
A-31
All the routines must comprise the instruction (4):
IMPLICIT REAL*8 [A-H] [OZ] or
IMPLICIT NONE
Those which will restore FORTRAN with IMPLICIT NONE will be considered better that others.
A-32
The local variables not used must be destroyed.
A-33
The instructions put in comments must be destroyed.
A-35
The labels should not appear that in front of the instruction CONTINUES or FORMAT (4).
A-36
To use: To ** 2 in the place of A*A or A ** 2.

3.6
Initialization - assignment

P.III.1
To use only whole expressions as values initial, final and like not implicit loop C.
M.III.6
Not to use instruction ASSIGN.
A-45
All numerical constants (smaller floating number, etc…) and mathematics (, 2, etc…) must be initialized by call to a function ENVIMA [D6.01.01]. If this one seems insufficient to emit an anomaly report.
A-46
It is useless and not very readable to use variables to handle constants numerical simple. For example, which should not be done:
REAL*8 ZERO, ONE
DATED ZERO, UN/0.D0,1.D0/
X = ZERO

MU2 = E/(UN+NU)
What it is necessary to do:
X = 0.D0
MU2 = E (1.D0+NU)
A-47
All the floating constants must comprise it. and D (4) (cf [§4.2]).

3.7
Structures of control

P.IV.1
To use only expressions of the whole type as parameters and meter of buckle.
P.IV.2
Not to modify the meter of loop in the body of the loop.
P.IV.7
Not to use the instruction PAUSES.
M.IV.1
To finish **each** loop C by a CONTINUOUS instruction.
P.IV.4
To use the instruction systematically CONTINUES, in particular as last instruction of the structures of control by having care, when these structures are imbricated, to allot CONTINUOUS to each one.
A-9
Not to use calculated GO TO.
Exceptions: routines TE0000, EX0000, EX0100, OPSEXE.
A-9
Not to use assigned GO TO.

A-10
Not to use the arithmetic IF.
A-11
Not to use the instruction safe STOP for routines JEFINI, JVFINM, JVVTAM (2).
A-34
The blocks IF empties are prohibited.
A-38
The blocks IF and the loops C must be indentés of two characters.
Example:

```
C 100 I=1, NR
X (I) = 0.D0
100 CONTINUOUS
C
IF (Y.LT.0.D0) THEN
Z=1.D0
ELSE
Z=2.D0
ENDIF
```

3.8 Units of program

P.V.4
Not to use the intrinsic functions TANK and ICHAR (4) (cf A-15).
Exceptions:
· routines of reading of the supervisor: LXSCAN, LXCAPS, LXINIT (2),
· JEVEUX: JEDEBU, JJVERN (2).
P.V.9

To use one type for the variables of a given COMMON. (cf [§4.4]).
P.V.12
To use instruction SAVE with each time remanence is wished.
M.V.3
Not to use the intrinsic functions as arguments of subroutine.
M.V.8
To finish an external function by END without coding RETURN. Thus not to use RETURN (4). It is not prohibited however to use:

```
GOTO 9999
...
9999 CONTINUOUS
END
```

M.V.9
Not to define external functions having the same name as functions intrinsic.
M.V.12
Not to use in a subroutine the optional returns of subroutines.
M.V.13
Not to use instruction ENTRY (4) except ENVIMA, GETVAL and VALXEM (2).
M.V.16
Not to use arguments of the form *étiq (cf M.V.12).
M.V.17
Not to use a BLOCK DATED (4).
M.V.20
Not to use the order DIMENSION: it is simpler to declare dimension with type (4).
M.V.21
To define only one commun run for each instruction COMMON and to use only one instruction COMMON by commun run.
M.V.22
For objects belonging to a commun run, to use the same names in all them units of program where this one appears.
M.V.23
Not to use the common white (common without name).
M.V.27
To use the notation * to code the upper limit of the last dimension of one table used as formal argument when this terminal is unknown under program.
M.V.28
To note * the length of a character string used like formal argument (4).
A-12
Not to use instruction PROGRAM. Exception: routine aster.
A-13

Not to use the instruction EQUIVALENCE (except for routines JEVEUX of UTILICRA)
(4) (cf also A-7).
A-14
Not to use instruction INTRINSIC (4).
A-15
To use only the authorized intrinsic functions (cf [§4.3]) (4).
A-21
The use of the COMMON is reserved for very particular uses (cf [§4.4])

A-26
To limit to 300 the number of lines of a routine (2).
A-27
To limit to 15 the number of arguments of a routine (2).
A-28
To preserve if possible the identifier of the arguments of the routine called in
appealing routine.
A-42
To use instruction external only in the event of need: when the “external” routine
passed in argument.
A-43
Not to call upon external libraries: NAG, GENERAL, LINPACK… (4).
A-48
To make external calls only in UTILICRA and BIBC (cf [§6]) ((2): for
UTILICRA and BIBC) ((4): for the others).
A-50
Not to use the BLAS (4) but their “hats” which are in the library
UTILICRA [§6.1]. Exception: UTILICRA (2).

3.9
Input-outputs

P.VI.4

At the time of a READ or WRITE, to specify the selected logical unit, by parameterizing sound number (variable integer). Not to use the asterisk.

P.VI.6

The value of the code return IOSTAT depends on the calculator. The standard indicates only that it will be null if very happened well, positive if there were error, negative if an end of file were met. To use only this property.

P.VI.7

To use only expressions of the whole type for the values initial, final and for the step value of the implicit loops C in the lists of input-output.

P.VI.8

To always give a list of input-output in the writing and reading orders and not to use the empty format.

P.VI.18

In an instruction FORMAT, not to use the specifications of edition, T, TL and TR.

M.VI.2

Not to use instruction PRINT.

M.VI.5

Not to use the descriptor nHh1… hn but “h1h2… hn”.

A-16

Not to use the OPEN instruction
Exceptions: routines, jxlir1, jxouvr, spycod, spyerr.

A-17

Not to use instruction BACKSPACE.

A-18

Not to use instruction INQUIRE.

A-19

Not to use the CLOSED instruction.
Exceptions: routines spycod, spyerr.

A-20

Not to use instruction ENDFILE.

3.10 Problems
of inter-compilation

A-22

To use the same type and the same length for a variable put in COMMON in all the routines which use it (4).

A-23

To call the routines with the good number of arguments (4).

A-24
Not to call a routine with an argument of type CHARACTER length different that that awaited by the program called (4).
Regulate for the routines having a formal argument of type CHARACTER
· to use declaration CHARACTER* (*) (M-V-28),
· to recopy if necessary arguments CHARACTER* (*) in variables local.

A-25
Not to call a routine with arguments of a type different from that which it waits. In particular not to call a routine with a complex argument if that Ci awaits two real arguments and conversely (4).

4 Some explanations

4.1 Alphabet

Certain routines use "illegal" characters (by the standard).

· The supervisor must be able to read the characters authorized in the process control language of Aster:

- tiny
- %, &! , _.

· JEVEUX uses: $ and &.

· The routines printing of the lines of "orders UNIX" use: $, I ("pipe" UNIX), \.

Summary of the exceptions concerning the alphabet:

That is to say: NR
= {characters authorized by the standard}
= {AB… Z01… 9 " white " ***/(). ': $}

· authorized characters: NR - {$} + {&}
all them routines

· authorized characters: NR + {&}
JEVEUX

· authorized characters: NR - {$} + {tiny, %, &! , _}
SUPERVISOR

· authorized characters: NR + {I, \}
impressions UNIX

The comments use only the characters authorized by the standard.

4.2

Declaration of the floating types
This problem is related to the double following requirement:
· to allow calculations with a precision considered to be reasonable: 13 significant figures on
CRAY,
· to be portable on machines whose REAL is length 32 bits.
Unfortunately these requirements are not compatible with the standard.
The “solution” chosen for this problem is detailed in note HI-75/94/068/A “the problem of the numbers floating in FORTRAN77 ".
Adopted solution:
· The only authorized orders of declaration are:
IMPLICIT NONE
) with
IMPLICIT REAL*8 [A-H] [OZ]
) choice
LOGICAL
INTEGER
REAL*8
COMPLEX*16
CHARACTER*…
If not codes return (4)
· Écrire constants in double precision double precision.
If not codes return (4)
REAL*8 R1, R2
COMPLEX*16 C1, C2

R1 = 1.D0/3.D0
C1 = (3.D0, 4.D0)

It is thus interdict to write:
R1 = 1. /3.
R1 = 1.E3/3.E3
C1 = (0. , 1.)
C1 = (0.E0,1.E2)

4.3 Functions

intrinsic

Intrinsic functions of the language: sines, cosine, racine_carrée, valeur_absolue,… have for the majority of them a generic form (i.e independent of the type of their arguments). They are these functions credits which should be used to ensure the portability.

Example: racine_carrée function
REAL
specific name: SQRT
DOUBLE PRECISION DOUBLE PRECISION
specific name: DSQRT
COMPLEX
specific name: CSQRT
Generic name: SQRT

In the same way, the conversion of the type must be done in a generic way. For example, conversion into entirety (truncation):
Specific:
INT:
REAL
->
INTEGER
IFIX:
REAL
->
INTEGER
IDINT:
DOUBLE PRECISION DOUBLE PRECISION
->
INTEGER
Generic name: INT

The intrinsic functions selected (cf notes HI-75/94/068/A) are:

· Fonctions of standard FORTRAN

Conversions of the type:

INT
DBLE

Generic arithmetic functions

AINT
ANINT
NINT
ABS
MOD
SIGN
DIM
MAX
MIN
SQRT
EXP
LOG
LOG10

Generic goniometrical functions

SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2
SINH
COSH
TANH

Functions particular to the character type

TANK
ICHAR
LEN
INDEX

· Fonctions except standard to be used

Conversions of the type:

DCMPLX

Functions particular to the complex type

DBLE
DIMAG
DCONJG

Functions particular to the integer type

IOR
IAND

4.4 COMMON

The manager of memory JEVEUX allows the routines of the code to exchange data structured in minimizing the number of the arguments: one transmits the name of the structure of data.

The use of COMMON is thus not recommended in Aster. It is not interdict to however use them. This use must however remain limited to "parcellings" of routines identified well. The use of COMMON is not to limit the number of the arguments passed to the routines. The use is justified:

· for reasons of performances and only for the routines of general use (CALCULATION, JEVEUX, SUPERVISOR, reading of the CATALOGUES),

· for the passage of the parameters to the routines used in external.

The use even of JEVEUX is impossible without "COMMON JEVEUX" [D6.02.01]. For those, one will always use the same sequence of declarations (recopied routine in routine):

```
C
C ---
BEGINNING DECLARATIONS NORMALISEES JEVEUX ---------------------------
----
```

C
CHARACTER*32
JEXNUM,
JEXNOM,
JEXR8, JEXATR
INTEGER
ZI
COMMON/IVARJE/
ZI (1)
REAL*8
ZR
COMMON/RVARJE/
ZR (1)
COMPLEX*16
ZC
COMMON/CVARJE/
ZC (1)
LOGICAL ZL
COMMON/LVARJE/
ZL (1)
CHARACTER*8
ZK8
CHARACTER*16
ZK16
CHARACTER*24
ZK24
CHARACTER*32
ZK32
CHARACTER*80
ZK80
COMMON/KVARJE/
ZK8 (1),
ZK16 (1),
ZK24 (1),
ZK32 (1),
ZK80 (1)
C
C ---
FINE DECLARATIONS NORMALISEES JEVEUX -----------------------------
----

For the other COMMON we will adopt the rule of following denomination:
COMMON TxyzPP where:
T is a character which indicates the type of the variables of the COMMON:

I
INTEGER
R
REAL*8
L
LOGICAL
C
COMPLEX*16
L
LOGICAL
K
CHARACTER
PP is a pair of characters which identifies at the same time a "package" of commun runs and parcelling routines which use this package.
xyz is three free characters making it possible to differentiate the commun runs from the same package.
Examples of what **could** be made:
PP = "I": COMMON necessary to JEVEUX
PP = "CA": COMMON necessary to the routine CALCULATION,
PP = "LC": COMMON necessary to routine NMCOMP,
PP = "GC": COMMON necessary to the routines of the supervisor.

4.5
Input-outputs
The inputs/outputs of the code are made in theory by a restricted number of routines:
· read/write on the data bases Aster (files of direct access): JEVEUX.
· formatted reading:
-

the supervisor is in charge of the reading of the command file,
- only orders (= GRID, FUNCTION,…) and them
orders of interfacing (= GIBI, IDEAS,…) are authorized with reading
on external files. For that, the only authorized instruction is the formatted READ.
The opening (OPEN) and the closing (CLOSED) of the files are made by the supervisor.
· formatted writing:
-
emissions of messages on the files ERROR, MESSAGE, RESULT: parcelling of
routines UTMESS, UTDEBM,… [D6.04.01],
-
writing of results: orders (= RESU, FUNCTION,…),
-
writing of information in the files RESULT or MESSAGE (key word IMPR of
certain orders).

Summary of the authorized instructions:

UTMESS (...)
all routines.
READ (nfic, fmt)
Orders
WRITE (nfic, fmt)
Orders and key word IMPR
OPEN/CLOSE,…
SUPERVIS, JEVEUX
direct access
JEVEUX

5

“Errors” detected by the compiler of the CRAY

The compiler of the CRAY cft77 [bib4] emits a certain number of messages classified in various categories:

· How
· Note
· Caution
· Warning
· Error
· ANSI
· ...

These messages have a heading standardized: cft77-i where I is a number which can belong to [1-9999], (all the numbers are not allotted!). Only the “Error” are “fatal” for the compiler, but the others messages (which we will call “errors” with quotation marks) can be revealing problems of programming.

The explanation of each one of these “errors” is given in the document [D2.02.02].

We thus decided to treat in “error” some of these messages (in other messages ANSI).

During a compilation (for example with the restitution of source), it is easy “to catch” the messages correspondent with a given number. One gives then a code return to compilation according to the level maximum of the detected “errors”.

The value of the code return is that of the restitution (cf asrest [D1.02.01]):

0
All is well
the restitution is accepted.
2
Alarm
the restitution is subjected to the visa of the ASA.
4
Error
the restitution is refused.

All the messages of cft77 lead to a code return 4 except:

· code return 2:
720 - Regulate A-7
726 - Regulate A-44
890 - Regulate A-5
· code return 0:
118 - Regulate A-4 and A-15
408 - Regulate A-2
881 - Regulate A-3
895 - Regulate A-40
· code return 4:
342 except for DEFVEM - A-41 Rule
753 except for JEIMPO - A-6 Rule

Data-processing handbook of Description

---

6 Calls
"external"

We call "external" call, a call towards an entrance point which is not:

· neither a routine (or ENTRY) FORTRAN nor a function C of Aster.
· nor an intrinsic function

Only libraries UTILICRA and BIBC can carry out "external" calls.

Let us recall that the call to a routine of an external library (NAG, GENERAL, LINPACK,…) is prohibited (A
43).

Ultimately an "external" call can be only the call to:

/
a routine BLAS
/
a specific function CRAY
/
a routine of BIBCAL

6.1 BLAS
:
SAXPY,
SCOPY,
SDOT,
SNRM2,
SSCAL,
SPDOT*

These routines change names on the machines "32 bits": DAXPY,…

Regulate:
To use the "hats" with these routines which are in UTILICRA:

R8AXPY, R8COPY, R8DOT, R8NRM2, R8SCAL, R8PDOT
* SPDOT is not an official BLAS.

6.2

Specific routines CRAY

· **Direct Access:** OPENDR, CLOSDR, READDR, WRITDR
· **"traditional":** AND, XOR, SHIFT1, SHIFTR, INTMAX, LOC, STRMOV
· **Temps:** CLOCK, DATE, TREMAIN, SECOND
· **Environnement UNIX:** GETCWD, GETENV, GETPID
· **Dynamic Storage:** HPALLOC, HPDEALLC, HPCHECK

6.3

BIBC

All the routines can call upon the routines of library BIBC. But this one does not have to contain that utilities which cannot be carried out in FORTRAN.

6.4

BIBCAL

Seul BIBC can appeal there.

---

7 Bibliography

[1]
FORTRAN77 - Guide for the writing of portable programs under the direction of F. Ficheux-Vapné - Editions Eyrolles Collection of the Management of the Studies and Research of Electricity of France
[2]
FORTRAN77 normalizes - ANSI
X3.9-1978
[3]
FORTRAN77 - Harry Katzan - Van Nostrand Reinhold Company 1978

[4]
CF77 Compiling System, Volume 2: To compile Manual Message - SR-3072 4.0 1990 - CRAY research, Inc.

Organization (S): EDF/IMA/MMN

Data-processing handbook of Description

D2.03 booklet: Establishment of the catalogues

D2.03.01 document

Rules concerning the writing of the Catalogues

Summary:

We give in this document, the rules (or councils) which must comply with the developer when it add or modifies a catalogue of order or a catalogue of finite element.

Contents

Data-processing handbook of Description
D2.03 booklet: Establishment of the catalogues
HI-75/95/009 Ind A

---

1 Presentation

We indicate here the rules (and councils) concerning the writing:

· of the catalogues of orders [D5.01.01 §2]

· of the catalogues of elements [D3.02.01 §3]

2 Catalogues of orders

1) As much as possible to use the possibilities of the supervisor concerning exclusions, the values by defect,…

2) Not to use the concept of argument multiple.

3) When an argument is of type "text":

- if it can take a finished number of values, to define it in TXM and to give the integral list of possibilities by the key word IN.
- if it is a free text: comment,… to use TX

4) The comments are welcome.

5) to make validate the vocabulary by the Project Leader.

3 Catalogues elements

3.1 catalogue of PHENOMENE_MODELISATION

1) The names of the phenomena and modelings must be validated by the Project Leader because they appear with the user.

3.2 catalogue SIZES

1) To give a name to the sizes of form where S can be worth:

- R: reality
-

C: complex
-
F: function (K8)
2) When one does not want to create a new too particular size, to use the sizes
“neutral”: NEUT_R or NEUT_K24.
3) When the catalogue of the SIZES is modified, to think of updating the document “description
sizes " [D4.04.02] and to classify the names of sizes alphabetically.
4) Not to define sizes of the type: L, K32, K80
5) To destroy CMPS in an existing size without to have checked only any type_elem
uses.
6) Not to change the order of the CMPS of an existing size without modifying the type_elem which
use.
7) When one introduces a new component into a size, to put it following the CMPS
existing. This avoids “breaking” programming too much “into hard”; for example, one
programmer can have made:
· checking that “DX” and “DY” are the first two CMPS of size “DEPL_R”,
· then use of DEPL_R (I), I = 1,2.

3.3 catalogues

OPTIONS

1) The names of OPTIONS must be validated by the Project Leader if they appear with the user.
2) Not to reinvent the names of the parameters for each option; to take as a starting point those already selected.
The usual form is as follows: nom_par = “P” //nom_gd. Examples: PMATERF, PGEOMER
3) To comment on each field parameter: example: PGEOMER, GEOM_R % field of geometry
4) When a new option is added, to think of putting “- 1” in the catalogues of
type_elem which erudite step to calculate this new OPTION yet.

3.4 catalogues

TYPE_ELEM

1) For the names of the local modes to take as a starting point the the names chosen by the close type_elem.
To respect the use:
-
Cxxxx: mode of the chart type
-
Nxxxx: mode of the chamno type
-
Exxxx: mode of the chamelem type
2) For a local mode of type CHAMELEM of the type “NR nno…”, to make sure that nno is well it a number of nodes of the associated type_maille.
3) When new a type_elem is added, to think of putting “- 1” compared to the OPTIONS existing that new the type_elem cannot calculate yet.
4) To put questions about the coherence of the type_elem which one modifies with the others type_elem:
-
why new the type_elem would have it a local mode “with the nodes” whereas all them do others have it “at the points of GAUSS”?
-
why new the type_elem use doesn't this field parameter, this CMP?
- ...

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D2.05 booklet: -
D2.05.01 document

Rules concerning the Structuring of the Data

Summary:

We indicate here the rules (and councils) concerning the Structuring of Data (SD):

- ***to define new types of SD,***
- ***with which objects JEVEUX?***

The reading of this document supposes the preliminary reading of the document [D4.01.01].

Titrate:
Rules concerning the Structuring of the data

Date:
01/12/05
Author (S):
J. Key PELLET
:
D2.05.01-B Page
: 2/4

1 Presentation

We indicate here the rules (and councils) concerning the Structuring of Data (SD):

·
to use new types of SD,
·
with which objects JEVEUX?

The reading of this document supposes the preliminary reading of the document [D4.01.01].

2
To define and use new type_SD

1-
To think "Structures of Data": not only for the Structures of Data "users" but also for all the objects of work. As soon as one must handle several objects "at the same time", to gather them in a type of Structures of Data that one will document.

2-
"To consult the asset":
-
to know the types of existing SD,
-
to measure the adequacy of the types of existing SD to its need,
-
to put the question (and to pose it with the others): is necessary it to modify an existing SD or in to create a news. In particular, it is necessary to avoid the multiplication of the types of SD

"user",

-

any modification (or introduction) of type of SD must be discussed in EDA.

3-

To use the names of the types of Structures of Data in the comments of the routines. Scrupulously to respect for that the orthography of these names [D2.01.02].

To write for example:

SUBROUTINE LOUSE (CART, NETTED)

C IN CART
: NAME Of a SD CHART
C IN NETTED: NAME Of a SD GRID

4-

To use new suffixes for very new type of Structures of Data. The list of suffixes currently used is given in [D4.01.02]. For reasons of legibility, one can also begin its suffixes with a common character string.
Example: SD LISTE_RELA (K19)
suffixes: ".RLCO", ".RLDD", ".RLNO",… (" .RL " --> "linear relation").

5-

A suffix must start with one "." (legibility).
One can use the characters: "A, B,… Z, 0, 1,… 9, _, &."

6-

To respect the "standard" lengths for the names of the Structures of Data: K8, K14 or K19.

7-

Not to multiply objects JEVEUX to make "prettier":

-
one can store 1 K8 and 1 K24 in a vector of 2 K24
-
Boolean insulated can be coded in an entirety (0 or 1),
- ...

Data-processing handbook of Description
D2.05 booklet: -
HT-66/05/003/A

8-
When one “begins” the structuring (for the types of Structures of Data moreover low level), to use the longest names (K19) to be able later on to create the new ones type of Structures of Data the container. (i.e to begin the suffixes “with the line”).

Example:

type_1er (K19) record
“. T1AA”:
OJB
...
“. T1BB”:
OJB
...
type_2nd (K14) record
“. T2AT1”
:
type_1er

...
The type of SD type_1er can be used as article of the type_2nd type.

9-
To think of the references “upstream”: it is sometimes useful to store in a Structure of Data names of the Structures of Data which gave him birth. This supposes that these Structures of Data are perennial (have an image in the base “TOTAL”).

10-
To avoid the redundancies within a type of Structures of Data, because the update becomes more problematic. Bad example: listr8 (list of realities) where one stores at the same time the complete list of realities and the list of the intervals of constant step (one of objects (.VALE) can constantly be recomputed with the others).

11-
When one declares types of Structures of Data "hats", for example:

type_1 (K8):: = record
/
$VIDE
:
type_2
/
$VIDE
:
type_3

it should be made sure that there is a means "of crossing" them"/"; i.e. to know to recognize if the Structure of Data is of type_2 type or type_3.
12 - Problem

"Hidden" SD:
It happens sometimes that a SD stores the name of other SD (for example, the SD_RESULTAT stores in its object .TACH the name of the fields which make the SD_RESULTAT). When SD referred is unknown (or hidden) of the user (it is the case of the fields of SD_RESULTAT), it is necessary to find a name for these referred SD.

The following rule will be adopted:
If one must name a hidden SD that one reference in a SD user named NOMU (K8), one will give to this hidden SD a name starting with NOMU.

Thanks to the compliance with this rule, order DETRUIRE/CONCEPT=NOMU will destroy correctly ALL objects JEVEUX created by the order having created NOMU.

To obtain a name of hidden SD complying with this rule, one can use the routine GNOMSD.

8.1

Titrate:
Rules concerning the Structuring of the data

Date:
01/12/05
Author (S):
J. Key PELLET
:
D2.05.01-B Page
: 4/4

3
Basic objects JEVEUX

1-
Not to use attribute "DOCU" of the OJB.

2-
To try not to use "LONUTI": one will in general seek to allocate "with just" them objects. In this case, "LONUTI'=' LONMAX".

To use attribute "LONUTI advisedly": it is with the user to update it; one should not he to give another significance that his: length really used of a vector.

3-
For the collections which one knows that they will be never very large, it is preferable of to create contiguous. In the contrary case, they should be created dispersed.
It will be said that a collection (or an object) is large if:

-
its volume can be higher than 1Méga word,
-
or if its volume can be higher than 10 times the number of ddls of the model.

4-
Not to use the pointers (name or length) divided between several collections.
If (for example) 2 collections must be reached by the same names, one can make:

-
to create a pointer of names,

-
to create the collections in "numbered" access
- to make
JENONU before the access to the collections.

5-
Not to store in OJB of the addresses memory of other OJB (because an address memory is by "temporary" definition). One does it for the type of Structures of Data mater_code and for well identified reasons of performance, but that must remain exceptional.

Organization (S): EDF/IMA/MMN
Data-processing handbook of Description
D2.06 booklet:
D2.06.01 document
Use of JEVEUX
Summary:
It is a question here of indicating some concepts of operation of the manager of memory JEVEUX in order to
to specify the use of the routines "user", to indicate the routines the best appropriate ones to certain actions and
to announce the difficulties of use. One presents here the rules of use in italic in each paragraph.

Contents

1

Use of the bases
The simple objects can be created by routines JECREO and WKVECT, the collections by the routine JECREC.
WKVECT makes it possible to connect three calls JECREO, JEECRA and JEVEUO for an object of vector kind.
The basic concept makes it possible to associate the various objects a file safeguarded or not at the end of the work.
objects to be preserved at the end of the work will be created on the TOTAL basis associated the class G. This base
allows to preserve the structures of data and to carry out continuations.
The objects of work will be created on the VOLATILE basis associated the class V. This base is destroyed at the end
work. By convention, one will use the characters && at the beginning of the name of any object associated with this base.
The LOCAL base associated the class L is reserved to the Supervisor.
It is pointed out that it is not possible to have objects of identical name on different bases.
routine I…. do not have among their arguments the name of the class and the scan for the noun is carried out
in the whole of the repertories associated with the various open bases.
base TOTAL
name of class G
preserved objects
base VOLATILE
name of class V
temporary objects
base LOCAL
name of class L
reserved to the Supervisor
2
Access by name
The access to the objects managed by JEVEUX is carried out using the name. One uses a function of coding which
provides starting from the name and various parameters a key of access (an entirety), this key allows then
to reach the various attributes. The access by name is relatively expensive (decoding of characters, management
of collision, etc…) also one in a variable preserves the last name (of simple object, collection and object
of collection) and the identifier (obtained starting from the key) associated, to avoid a new call to the function of
coding.
Note:
It is thus recommended to carry out all the request for the same object JEVEUX in way

sequential in order to profit from this possibility.
Data-processing handbook of Description
D2.06 booklet:
HI-75/95/009 Ind A

3
Access to the segments of values

Routines WKVECT and JEVEUO return to the user a relative address in one of tables ZR, ZI,
Will ZC, or ZK, (one note Z thereafter? one of these variables FORTRAN). This address is valid as long as there is not
have release.

The concept of access in writing or reading makes it possible to avoid systematic unloading on disc of segment of values and limit thus the number of the inputs/outputs on disc. Objects reached in reading will not be safeguarded on disc at the time of the release. The call to WKVECT carries out a request in writing.

A segment of values could be reached in writing then released, the manager then preserves it in memory
and its unloading differs on disc at the time of a forthcoming research of place. A new access in reading
return the address of the segment déchargeable, a modification of the contents can thus take place without the knowledge of
the user if this last does affect the contents of table Z? with the address indicated.

Regulate use:
The user, when it carries out an access in reading, should not modify the contents of the table
Z? with the address provided at the time of the request and must avoid passing it in argument of under program it would not have the whole control.

Do the calls to routine JEVEUO return an address compared to a variable Z? of the same type as the object
JEVEUX (this address is measured in the length of the type). The standardized commun run must appear in all

unit of program carrying out this type of call.
INTEGER
ZI
COMMON/IVARJE/
ZI (1)
REAL*8
ZR
COMMON/RVARJE/
ZR (1)
COMPLEX*16
ZC
COMMON/CVARJE/
ZC (1)
LOGICAL
ZL
COMMON/LVARJE/
ZL (1)
CHARACTER*8
ZK8
CHARACTER*16
ZK16
CHARACTER*24
ZK24
CHARACTER*32
ZK32
CHARACTER*80
ZK80
COMMON/KVARJE/
ZK8 (1), ZK16 (1), ZK24 (1), ZK32 (1), ZK80 (1)
Note:
Not to modify the noun of the variables of the commun run of reference.
The access to a segment of values is carried out in the following way: if JTAB indicates the address returned by
routine JEVEUO for an object of vector kind and type I, KTAB that for an object of the type C (complex):
ZI (JTAB)
is the first value of a vector of entireties,
ZI (JTAB + I - 1)
is Iéme value of a vector of entireties,
ZC (KTAB + I - 1)
is Iéme value of a vector of complexes.
Data-processing handbook of Description
D2.06 booklet:

HI-75/95/009 Ind A

4 Initialization of the values

At the time of the call with JEVEUO or WKVECT the manager of memory carries out a search for place in
main memory. If the object does not have an image on disc (i.e. at the time of its first access in writing), the segment of value is initialized according to the type of the object: 0. for realities, (0. , 0.) for complexes, 0 for the entireties, "(white) for the characters.

Regulate use:

It is useless to carry out a loop of initialization before the first use of the segment of values.

5 Release of the segments of values and concept of mark

If nothing is done (not call to JELIBE), the segments of values brought back in memory remain there and that can
to lead quickly to its saturation. On another side, if one releases an object without taking precautions (one mark), one is likely to return invalid the address of an object requested upstream of the programming. One
systematic solution with this problem is not adopted yet today.

What is currently made (and which could be called into question):

· one releases few objects (the large ones) what makes it possible to make sure that the releases are not dangerous (good knowledge of the use of these objects);

· one uses the marks to release the objects.

The rules of use concerning this problem will be soon selected. While waiting:

Regulate use:

Not to release the objects systematically.

6 Safeguards

Routine JESAUV makes it possible in theory to protect the objects which one has just created (to avoid rewriting them
contents). In practice, this routine is almost never used. For the homogeneity of the code it us
seem that a rule should be selected: to save (or not) all the concepts. This decision was not yet

catch.

Regulate use:

Not to use JESAUV and JEDELI.

---

7 The allowance of a great segment of values

A lack of place in the space managed by JEVEUX led to the brutal stop of the program (error message according to: <S> <JJALLS02> CLOSING OF the BASES ON ERROR JEVEUX), only concepts created by
the preceding operators are preserved in the TOTAL base. It is possible to avoid this type of problem by taking the precaution to call upon routine JEDISP in order to obtain the high limits of size in the order descending of N zones of space available in memory (N being an entirety provided by the user).

Regulate use:

This call must be reserved for the allowance of very large objects in portions of significant programming (for example solveurs).

Indeed the research of the storage areas requires a complete course of the chaining of the segments of value and can appear expensive.

8 Detection of crushings report

JEVEUX dynamically manages a zone sequential memory whose size is determined by the parameter memory defined in the completion of the work. The various segments of values associated with the objects are deposited
in this zone ones behind the others. Each segment of values is framed by 4 entireties in front of and 4 behind. These entireties make it possible to define a chaining before and a back chaining, to store the state and it
statute, the identifier and the class as well as a shift to manage the types of length higher than the unit of addressing.

The crushing of the entireties located around the segment causes the rupture of the chaining and the loss of the identity of
the object associated with the segment with values. Such a crushing is generally detected at the time of one
search for place in memory (at the moment when the chaining is traversed) and not at the time of the overflow.
It results in one of the following error messages:
<S> <JJLIRS> <ECRASEMENT POSSIBLE UPSTREAM ADRESSE> nnnn
In this case one crushed one of the entireties located in front of the segment of values.
<S> <JJLIRS> <ECRASEMENT POSSIBLE DOWNSTREAM ADRESSE> nnnn
In this case one crushed one of the entireties located behind the segment of values.

Regulate use:

The developer has routine JXVERI then to instrument its code.
This routine checks the integrity of the chaining and can announce the frangible joint. It detects in more one
incursion out of the zone licit memory. It is possible to implement at less expenses in
order BEGINNING or CONTINUATION the call to this subroutine before each order, which allows
to determine which order carries out crushing. The developer will be able then, by instrumenting them
routines associated with the order, to proceed by dichotomy to determine the routine or the instructions erroneous.
Crushing can also be less important and affect only one word in front of or behind the segment of values
without breaking the chaining, it is the case when one makes an error of index in table Z?. Contents of
the use or of the statute of the segment of value is then affected, this information can be obtained while consulting
the result of the impressions of the memory division by routine JEIMPM.

9

To increase a vector
Regulate use:
The user has routine JUVECA to increase a simple object of vector kind.
It is a surcouche written starting from the routines user JEVEUX. It builds a temporary object and destroys
the original after recopy. One obtains the new relative address in return among the arguments. These various
operations can be rather expensive, it is thus preferable to minimize the number of calls for one
even vector and rather to double the size that to increase it progressively. Attention, this routine destroys
mark associated with the initial object and assigns the current mark to the new object.

10
The recopy of the objects
It is possible to recopy objects JEVEUX on the same basis or from one base to another. The recopy of
simple objects does not pose a particular problem, on the other hand it is more delicate to handle
collections.
A collection can be pressed on an external repertory of names or an external pointer length. These
simple objects must be created and partly managed independently (for example their destruction must be
explicit). Their name can thus be without relationship with the name of the collection.
Two cases are possible:
· the recopy is carried out on different bases: the external pointers will be duplicated and
will become internal pointers with the collection,
· the recopy is carried out on the same basis: the external pointers can be preserved or
well it are duplicated and become internal.
If the receptacle is already existing, it is destroyed before recopy.
Regulate use:
To use JEDUPO or JEDUPC - not to use COPISD, COPIOB or COPIOC.

11
Routines working on groups of objects

The organization of the structures of data of Aster rests mainly on the names of the objects. One handle within the code of the "concepts" built starting from a name provided by the user like result of orders. It thus appeared convenient to be able to handle group of objects while providing under character string, which is required in the names of all the objects present in the repertories.
Routines JELIBC, JEDETC and JEDUPC apply to lists of objects. They allow, in the order, of to release, to destroy and duplicate the objects of these lists.
These routines offer more flexibility to the developer to manage the objects (structures of data).

Note:

It is little recommended to use them in the subroutines of low level, the cost of seek using the under-chain being able to affect the performances appreciably.

Mechanical department and Digital Models

Index:
With
Diffusion:
Developers

Data-processing handbook of Description
D2.07 booklet:
D2.07.01 document
Rules concerning the inputs/outputs

Summary:

This document lists the rules concerning the inputs/outputs which the developers must respect of Aster.

EDF
Direction of the Studies and Research
Electricity of France

Code_Aster ®
Version
4
Titrate:
Rules concerning the inputs/outputs
Date:
28/01/1999
Authors:
J. PELLET
Key:
D2.07.01
Page: 2/6

1
Introduction

The Aster orders exchange data. Generally these data are Structures of
Data (or concepts "user") for which, the programmer does not take place to make "READ" (for its data) nor of "WRITE" (for its results). In this case, the "left" "entries"/are made by JEVEUX [D6.02.01].

It happens however that **certain** orders (in general of the procedures) have to read data on one file or to write a result. The rules concerning these orders are given to [§4].

At the time of a calculation, an order can want to transmit an **alarm** or error message, in it case, it will use "package" UTMESS [D6.04.01]. One will speak again about it to [§5].

Lastly, an order can want to write messages of information concerning the unfolding of calculation. These impressions are controlled by the key word INFORMATION of the order. These impressions are the subject of
[§6] for which one uses package INFXXX [D6.04.02].

2
Various files of Aster and their use

2.1
Data bases: "TOTAL", "VOLATILE" and "LOCAL"

(IN/OUT) they are the files with direct accesses managed by JEVEUX. JEVEUX is the only one with reading/writing on these
files.

2.2
General and essential files

·
(IN) command file (.comm) and (of include): these files are only read by
supervisor,
·
(OUT) file "error": only UTMESS can write there; a trace of the errors there is found,

·
(OUT) file "MESSAGE": one finds there:
-
impressions of the UTMESS,
-
impressions of INFORMATION [§3.5],
-
echoes of the orders by the supervisor,
-
the result of the "debug post mortem' if the execution finishes badly.
·
(OUT) file "RESULT": one finds there information asked explicitly by the user
: impression with the format "Aster" of the results (orders IMPR_RESU,…),
be added to that:
-
impressions of the UTMESS: alarms and errors,
-
the summary of the times placed in the various orders.

Note:

Approximately, the difference between the files "MESSAGE" and "RESULT" is as follows:
·
the file "RESULT" contains information interesting the sleeping partner of
the study,
·
the file "MESSAGE" contains information interesting that which makes a study
to control the courses of calculations.

2.3
Particular files

They are the other files. They are known as "private individuals" because they are used only by some orders
or procedures. For example:
·
file grid Aster (.mail),

·
file grid Gibi (.mgib),
·
file result Castem (.cast),
·
...

3 Various types of messages

When a programmer wishes to print information in a file, it is important for him
to try "to typify" this information, because from this type, the name rises from the file where one prints and the way
to print: UTMESS or WRITE.

3.1 The error

It prevents the continuation of calculation. One must always use UTMESS <F>, <E> or <S> (cf [D6.04.01]).

3.2 Alarm

The continuation of calculation is possible but the use is strongly disadvised;
Examples: negative Young modulus, affected in double,…
UTMESS <A> should be used

3.3 The result

It is a **result** of the **study**. It **is asked explicitly** by the user.
It is necessary to use WRITE (IFR,…)…
where IFR is the logical unit of the file **result**:
·
"RESULT" (with the format Aster),
·
"IDEAS" (with format IDEAS),
·
...

3.4 The echo of the data

Examples:
"You chose the method TRICK"
"For "the SMALLER" option, the frequencies are ignored.

These impressions must be avoided: they do nothing but repeat what the user wrote or what must to be known as in the Documentation of Use (Handbook U).
Data-processing handbook of Description
D2.07 booklet:
Index A

3.5
INFORMATION

It is information which relates to the course of calculation.

Examples:

·
iteration count to converge,
·
coefficient of conditioning of Lagranges,
·
criterion of convergence reached,…

It can be also "data-processing" information more:

·
cut memory (or disc) of a matrix
·
time spent in the factorization of a matrix,
·
...

It can be finally information intended to reassure the user (confirmation of its data):

·
a number of nodes, meshs,
·
list nodes likely to come into contact.

The writing of INFORMATION is made by of WRITE on the file "message" it is necessary to comply with the rules of [§6].

Information of the INFORMATION type is associated a **level** (1 or 2): INFORMATION of level 1 is

more
important that INFORMATION of level 2. The INFORMATION of level 1 will be described systematically in the Handbook
of Use (cf [D6.04.02 §3]. They are contractual.
When the user asks for INFORMATION: 1 (defect), one prints the INFORMATION of level 1.
Lorqu' it requires INFORMATION: 2, one prints the INFORMATION of level 1 and level 2.

4
Read/write in a file different from “ERROR” or “MESSAGE”

Orders making of the readings/writing in a file different from the files “ERROR” or “MESSAGE” [§5] are in a limited number.

·
If an order reads a file, one will try to give him a name of form LIRE_XXX: LIRE_MAILLAGE, LIRE_FONCTION,…,
·
if an order writes in a file, one will try to give him a name of form IMPR_XXX: IMPR_RESU, IMPR_COURBE,…,
·
orders of interface of entry: PRE_IDEAS, PRE_GIBI,… read in general on a file and write in another.

R1:
The orders should not read and write in files except if they are conceived for that (cf R2). The files should not be used to be exchanged information between orders (the SD are there for that) nor to make “pagination” memory (it is JEVEUX which takes care some).
R2:
The orders which read/write in files do it always explicitly.
I.e. they use standard key word:
/FILE: when the file is named
/UNITE_XXX: when one uses the logical unit of the file
except the orders LIRE_XXX (which are **operators**) other orders who read and write in files are **all** of **procedures** (IMPR_XXX, PRE_XXX,…) what wants to say that their **role** is to read and/or write.

5
Emission of alarm or error message

R3:
All order can transmit an error message (more or less immediate stop)
or of **alarm** (one continues the execution). For that, it will use them **exclusively**
routines UTMESS, UTDEBM,… [D6.04.01].
The transmitted messages will go then automatically in “the good” files
predetermined: “ERROR”, “MESSAGE” and “RESULT”.

R4:
Only the supervisor can make UTMESS <I>, the orders do not make a UTMESS
<I> (information). The messages of “information” are printed with of WRITE (cf.
[§6]).

6
Writing of messages of information, key word INFORMATION

In this paragraph, one calls INFORMATION, a message of information.

R5:
INFORMATION is always written by WRITE. The logical unit (of the file “MESSAGE”)
is always recovered by routine INFNIV.

R6:
INFORMATION has a level: 1 or 2. The INFORMATION of level 1 are **contractual**; they
are described in the Documentation of Use. The person in charge for the order
must present in meeting EDA any evolution of the INFORMATION of level 1.

R7:
The level of impression chosen by the user (1 or 2) is always via the key word
INFORMATION: /1 /2. Value 1 is always the default value.
When the user asks for INFORMATION: 2, that wants to say that it wishes the INFORMATION of
level 1 **and** of level 2.

R8:
Treatment of key word IMPR in an order OPOOXX:
CALL INFMAJ

R9:
Impression of INFORMATION of level 1:
CALL INFNIV (INF, NIV)! recovery of the logical unit and
! level of impression asked
IF (NIV.GE.1) WRITE (INF, FMT)…

R10:
Impression of INFORMATION of level 2:

```
CALL INFNN (INF, NIV)
IF (NIV.EQ.2) WRITE (INF, FMT)…
```

Intentionally white left page.

Mechanical department and Digital Models
Index:
With
Diffusion:
Users - Developers

Data-processing handbook of Description
D4.02 booklet:
D4.02.01 document

Structures of data listr8 and listis

Summary:

One describes the structures of data here:
listr8: list realities created by DEFI_LIST_REEL [U4.21.04]
listis: list entireties created by DEFI_LIST_ENTI [U4.21.05]

EDF
Direction of the Studies and Research
Electricity of France
Project Codes of Mechanics

---

1 Structures of data listr8 and listis in “2 words”

The structure of data listr8 contains a list of realities.

The structure of data listis contains a list of entireties.

2

Tree structure

Listr8
(K19)
:: =
record

".BINT"
:
S
V
R8

".LPAS"
:
S
V
R8

".NBPA"
:
S
V
I

".VALE"
:
S
V
R8

$VIDE
:
TITRATE
LISTIS
(K19)
:: =
record

".BINT"
:
S
V
I

".LPAS"
:

S
V
I

“.NBPA”
:
S
V
I

“.VALE”
:
S
V
I

$VIDE
:
TITRATE
3
Contents of the objects
We detail the contents of the objects of the structure of data listr8, those of the structure of data listis are identical in all points to the only difference of objects “.BINT”” .LPAS " and “.VALE” which contain entireties in the place of realities.
The structure can seem complicated to store a list of realities. Object “.VALE” would be enough. It indeed the list contains. The structure is conceived to benefit owing to the fact that the numbers of list can be regularly spaced: list “with constant step” per pieces. In this case, some algorithms use this concept of constant step.
The description of the list in fact is doubled:
·
“.VALE”: the list of the values contains,
·
“.BINT”,” .LPAS " and “.NBPA”: information equivalent to this list contains.

Page: 3/4
If the list can be represented graphically by:

1
1
2
2
2
3
3
a0
a1
has
has
has
2
has
3
a4
a5
6
7
“.VALE”
:
S
V
R8
dim =
8
v (1)
=
a0
v (2)
=
a1
...
v (8)
=
a7
“.BINT”
:
S
V

R8
dim =
4
v (1)
=
a0
v (2)
=
a2
v (3)
=
a5
v (4)
=
a7
This object contains the ends of the zones where the step is constant.
“.LPAS”
:
S
V
R8
dim =
3
v (1)
=
1
value of the 1st step
v (2)
=
2
value of the 2nd step
v (3)
=
3
value of the 3rd step
“.NBPA”
:
S
V
I
dim =
3
v (1)
=
2
a number of intervals length 1
v (2)

=
3
a number of intervals length 2
v (3)
=
2
a number of intervals length 3

4 Examples

4.1 Command file

BEGINNING
();
%
listr8
=
DEFI_LIST_REEL
(
TITRATE
:
“this is a listr8”
BEGINNING
:
1.
INTERVAL
:
(JUSQU_A: 5. NOT: 2.)
INTERVAL
:
(JUSQU_A: 7. NUMBERS: 2)
);
%
listis
=
DEFI_LIST_ENTI
(
TITRATE
:
“this is a listis”
VALE
:
(1,3,5,6,7)
);
%
IMPR_CO
(CO: (listr8, listis)
ATTRIBUTE: “NOT”

BASE: “G” IMPR: 1);

END ();

4.2
Contents of the objects

====> IMPR_CO OF THE STRUCTURE OF DATA: LISTR8????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 5
==
======
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--

SEGMENT IMPRESSION OF VALUES >LISTR8
.BINT
<

1 - 1.00000E+00 5.00000E+00 7.00000E+00

SEGMENT IMPRESSION OF VALUES >LISTR8
.LPAS
<
1 - 2.00000E+00 1.00000E+00

SEGMENT IMPRESSION OF VALUES >LISTR8
.NBPA
<
1 -
2
2

SEGMENT IMPRESSION OF VALUES >LISTR8
.TITR
<
1 - > this is a listr8
<

SEGMENT IMPRESSION OF VALUES >LISTR8
.VALE
<
1 - 1.00000E+00 3.00000E+00 5.00000E+00 6.00000E+00 7.00000E+00
====> FINE IMPR_CO OF STRUCTURE OF DATA: LISTR8?????????????????
====> IMPR_CO OF THE STRUCTURE OF DATA: LISTIS?????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 5
==
======
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >LISTIS
.BINT
<

1 -
1
3
5
6
7

SEGMENT IMPRESSION OF VALUES >LISTIS
.LPAS
<

1 -
2
2
1
1

SEGMENT IMPRESSION OF VALUES >LISTIS
.NBPA
<

1 -
1
1
1
1

SEGMENT IMPRESSION OF VALUES >LISTIS
.TITR
<
1 - > this is a listis
<

SEGMENT IMPRESSION OF VALUES >LISTIS
.VALE
<

1 -
1
3
5
6
7

Titrate:
Structure of Data function, fonction_C and fonction_I
Date:
03/06/02
Author (S):
***COURTEOUS Mr.**, J. Key PELLET*
:
D4.02.02-B Page
: 1/12

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D4.02 booklet:
D4.02.02 document

Structures of Data function, fonction_C and fonction_I

1 General information

The types of concepts function, fonction_C and fonction_I represent functions (with the direction mathematics of the term) of one, two (or more) variable.

The function have actual values, the fonction_C complex values and the fonction_I whole values. In the continuation of this document, one will speak only about the functions with actual values (function) by knowing that all can be transposed to the fonction_C and fonction_I.

A function can be:

· "tabulée": i.e. that it is known only in certain points. In this case, its evaluation

can require an interpolation or an extrapolation,
· "interpreted": its representation then contains the mathematical "formula" of the function. (One also speak about function "formulates").

Note:

An "interpreted" function (of a variable) can be tabulée (for certain values of its variable) by order CALC_FONC_INTERP. For such a tabulée function, the evaluation in a point different from the points of tabulation can be done by interpolation or while calculating exactly the function formulates having been used for the tabulation (object .FITA).
A formula can have variables as many as necessary. On the other hand, functions tabulées can have only 0, 1 or 2 variables. One will speak then about "constant function", "fonction_1" or "tablecloth".
The variables of a function (as its result) "are typified": "TEMP", "INST", "EPSI", … One will speak then about the name of the variables (or parameters) and the result.

---

2 Tree structure

FUNCTION (K19):: = record

F
"$VIDE":
TITRATE

".PROL": OJB
S V K16

| % if the function is interprêtée:
“.ADVA”:
OJB
S
V
I
“.INFX”:
OJB
S
V
I
“.NOVA”:
OJB
S
V
K8
“.POLO”:
OJB
S
V
I

| % if the function is tabulée:

/% if constant function or fonction_1:

“.VALE”
: OJB
S V R

% if tabulée starting from a function formulates:
“.FITA”:
OJB
S

V
K24

/% if tablecloth:

".PARA":
OJB
S

V
R
".VALE":
OJB
XC
V
R

3
Contents of the basic objects

3.1 Object
".PROL"

Object ".PROL" is length 5 for the tabulées functions of 0 or 1 variable and the functions interpreted. It is length 6 + 2 nf for the tablecloths, if nf is the number of fonction_1 composing the tablecloth.*

· .PROL (1):

Type of the function.

/"CONSTANT"
: constant function
/"FUNCTION"
: fonction_1 real

/“FONCT_C”
: fonction_1 complex
/“TABLECLOTH”
: function with 2 variables (tablecloth)
/“INTERPRE”
: interpreted function

· .PROL (2):

Type of interpolation wished between the points of tabulation. Relate to only the functions tabulées.

/“INTERPRE”:
one uses the exact formula of the function formulates having given rise to the function tabulée.
/“XXX FFF” where XXX and FFF = “NOT” or “FLAX” or “LOG”.

XXX relates to the parameter and FFF the function.

“NOT”: interpolation is prohibited,
“FLAX”: linear interpolation,
“LOG”: interpolation logarithmic curve.

That is to say a function F (X).

One will make a linear interpolation between the 2 points framing the sought point, but for this linear interpolation, one will use possibly the logarithm of X or F.

For example if “FLAX LOG”, one will use X and log (F).

Note:

For a tablecloth, XXX relate to the 2nd parameter of the function.

· .PROL (3):

Name of the last parameter.

/“TOUTPARA” for a constant function,
/name of the only parameter (i.e of the variable) for a fonction_1,
/name of the 2nd parameter for a tablecloth,
/'' for a not tabulée interpreted function.

· .PROL (4):

/name (or type) of the result of the function,
/“TOUTRESU”.

· .PROL (5):

“prolongation” desired with the function apart from its field of tabulation (extrapolation).
concern that the tabulées functions.

“xy”
where X and y = “E” or “It or “It or “I”.

X:
prolongation “on the left” (for a parameter lower than the smallest parameter of tabulation),
y:
prolongation “on the right” (for a parameter higher than the greatest parameter of tabulation).

“E”: excluded prolongation,
“It: constant prolongation,
“It: linear prolongation (starting from the 2 first or of the last 2 points),
“I”: interpreted prolongation (starting from the formula).

Note:

For a tablecloth, these prolongations relate to the 2nd parameter (values of the object “.PARA”).
There is not prolongation “logarithmic curve”.

· .PROL (6):

For the tablecloths: name of the 1st parameter (name of the parameter of the fonction_1 composing the tablecloth).

· .PROL (7):

Type of interpolation wanted for the 1st fonction_1 composing the tablecloth

("FLAX LOG", "LOG LOG",…) (see .PROL (2)).

· .PROL (8):

Type of prolongation wanted for the 1st fonction_1 composing the tablecloth

("EL", "DC",…) (see .PROL (5)).

· .PROL (9):

Type of interpolation wanted for the 2nd fonction_1 composing the tablecloth

· .PROL (10)…

3.2 Object
".PARA"

This object contains the values of the "2nd" variable of the tablecloth. With each value of this 2nd variable corresponds an object of the collection ".VALE" which contains the values of the fonction_1 associated this variable.

3.3 Object
".VALE"

1) For a fonction_1, this object contains the numerical values of the points of tabulation.
That is to say N: the number of points of tabulation,

V (1,…, N)
:
X-coordinates

points,
*V (n+1,…, 2*n)*
:
values of the function at the points.

Note:

If the function is with complex values, storage is as follows:

V (n+1): real part of the function at the 1st point,

V (n+2): imaginary part of the function at the 1st point,

V (n+3): real part of the function at the 2nd point,

V (n+4): imaginary part of the function at the 2nd point,
...

*V (3*n): imaginary part of the function at the last point.*

The number of points of discretization (N) can be obtained by division by 2 (or 3) of the attribute “LONMAX” of object” .VALE ".

2) For a tablecloth, this object is a contiguous numbered collection. Each object I of collection with the same structure as object “.VALE” of the fonction_1 (above). It describes the attached function with the ième value of the 2nd variable of the tablecloth.

3.4 Object
“.FITA”

When the fonction_1 was obtained by CALC_FONC_INTERP, this object contains the name of the function *"formula" which was used for calculation of the values of the tabulée function. The name of the function formulates initial is necessary if the interpolation for the tabulée function is worth: "INT (erprete)".*

3.5 Object ".NOVA"

This object contains the name of the variables of the function "formulates".

3.6 Objects ".ADVA"," .INFX ", ".POLO"

These objects contain "pointers" towards objects "supervisor" ("&&SYS FI.ARITE", '&&SYS FI.CLASSE, '&&SYS FI.NOM,…) who are not currently documented. Us them will thus not comment on here.

Data-processing handbook of Description
D4.02 booklet:
HT-66/03/007/A

---

4 Examples

4.1
Constant function: f1

f1=defi_constante (vale: 1.2 nom_resu: "nom_res1");
impr_co (Co: f1);

====> IMPR_CO OF THE STRUCTURE OF DATA: F1
--
SEGMENT IMPRESSION OF VALUES >F1 .PROL <
1 - >CONSTANT<>LIN FLAX <>TOUTPARA<>NOM_RES1<>CC <
--
SEGMENT IMPRESSION OF VALUES >F1 .TITR <
1 - >ASTER 3.07.02 F1 CONCEPT CALCULATES the 20/08/96 A 09:36: 57 OF FUNCTION TYPE <
--
SEGMENT IMPRESSION OF VALUES >F1 .VALE <
1 - 1.00000E+00 1.20000E+00

4.2
Tabulée real function: f2

lpara=defi_list_reel (beginning: 3. interval: (jusqu_a: 6. a number: 3));
lfonc=defi_list_reel (beginning: 3.2 interval: (jusqu_a: 6.2 a number: 3));
f2=defi_fonction (
titrate: "this is my title"
Interpol: "not" will nom_para: "DX" nom_resu: "mom_res2"
prol_gauche: "excluded" prol_droit: "constant"

will vale_para: vale_fonc will lpara: lfonc
);
impr_co (Co: f2);

====> IMPR_CO OF THE STRUCTURE OF DATA: F2
--
SEGMENT IMPRESSION OF VALUES >F2 .PROL <
1 - >FONCTION<>NON NOT <>DX <>MOM_RES2<>EC <
--
SEGMENT IMPRESSION OF VALUES >F2 .TITR <
1 - >ceci is my title <
--
SEGMENT IMPRESSION OF VALUES >F2 .VALE <
1 - 3.00000E+00 4.00000E+00 5.00000E+00 6.00000E+00 3.20000E+00
6 - 4.20000E+00 5.20000E+00 6.20000E+00

4.3
Tabulée complex function: f3

f3=defi_fonction (
Interpol: (“flax”, “log”) will nom_para: “INST” nom_resu: “mom_res3”
prol_gauche: “linear” prol_droit: “constant”
vale_c: (0. 1.2.2.2 1. 3.7.4.7 2. 5.6.6.6 3. 3.5.4.5)
);
impr_co (Co: f3);

```
====> IMPR_CO OF THE STRUCTURE OF DATA: F3
-----------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >F3 .PROL <
1 - >FONCT_C <>LIN LOG <>INST <>MOM_RES3<>LC <
-----------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >F3 .TITR <
1 - >ASTER 3.07.02 F3 CONCEPT CALCULATES the 20/08/96 A 09:36: 57 OF TYPE
FONCTION_C <
-----------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >F3 .VALE <
1 - 0.00000E+00 1.00000E+00 2.00000E+00 3.00000E+00 1.20000E+00
6 - 2.20000E+00 3.70000E+00 4.70000E+00 5.60000E+00 6.60000E+00
11 - 3.50000E+00 4.50000E+00
```

4.4
Real tablecloth defined by functions: nap1

```
f21=defi_fonction (nom_resu: "bid1" will nom_para: "temp"
Interpol: ("flax", "flax") prol_droit: "linear"
vale: (1.2 3.7.4.2.6.7));
f22=defi_fonction (nom_resu: "bid2" will nom_para: "inst"
Interpol: ("log", "log") prol_droit: "constant"
vale: (10.2 30.7 40.2 60.7));
f23=defi_fonction (nom_resu: "bid2" will nom_para: "inst"
Interpol: ("log", "flax") prol_gauche: "linear"
vale: (11.2 31.7 41.2 61.7));
nap1=defi_nappe (
Interpol: ("flax", "log") will nom_para: "PULS" nom_resu: "mom_nap1"
prol_gauche: "excluded" prol_droit: "constant"
para: (8.9 12.9 17.9)
function: (f21 f22 f23)
);
impr_co (Co: nap1);
```

```
====> IMPR_CO OF THE STRUCTURE OF DATA: NAP1
-----------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >NAP1 .PARA <
1 - 8.90000E+00 1.29000E+01 1.79000E+01
-----------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >NAP1 .PROL <
1 - >NAPPE <>LIN LOG <>PULS <>MOM_NAP1<>EC <>TEMP <>LIN FLAX <
8 - >EL <>LOG LOG <>EC <>LOG FLAX <>LE <
-----------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >NAP1 .TITR <
```

1 - >ASTER 3.07.02 CONCEPT NAP1 CALCULATES THE 20/08/96 A 09:36: 58 OF FUNCTION TYPE <

IMPRESSION OF THE COLLECTION: NAP1 .VALE
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NAP1 .VALE< OC: 1
1 - 1.20000E+00 4.20000E+00 3.70000E+00 6.70000E+00
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NAP1 .VALE< OC: 2
1 - 1.02000E+01 4.02000E+01 3.07000E+01 6.07000E+01
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NAP1 .VALE< OC: 3
1 - 1.12000E+01 4.12000E+01 3.17000E+01 6.17000E+01
Data-processing handbook of Description
D4.02 booklet:
HT-66/03/007/A

Code_Aster ®
Version
6.4

Titrate:
Structure of Data function, fonction_C and fonction_I
Date:
03/06/02
Author (S):
COURTEOUS Mr., J. Key PELLET
:
D4.02.02-B Page
: 8/12

4.5
Definite real tablecloth completion: nap2

nap2=defi_nappe (
Interpol: (“flax”, “log”) will nom_para: “PULS” nom_resu: “mom_nap2”
prol_gauche: “excluded” prol_droit: “constant”
para: (8.9 12.9)
nom_para_fonc: “EPSI”
defi_fonction: (prol_droit: “linear”
vale: (1.2 3.5.2.2.4.5 3.2.6.5))
defi_fonction: (prol_gauche: “constant”
vale: (1.2 3.7.4.2.6.7))
);
impr_co (Co: nap2);

====> IMPR_CO OF THE STRUCTURE OF DATA: NAP2
--
SEGMENT IMPRESSION OF VALUES >NAP2 .PARA <
1 - 8.90000E+00 1.29000E+01
--
SEGMENT IMPRESSION OF VALUES >NAP2 .PROL <
1 - >NAPPE <>LIN LOG <>PULS <>MOM_NAP2<>EC <>EPSI <>LIN FLAX <
8 - >EL <>LIN FLAX <>CE <
--
SEGMENT IMPRESSION OF VALUES >NAP2 .TITR <
1 - >ASTER 3.07.02 CONCEPT NAP2 CALCULATES THE 20/08/96 A 09:36: 58 OF FUNCTION TYPE <
--
IMPRESSION OF THE COLLECTION: NAP2 .VALE
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NAP2 .VALE< OC: 1
1 - 1.20000E+00 2.20000E+00 3.20000E+00 3.50000E+00 4.50000E+00
6 - 6.50000E+00
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NAP2 .VALE< OC: 2
1 - 1.20000E+00 4.20000E+00 3.70000E+00 6.70000E+00

4.6
Function formulates to 1 variable: ff1

```
! formulate (real: (ff1 (real: inst) =
2.3* (cos (3.2+sqrt (inst))) - heavysid (inst - pi)
));
impr_co (Co: ff1);
```

====> IMPR_CO OF THE STRUCTURE OF DATA: FF1
--
SEGMENT IMPRESSION OF VALUES >FF1 .ADVA <
1 - 53 0 0 0 0
6 - 0 0 0 0 0
--
SEGMENT IMPRESSION OF VALUES >FF1 .INFX <
1 - 54 26 23 12 23
6 - 55 24 7 23 53
11 - 40 40 40 25 21
16 - 23 53 25 43 40
21 - 42 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0

46 - 0 0 0 0 0
--
SEGMENT IMPRESSION OF VALUES >FF1 .NOVA <
1 - >INST <> <> <> <> <> <> <
8 - > <> <> <
--
SEGMENT IMPRESSION OF VALUES >FF1 .POLO <
1 - 54 55 53 7 24
6 - 12 26 53 43 25
11 - 21 25 40 25 21
16 - 23 53 25 43 40
21 - 42 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0
46 - 0 0 0 0 0
--
SEGMENT IMPRESSION OF VALUES >FF1 .PROL <
1 - >INTERPRE<>INTERPRE<> <>TOUTRESU<>II <

4.7
Function formulates with 2 variables: ff2

! formulate (real: (ff2 (real: X, reality: y) =

```
2.3* (cos (3.2+sqrt (X))) - (ff1 (y) +3.4*f2 (X))
));
impr_co (Co: ff2);

====> IMPR_CO OF THE STRUCTURE OF DATA: FF2
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF2 .ADVA <
1 - 57 58 0 0 0
6 - 0 0 0 0 0
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF2 .INFX <
1 - 54 26 23 12 23
6 - 55 24 7 23 57
11 - 40 40 40 25 23
16 - 20 52 23 58 40
21 - 24 59 26 22 60
26 - 23 57 40 40 42
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0
46 - 0 0 0 0 0
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF2 .NOVA <
1 - >X <>Y <> <> <> <> <> <
8 - > <> <> <
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF2 .POLO <
1 - 54 55 57 7 24
6 - 12 26 58 52 20
11 - 59 57 60 22 26
16 - 24 25 23 58 40
21 - 24 59 26 22 60
26 - 23 57 40 40 42
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0
46 - 0 0 0 0 0
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF2 .PROL <
1 - >INTERPRE<>INTERPRE<> <>TOUTRESU<>II <
```

4.8
Function formulates to 1 variable: ff3 tabulée thereafter

! formulate (real: (ff3 (real: X) =sqrt (X)));
&ff3=calc_fonc_interp (
function: ff3
Interpol: "int" prol_droit: "linear"
vale_r: (1. 4. 9. 16.)
);
impr_co (Co: ff3);

====> IMPR_CO OF THE STRUCTURE OF DATA: FF3
--
SEGMENT IMPRESSION OF VALUES >FF3 .ADVA <
1 - 62 0 0 0 0
6 - 0 0 0 0 0
--
SEGMENT IMPRESSION OF VALUES >FF3 .FITA <
1 - >FF3 <
--
SEGMENT IMPRESSION OF VALUES >FF3 .INFX <
1 - 7 23 62 40 42
6 - 0 0 0 0 0
11 - 0 0 0 0 0
16 - 0 0 0 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0 0 0

41 - 0 0 0 0 0
46 - 0 0 0 0 0

SEGMENT IMPRESSION OF VALUES >FF3 .NOVA <
1 - >X <> <> <> <> <> <> <
8 - > <> <> <

SEGMENT IMPRESSION OF VALUES >FF3 .POLO <
1 - 62 7 62 40 42
6 - 0 0 0 0 0
11 - 0 0 0 0 0
16 - 0 0 0 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0
46 - 0 0 0 0 0

SEGMENT IMPRESSION OF VALUES >FF3 .PROL <
1 - >FONCTION<>INT INT <>X <>TOUTRESU<>EL <

SEGMENT IMPRESSION OF VALUES >FF3 .TITR <
1 - >ASTER 3.07.02 CONCEPT FF3 CALCULATES THE 20/08/96 A 09:36: 58 OF FUNCTION TYPE <

SEGMENT IMPRESSION OF VALUES >FF3 .VALE <
1 - 1.00000E+00 4.00000E+00 9.00000E+00 1.60000E+01 1.00000E+00
6 - 2.00000E+00 3.00000E+00 4.00000E+00

4.9
Function tabulée starting from an other function

```
! formulate (real: (ff41 (real: X) =sqrt (X)));
! formulate (real: (ff42 (real: X) =2.*ff41 (X) *sqrt (X)));
ff4=calc_fonc_interp (
function: ff42 nom_resu: “nom_res4”
Interpol: (“log”, “flax”)
prol_droit: “excluded” prol_gauche: “interpre”
vale_r: (1.6 2.6.3.6.4.6)
);
impr_co (Co: ff4);
```

```
====> IMPR_CO OF THE STRUCTURE OF DATA: FF4
---------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF4 .FITA <
1 - >FF42 <
---------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF4 .PROL <
1 - >FONCTION<>LOG FLAX <>X <>NOM_RES4<>IE <> <
---------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF4 .TITR <
1 - >ASTER 3.07.02 CONCEPT FF4 CALCULATES THE 20/08/96 A 09:36: 58 OF FUNCTION
TYPE <
---------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >FF4 .VALE <
1 - 1.60000E+00 2.60000E+00 3.60000E+00 4.60000E+00 3.20000E+00
6 - 5.20000E+00 7.20000E+00 9.20000E+00
```

Intentionally white left page.

Key:
D4.02.03
Page: 1/18
Mechanical department and Digital Models
Index:
With
Diffusion:
Users - Developers
Data-processing handbook of Description
D4.02 booklet:
D4.02.03 document
Structures of Data curved and surface

Summary:
This document described:

- *the structure of data curves produced by operator INTE_MAIL_2D. A curve is is a whole of meshs segment, is a meeting of segments of right-hand side and/or arcs of ring,*
- *the structure of data surfaces produced by operator INTE_MAIL_3D. Currently, one object of the surface type can contain only segments of right-hand side among the meshs of one grid 3D.*

EDF
Direction of the Studies and Research
Electricity of France
Project Codes of Mechanics

Code_Aster ®

Version
4
Titrate:
Structures of Data curved and surface
Date:
28/01/1999
Author (S):
X. DESROCHES
Key:
D4.02.03
Page: 2/18
Contents

1
SD Courbe: general

Object of a curved type follows a curve on a geometry 2D. This curve is one of both following types:

- *a meeting of segments of right-hand side and/or arcs of circle,*
- *a whole of meshs SEG2 or preexistent SEG3.*

This concept is produced by operator INTE_MAIL_2D.

2
Relations between the curved SD and the other SD

No if it is not that a curve is located compared to a grid.

3
Tree structure of the curved SD

curve (K8):: = record
“.NOMMAIL”: S.E.K8
“.TYPCOURBE”: S.E.K8/“LISTMAIL”: courbe_LM
/“SGTDARCC”: courbe_SA
courbe_LM:: = record
“.CHEMIN”: XC V I numbered
“.MAIL1”: XC V I numbered
“.MAIL2”: XC V I numbered
courbe_SA:: = record
“.XYASGT”:

S V R8
“.XYBSGT”:
S V R8
“.XYCARC”:
S V R8
“.XSARC”:
S V R8
“.XRARC”:
S V R8
“.EXSGT”:
XC V R8
“.ORSGT”:
XC V R8
“.MAIL1”:
XC V I
“.MAIL2”:
XC V I
“.CNXEX”:
XC V I
“.CNXOR”:
XC V I
“.FACEX”:
XC V I
“.FACOR”:
XC V I
“.PAREX”:
XC V R8
“.PAROR”:
XC V R8
Data-processing handbook of Description
D4.02 booklet:

Index A

---

4 Contents of the objects of the curved SD

4.1. Common articles

".NOMMAIL": S.E.K8: the name of the concept of the grid type contains

4.2. SD courbe_LM

Collections WAY, MAIL1 and MAIL2 have the same number of objects of collection. This number is the number of disjoined ways subjacent with the list of meshs obtained starting from the operands of key word factor DEFI_CHEMIN.

Structure of the objects of collection:

One is interested in Ième OC and one notes:

CHM = WAY (I)

M1 = MAIL1 (I)

M2 = MAIL2 (I)

If CHM consists of NR meshs 1-D, then:

length (CHM) = NR + 1

CHM (J), J = 1,…, NR gives the numbers of the meshs 1D describing the way

CHM (N+1) {0, CHM (1)}

If CHM (N+1) = 0

then the way is simple

if not the way is cyclic

By convention: length (M1) = long (m2) = long (CHM)

As follows:

- M1 (NR + 1) = m2 (NR + 1) = 0
- M1 (J), J = 1,…, NR gives the number of the first mesh 2D which admits the mesh 1D CHM (J) for face. Thus M1 (J) _ 0 for J = 1,…, NR
- If m2 (J) = 0, then the mesh 1D CHM (J) is face of only one mesh 2D, if not m2 (J) the number of the 2nd mesh 2D admitting contains the mesh 1D CHM (J) for face.

Example 1:

CHM:
37
11
23
0
39
11
M1:
48
21
63
0
23
37
21
48
63
M2:
0
39
0
0

Example 2:
13
24
CHM:
3
24

13
9
3
151
169
9
M1:
169.169.151 137
0
137
3
180
121
M2:
180
0
0
151
0

4.3

SD courbe_SA

4.3.1

Substructure of description of the segments and arcs brought into play

One notes:

·

Nb_sgt the number of segments of right-hand side,

·

Nb_arc the number of arcs of circle.

XYASGT S V R8:

contains the co-ordinates of the points origin of the segments of right-hand sides

XYBSGT S V R8:

contains the co-ordinates of the points end of the segments of right-hand sides

length (XYASGT) = long (XYBSGT) = 2 * (Nb_sgt + 1)

fictitious co-ordinates

2 co-ordinates

XYASGT:

0 0 *x1* 1

2

2 ....

With yA teststemxà yA

1

XYBSGT:

0 0

y1

2 ....

B x2
xB
B yB
One represents the absence of segments in the curve by a vector XYASGT (and thus XYBSGT) of length 2 initialized to 0.
If there is at least a segment, then the co-ordinates of the point origin Have of Ième segment and of not end Bi of Ième segment are:
I
XYASGT (2*I+1) <-- X With
I
XYASGT (2*I+2) <-- yA
idem for B (with XYBSGT)
XYCARC, XSARC, XRARC: S V R8
Contain, respectively, the co-ordinates of the centers, terminals of the angular sectors and value rays.

---

length (XYCARC) = long (XSARC) = 2* (Nb-arc + 1)
length (XRARC) = Nb_arc + 1
If no arc is used at the time of the call, then:
length (XYCARC) = long (XSARC) = 2
length (XRARC) = 1
and the 3 vectors are initialized to 0.
If not:
I
XYCARC (2*I+1) <-- xC
I
XYCARC (2*I+2) <-- yC
I
XRARC (I+1) <-- R

I
XSARC (2*I+1) <-- inf
I
XSARC (2*I+2) <-- sup

4.3.2

Substructure of location on the curves

Collections ORSGT and EXSGT

The curve (segment or arc) is parameterized according to:

X (*S*)
C (*has*, *B*
=

)
{*M* (*S*);
where
S
[*has*, *B*]}
M (*S*) *y* (*S*)
then:
I = *NR*
C = (
C sor, *sex*
I
I
)
I =1

where *NR* is the number of meshs intersected by the curve:

C (*sor sex*
however
ex
,
=

I
)
{*M* (*S*) *C*; *S* [*S*, *S*
I
I
I
]}
Then
length (ORSGT) = long (EXSGT) = NR
however

S
ORSGT (I) <--
I
ex
S
EXSGT (I) <--
I

4.3.3
Substructure of location in
Collections MAIL1, MAIL2, FACOR, FACEX, PAROR, PAREX

however
ex
If *C* (*S*, *S*
I
I) is the contribution of a *Ki* mesh to the intersection *C*, then:
K C {*M* (*sor*), *M* (*sex*

I
I
)}

---

however
ex
1
The 2 points *M* (*if*) and *M* (*if*) can be located in the *Ki* mesh by the data of the faces
containing each point and by the curvilinear X-coordinate (variable between 0. and 1.) on the faces.
2
however
ex

Moreover, one second *Ki* mesh can give *C* (*S*, *S*
I
I).

Example:

N3
Face=3
Face=2
M sor
(*I*)
Piece I
M sex
(*I*
ror
)
I
rex
K1
I
N1
Face=1
N2
MAIL1 (I)
<--
number of the *K1* mesh
MAIL2 (I)
<--
0 if piece I is obtained only for the mesh 2D *K1*
K 2 if *K2* is the 2nd mesh 2D giving piece I
FACOR (I)
<--
3 (as a face of *K1*)
FACEX (I)
<--
2 (as a face of *K1*)
PAROR (I)
<--
however
R

I
PAREX (I)
<--
ex
R

I

4.3.4
Substructure of connexity
Collections CNXOR and CNXEX

A OC of collections CNXOR and CNXEX is a vector of entireties dimensioned with the numbers of related components of *C*.

For the curve *C* corresponding to the OC, if *C* is composed of *NR* elementary pieces, then:
the related component number I of *C* is consisted of the meeting of the new numbers:
CNXOR (I), CNXOR (I) +1,…, CNXEX (I)

4.3.5
Length of the collections and objects of collection of courbe_SA

ORSGT
EXSGT
PAROR
PAREX
Nb_OC = Nb_sgt + Nb_arc
FACOR

FAX
length of a OC: unknown factor a priori
MAIL1
but all identical
MAIL2

CNXOR
Nb_OC = Nb_sgt + Nb_arc
CNXEX

length of a OC: unknown factor a priori
but all identical

5

Examples of curved SD

```
BEGINNING ();
m = LIRE_MAILLAGE ();
%
%
% CREATION Of a CURVE OF the SEGMENT TYPE
%
segment=INTE_MAIL_2D (GRID: m
DEFI_SEGMENT: (ORIGIN: (0. , 0.)
END: (10. , 0.)));
IMPR_CO (CO: segment);
%
%
% CREATION Of a CURVE OF the WAY TYPE (LIST OF MESHS)
%
chemin=INTE_MAIL_2D (GRID: m
DEFI_CHEMIN: (GROUP_MA: GRMA2));
IMPR_CO (CO: way);
%
%
% CREATION Of a CURVE OF the ARC TYPE
%
arc =INTE_MAIL_2D (GRID: m
DEFI_ARC: (CENTER: (0. 0.) RAY: 1. SECTOR: (0. 90.)));
IMPR_CO (CO: arc);
end ();
```

====> IMPR_CO OF THE STRUCTURE OF DATA: SEGMENT????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 17
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

IMPRESSION OF THE COLLECTION: SEGMENT .CNXEX
OBJECT IMPRESSION OF COLLECTION >SEGMENT .CNXEX < OC: 1
1 - 14

IMPRESSION OF THE COLLECTION: SEGMENT .CNXOR
OBJECT IMPRESSION OF COLLECTION >SEGMENT .CNXOR < OC: 1
1 - 1

IMPRESSION OF THE COLLECTION: SEGMENT .EXSGT
OBJECT IMPRESSION OF COLLECTION >SEGMENT .EXSGT < OC: 1
1 - 1.90901E-02 5.39950E-02 9.91951E-02 1.52721E-01 2.13434E-01
6 - 2.80566E-01 3.53553E-01 4.31959E-01 5.15432E-01 6.03682E-01
11 - 6.96461E-01 7.93560E-01 8.94794E-01 1.00000E+00

IMPRESSION OF THE COLLECTION: SEGMENT .FACEX
OBJECT IMPRESSION OF COLLECTION >SEGMENT .FACEX < OC: 1
1 - 1 1 1 1 1
6 - 1 1 1 1 1
11 - 1 1 1 1

IMPRESSION OF THE COLLECTION: SEGMENT .FACOR
OBJECT IMPRESSION OF COLLECTION >SEGMENT .FACOR < OC: 1
1 - 1 1 1 1 1
6 - 1 1 1 1 1
11 - 1 1 1 1

IMPRESSION OF THE COLLECTION: SEGMENT .MAIL1
OBJECT IMPRESSION OF COLLECTION >SEGMENT .MAIL1 < OC: 1
1 - 1 9 17 25 33
6 - 41 49 57 65 73
11 - 81 89 97 105

IMPRESSION OF THE COLLECTION: SEGMENT .MAIL2
OBJECT IMPRESSION OF COLLECTION >SEGMENT .MAIL2 < OC: 1
1 - 113.121.129 137 145
6 - 153.161.169 177 185
11 - 193.201.209 217

SEGMENT IMPRESSION OF VALUES >SEGMENT .NOMMAIL <

1 - >M <

IMPRESSION OF THE COLLECTION: SEGMENT .ORSGT
OBJECT IMPRESSION OF COLLECTION >SEGMENT .ORSGT < OC: 1
1 - 0.00000E+00 1.90901E-02 5.39950E-02 9.91951E-02 1.52721E-01
6 - 2.13434E-01 2.80566E-01 3.53553E-01 4.31959E-01 5.15432E-01
11 - 6.03682E-01 6.96461E-01 7.93560E-01 8.94794E-01

IMPRESSION OF THE COLLECTION: SEGMENT .PAREX
OBJECT IMPRESSION OF COLLECTION >SEGMENT .PAREX < OC: 1
1 - 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00
6 - 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00
11 - 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00

IMPRESSION OF THE COLLECTION: SEGMENT .PAROR
OBJECT IMPRESSION OF COLLECTION >SEGMENT .PAROR < OC: 1
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >SEGMENT .TYPCOURBE <
1 - >SGTDARCC<

SEGMENT IMPRESSION OF VALUES >SEGMENT .XRARC <
1 - 0.00000E+00

SEGMENT IMPRESSION OF VALUES >SEGMENT .XSARC <
1 - 0.00000E+00 0.00000E+00

Data-processing handbook of Description
D4.02 booklet:

Index A

Code_Aster ®

Version
4
Titrate:
Structures of Data curved and surface
Date:
28/01/1999
Author (S):
X. DESROCHES
Key:
D4.02.03

Page: 10/18
SEGMENT IMPRESSION OF VALUES >SEGMENT .XYASGT <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >SEGMENT .XYBSGT <
1 - 0.00000E+00 0.00000E+00 1.00000E+01 0.00000E+00

SEGMENT IMPRESSION OF VALUES >SEGMENT .XYCARC <
1 - 0.00000E+00 0.00000E+00
====> FINE IMPR_CO OF STRUCTURE OF DATA: SEGMENT????????????????
====> IMPR_CO OF THE STRUCTURE OF DATA: WAY????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 5
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

IMPRESSION OF THE COLLECTION: WAY .CHEMIN
OBJECT IMPRESSION OF COLLECTION >CHEMIN .CHEMIN < OC: 1
1 - 483.482.481 480 516
6 - 517.518.520 0
OBJECT IMPRESSION OF COLLECTION >CHEMIN .CHEMIN < OC: 2
1 - 556.554.553 552 588
6 - 589.590.591 0

IMPRESSION OF THE COLLECTION: WAY .MAIL1
OBJECT IMPRESSION OF COLLECTION >CHEMIN .MAIL1 < OC: 1
1 - 112.110.108 106 218
6 - 220.222.224 0
OBJECT IMPRESSION OF COLLECTION >CHEMIN .MAIL1 < OC: 2
1 - 336.334.332 330 442
6 - 444.446.448 0

IMPRESSION OF THE COLLECTION: WAY .MAIL2
OBJECT IMPRESSION OF COLLECTION >CHEMIN .MAIL2 < OC: 1
1 - 0 0 0 0 0
6 - 0 0 0 0
OBJECT IMPRESSION OF COLLECTION >CHEMIN .MAIL2 < OC: 2
1 - 0 0 0 0 0
6 - 0 0 0 0

SEGMENT IMPRESSION OF VALUES >CHEMIN .NOMMAIL <
1 - >M <

SEGMENT IMPRESSION OF VALUES >CHEMIN .TYPCOURBE <
1 - >LISTMAIL<

====> FINE IMPR_CO OF STRUCTURE OF DATA: WAY????????????????
====> IMPR_CO OF THE STRUCTURE OF DATA: ARC????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 17
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

IMPRESSION OF THE COLLECTION: ARC .CNXEX
OBJECT IMPRESSION OF COLLECTION >ARC .CNXEX < OC: 1
1 - 10

IMPRESSION OF THE COLLECTION: ARC .CNXOR
OBJECT IMPRESSION OF COLLECTION >ARC .CNXOR < OC: 1
1 - 1

IMPRESSION OF THE COLLECTION: ARC .EXSGT
OBJECT IMPRESSION OF COLLECTION >ARC .EXSGT < OC: 1
1 - 1.26966E-01 2.52680E-01 2.68597E-01 5.23599E-01 6.56873E-01
6 - 8.48061E-01 1.00042E+00 1.34127E+00 1.37872E+00 1.57080E+00

IMPRESSION OF THE COLLECTION: ARC .FACEX
OBJECT IMPRESSION OF COLLECTION >ARC .FACEX < OC: 1
1 - 3 2 2 2 2
6 - 2 3 3 3 2

IMPRESSION OF THE COLLECTION: ARC .FACOR
OBJECT IMPRESSION OF COLLECTION >ARC .FACOR < OC: 1
1 - 1 1 1 3 1
6 - 3 1 1 2 1

Page: 11/18

IMPRESSION OF THE COLLECTION: ARC .MAIL1
OBJECT IMPRESSION OF COLLECTION >ARC .MAIL1 < OC: 1
1 - 25 18 19 20 21
6 - 22 23 16 15 8

IMPRESSION OF THE COLLECTION: ARC .MAIL2
OBJECT IMPRESSION OF COLLECTION >ARC .MAIL2 < OC: 1
1 - 0 0 0 0 0
6 - 0 0 0 0 0

SEGMENT IMPRESSION OF VALUES >ARC .NOMMAIL <
1 - >M <

IMPRESSION OF THE COLLECTION: ARC .ORSGT
OBJECT IMPRESSION OF COLLECTION >ARC .ORSGT < OC: 1
1 - 0.00000E+00 1.26966E-01 2.52680E-01 2.68597E-01 5.23599E-01
6 - 6.56873E-01 8.48061E-01 1.00042E+00 1.34127E+00 1.37872E+00

IMPRESSION OF THE COLLECTION: ARC .PAREX
OBJECT IMPRESSION OF COLLECTION >ARC .PAREX < OC: 1
1 - 4.93501E-01 5.24427E-02 6.15174E-02 2.78594E-01 4.42575E-01
6 - 7.31218E-01 6.33209E-01 1.04904E-01 7.35628E-02 1.00000E+00

IMPRESSION OF THE COLLECTION: ARC .PAROR
OBJECT IMPRESSION OF COLLECTION >ARC .PAROR < OC: 1
1 - 1.50369E-02 5.06499E-01 9.47557E-01 9.38483E-01 7.21406E-01
6 - 5.57425E-01 2.68782E-01 3.66791E-01 8.95096E-01 9.26437E-01

SEGMENT IMPRESSION OF VALUES >ARC .TYPCOURBE <
1 - >SGTDARCC<

SEGMENT IMPRESSION OF VALUES >ARC .XRARC <
1 - 0.00000E+00 1.00000E+00

SEGMENT IMPRESSION OF VALUES >ARC .XSARC <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 1.57080E+00

SEGMENT IMPRESSION OF VALUES >ARC .XYASGT <
1 - 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >ARC .XYBSGT <
1 - 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >ARC .XYCARC <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
====> FINE IMPR_CO OF STRUCTURE OF DATA: ARC????????????????
Data-processing handbook of Description
D4.02 booklet:

Index A

6
SD Surface: general

An object of the surface type contains segments of right-hand side among the meshs 3D of a grid.
This concept is produced by operator INTE_MAIL_3D.

7
Relations between the SD surfaces and the other SD

No if it is not that a surface is located compared to a grid.

8
Tree structure of the SD surfaces

Surface (K8):: = record
“.NOMA”: Indirect OJB (1) S.E.K24
(1): GRID
“.NSDS”: Indirect OJB (*) S V K24 DOCU (“SGT3”)
(*) (1: 13): SURFACE_1D
/* dimension: a number of segments: Nbseg *
/* NSDS (I): = nom_surface // ' K1S'//Codent (K4Segi) *
K8 (with more the 9999 segments)

K13
SURFACE_1D (K13):: = record
“.DESC”: OJB S V R LONG (6) DOCU (“SGT3”)
“.SGTEL”: REPERAGE_1D
$vide: REPERAGE_
“.CONEX.ORIG”: OJB S V I
“.CONEX.EXTR”: OJB S V I

REPERAGE_1D (K19):: = record
“.ORIG”: OJB S V R8
“.EXTR”: OJB S V R8
“.TYPE”: OJB S V I
REPERAGE_ (K13):: = record
“.MAIL”: OJB XC V I
“.FACE.ORIG”: OJB S V I
“.FACE.EXTR”: OJB S V I
“.CREFM.ORIG”: OJB S V R8
“.CREFM.EXTR”: OJB S V R8
“.ARETE.ORIG”: OJB S V I
“.ARETE.EXTR”: OJB S V I
“.CREFF.ORIG”: OJB S V R8
“.CREFF.EXTR”: OJB S V R8

---

9 Location of a segment in a grid 3D

One notes:

·
the field with a grid,
·
T H the whole of the meshs 3D,
·
K a mesh 3D,
·
K the border of *K*; *K* is a union of faces *F*. A face is a triangle or one quadrangle,
·
K the border of *F*; *K* is a union of rectilinear edges,

.
S = [*A*,]
B the segment to be located.
In fact, one seeks to locate *S* in
K
.
K T K

9.1

Location of a point on S

Line AB admits the parametric representation *AM* = *T AB*
T R.
The segment *S* corresponds to the interval *T*
[
0]
1
, .

9.2

Decomposition of *S*

S
1
2
is broken up into elementary segments *S* = {*A*, *A*
I
I
I} so that:
N
S =
S
I
U =1
The family (*If*)
is ordered with the direction:
I =1,…, *N*
J

AA
J
I
= *T AB*
I = 1,…, *N*
J =
I
1,2

with:
0
1
2
1
2
1
2
1
2
T < *T* *T* < *T* *T* < *T* *T* < *T1*
1
1
2
2
I
I
N
N

9.3

Location of an elementary segment in

That is to say *E* = {*K* *T*; *S* *K*
I
H
I
}
If is located in by the data of *I.E.(internal excitation)*.
Data-processing handbook of Description
D4.02 booklet:

Index A

3 situations are possible:

1)
S K
=
I
I then *E*
{*K*
I
I
.
1 }
1

2)
S K

=
I
I and *S*
K (*S*
), then *E*
{*K*, *K*
I
I
I
. *K*
1
2 }
1
I
i1
I is included in a face of *Ki1*
i2
is the mesh 3D which admits for face the face of *Ki* containing *S*

1
I.
3)
S K

=
I
I and *S*
K
(*S*
), then
1
I
i1
I is included in an edge of *Ki1*
E = {*K*, *K*, *K*
I
I
I
I
is the whole of the meshs 3D which admit for common edge
1
2
p}
the edge of *Ki* containing *S*
1
I.
Thus, an elementary segment can be obtained starting from several meshs.
10
Contents of the objects of the SD surface
N
S = [*A*,]
B
S
= [*A1*, *A2*]
K [*A1*, *A2*
I
I

I
I]
K
{*K*, *K*

I
I
1
pi}
I =1
SURFACE_1D
Name
OJB
Type
Length
Contents
14 19
20 24
.DESC
S V R
6
X
y Z
X
y Z
With
With
With
B
B
B, co-ordinates of
ends of the segment
REPERAGE_1D
Name
OJB
Type
Length
Contents
14 19
20 24
.SGTEL
.ORIG
S V R
N
T1, *T1*, *T1*, *T1*
1
2
I
N, coordinated
1

parametric of the points *Have* (origin)
.EXTR
S V R
N
t2, *t2*, *t2*, *t2*
1
2
I
N, coordinated
2
parametric of the points *Have* (end)
.TYPE
S V I
N
1
2
1 --> [*A*, *A*
I
I] is a sgt_arête
2 --> sgt_face
3 --> sgt_interne
Data-processing handbook of Description
D4.02 booklet:

Index A

Code_Aster ®

Version
4
Titrate:
Structures of Data curved and surface
Date:
28/01/1999
Author (S):
X. DESROCHES
Key:
D4.02.03
Page: 15/18
REPERAGE_
Name
OJB
Type
Length
Contents
14 19

20 24
.MAIL
XC V I
variable for
1
2
list meshs 3D containing [*A*, *A*
I
I]:
OC NMAX
OC = N
K1,
, *Ki*
I
Pi
.FACE
.ORIG
S V I
N
1
list numbers of face of *Ki* containing
A1
1
1
I -1 if *Have* is interior in *Ki*
.EXTR
S V I
N
2
idem .ORIG for *Have*
.CREFM
.ORIG
S V R
3n
1
1
coordinated reference of *Have* in *Ki*:
(*r1*, *r2*, *r3*) *I* =1,…, *N rj*
I
I
I
I
1 .
1
3

If *Have* is contained in a face, *laughed* is not
not used (see .CREFF)
.EXTR
S V R
3n
2
idem .ORIG for *Have*
.ARETE
.ORIG
S V I
N
list numbers of edge of *K i1* container
A1
1
I -1 if *Have* is interior with *K I*
1
.EXTR
S V I
N
2
idem .ORIG for *Have*
.CREFF
.ORIG
S V R
2n
1
coordinated reference of *Have* on the face
1
1
2
K
=
I the container: (*R*, *R*) *I*
1,…, *N*
I
I
.
.EXTR
S V R
2n
2
idem .ORIG for *Have*
Name
OJB
Type

Length
Contents
14 19
20 24
.CONEX
.ORIG
S V I
variable
pointer of related beginning of part in
REPERAGE_1D
.EXTR
S V I
variable
pointer of related end of part in
REPERAGE_1D
Data-processing handbook of Description
D4.02 booklet:

Index A

Code_Aster ®

Version
4
Titrate:
Structures of Data curved and surface
Date:
28/01/1999
Author (S):
X. DESROCHES
Key:
D4.02.03
Page: 16/18

11 Example of SD surfaces

```
%
% CONCEPT OF the TYPE surfaces
%
BEGINNING ();
PRE_GIBI ();
EMAIL =LIRE_MAILLAGE ();
&MAIL =DEFI_GROUP (GRID: EMAIL CREA_GROUP_NO: (TOUT_GROUP_MA: “YES”));
SEG1 = INTE_MAIL_3D (GRID: EMAIL
DEFI_SEGMENT: (ORIGIN: (.015 .02 0.)
END: (.055 .05 0.)));
impr_co (Co: seg1);
```

```
END ();
====> IMPR_CO OF THE STRUCTURE OF DATA: SEG1????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 17
================================================================================
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 .NOMA <
1 - >MAIL <
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 .NSDS <
1 - >SEG1 S1 <
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .ARETE.EXTR <
1 - 1 1
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .ARETE.ORIG <
1 - 1 1
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .CONEX.EXTR <
1 - 2
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .CONEX.ORIG <
1 - 1
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .CREFF.EXTR <
1 - 1.00000E+00 -1.00000E+00 1.00000E+00 -1.00000E+00
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .CREFF.ORIG <
1 - -1.00000E+00 -1.00000E+00 -1.00000E+00 -1.00000E+00
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .CREFM.EXTR <
1 - -1.00000E+00 -1.00000E+00 1.00000E+00 -1.00000E+00 -1.00000E+00
6 - 1.00000E+00
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .CREFM.ORIG <
1 - -1.00000E+00 -1.00000E+00 -1.00000E+00 -1.00000E+00 -1.00000E+00
6 - -1.00000E+00
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .DESC <
1 - 1.50000E-02 2.00000E-02 0.00000E+00 5.50000E-02 5.00000E-02
6 - 0.00000E+00
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >SEG1 S1 .FACE .EXTR <
1 - 2 2
```

SEGMENT IMPRESSION OF VALUES >SEG1 S1 .FACE .ORIG <
1 - 2 2

IMPRESSION OF THE COLLECTION: SEG1 S1 .MAIL
OBJECT IMPRESSION OF COLLECTION >SEG1 S1 .MAIL < OC: 1
1 - 2 1

OBJECT IMPRESSION OF COLLECTION >SEG1 S1 .MAIL < OC: 2
1 - 4 3

SEGMENT IMPRESSION OF VALUES >SEG1 S1 .SGTEL.EXTR <
1 - 5.00000E-01 1.00000E+00

SEGMENT IMPRESSION OF VALUES >SEG1 S1 .SGTEL.ORIG <
1 - 0.00000E+00 5.00000E-01

SEGMENT IMPRESSION OF VALUES >SEG1 S1 .SGTEL.TYPE <
1 - 1 1
====> FINE IMPR_CO OF STRUCTURE OF DATA: SEG1????????????????

Intentionally white left page.

Organization (S): EDF-R & D /AMA

D4.02 booklet: -
Document: D4.02.05

Description of the structure of data counts

Summary:

This document describes the structure of data counts.
Data-processing handbook of Description
D4.02 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data COUNTS

Date:
01/12/05
Author (S):
J. Key PELLET
:
D4.02.05-C Page
: 2/4

1
Tree structure of the structure of data

COUNT (K19):: == record

“.tbba” OJB S V K8

dim = 1

".TBNP" OJB S V I

dim = 2

".tblp" OJB S V K24
*dim = 4*nbre of parameters*

Note:

Contrary to what is written above, the name of a table does not have 19 characters; us will see below that the convention of names chosen for the objects containing the Boolean ones the table imposes that the true name of a table is limited to 17 characters.

2
Contents of basic objects JEVEUX

".tbba":
Described the base where the table is defined: "G", "V"
".tbnp":
(1) A number of parameters of the table

(2) A number of lines of the table
".tblp":
Described the parameters of the table. For each parameter:
(1) Name of the parameter
(2) Type of the parameter (I, R, C, K8, K16, K24, K32)
(3) Name of object JEVEUX containing the values associated with the parameter
(4) Name of object JEVEUX containing the Boolean ones associated the parameter

With each parameter 2 objects JEVEUX are associated which contain all the values defined in the table.

Names of objects JEVEUX created:

for the parameter of number ipar:

*.tblp ((ipar-1) *4 + 3) = nom_table (1: 19)/".00IJ"*
*.tblp ((ipar-1) *4 + 4) = nom_table (1: 17)/"LG.00IJ"*

where 00IJ is the number ipar coded on 4 characters (a number tallied on the right).

*The object ".00IJ" (of .tblp type ((ipar-1) *4 + 2)) contains the values of the table for*

parameter ipar.

This object is a vector JEVEUX dimensioned at least with the number of lines of the table. To know if the line ilgn of the table contains the parameter ipar, it is necessary to use the vector of "Boolean" "LG.00IJ":

"LG.00IJ" (ilign) = 0 (blank cell) or 1 (full cell).

3 Example

That is to say the table which is printed in the form:

WITH B
C
D
12 **-**
Z1
-
13 3.50000E+00
- -
14 **-**
- *15*

The contents of objects JEVEUX are as follows:

SEGMENT IMPRESSION OF VALUES >MA .0001 <
1 - 12 13 14 0 0
6 - 0 0
SEGMENT IMPRESSION OF VALUES >MA .0002 <
1 - 0.00000E+00 3.50000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00
SEGMENT IMPRESSION OF VALUES >MA .0003 <
1 - >Z1 <> <> <> <> <> <> <
SEGMENT IMPRESSION OF VALUES >MA .0004 <
1 - 0 0 15 0 0
6 - 0 0
SEGMENT IMPRESSION OF VALUES >MA .TBBA <
1 - >G <
SEGMENT IMPRESSION OF VALUES >MA .TBLP <
1 - >A <>I <
3 - >MA .0001<>MA LG.0001<
5 - >B <>R <
7 - >MA .0002<>MA LG.0002<
9 - >C <>K8 <
11 - >MA .0003<>MA LG.0003<
13 - >D <>I <
15 - >MA .0004<>MA LG.0004<
SEGMENT IMPRESSION OF VALUES >MA .TBNP <
1 - 4 3
SEGMENT IMPRESSION OF VALUES >MA LG.0001 <
1 - 1 1 1 0 0 0 0
SEGMENT IMPRESSION OF VALUES >MA LG.0002 <
1 - 0 1 0 0 0 0 0
SEGMENT IMPRESSION OF VALUES >MA LG.0003 <
1 - 1 0 0 0 0 0 0
SEGMENT IMPRESSION OF VALUES >MA LG.0004 <
1 - 0 0 1 0 0 0 0

01/12/05
Author (S):
J. Key PELLET
:
D4.02.05-C Page
: 4/4

Intentionally white left page.
Data-processing handbook of Description
D4.02 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structures of data FORMAT_IDEAS

Date:
01/12/05
Author (S):
J. PELLET, F. LEBOUVIER Key
:

D4.02.06-B Page
: 1/8

Organization (S): EDF-R & D /AMA, Cad Delta

Data-processing handbook of Description
D4.02 booklet: -
D4.02.06 document

Structure of data FORMAT_IDEAS

Summary:

One describes here the structure of data FORMAT_IDEAS. This SD is used at the time of the éxécution of the order
LIRE_RESU, it makes it possible to locate and extract from the universal file "unv" of IDEAS, the results desired by
the user.

Count

matters

Code_Aster ®
Version
8.1

Titrate:
Structures of data FORMAT_IDEAS

Date:
01/12/05
Author (S):

1 General information

The user defined in order LIRE_RESU information (TYPE_RESU, NOM_CHAMP, INST,…) who characterize the result which it wishes to extract from the universal file of IDEAS. These information is not exploitable directly, it is necessary to convert them with the format of "universal" file.

The structure of data FORMAT_IDEAS contains all the search criteria to the "universal" format, allowing to extract the results desired by the user. If the user does not specify in syntax of order LIRE_RESU, the key word factor FORMAT_IDEAS, the structure of data FORMAT_IDEAS is initialized by defect [U4.26.03]. In the contrary case, the user defines its proper search criteria.

2 Tree structure

FORMAT_IDEAS (K16)
:: = record

".FID_NOM":
OJB S
V
K16

Length = nbnoch
".FID_NUM"
:
OJB S
V
I
Length
=
nbnoch
".FID_PAR":
OJB
S
V

I
Length
=
*nbnoch*800*
".FID_LOC"
:
OJB S
V
I
Length
=
*nbnoch*10*
".FID_CMP":
OJB S
V
K8
Length
=
*nbnoch*1000*
".FID_NBC":
OJB S
V
I
Length
=
nbnoch

3
Contents of the objects

Convention:

nbnoch = a number of fields to reading

· ".FID_NOM": S V K16

This object contains the name of the fields to reading.

For I = 1, nbnoch
v (I): name of the iième field to reading

ex: "DEPL", "QUICKLY",…, "SIEF_ELNO"

.

"FID_NUM": S V I

This object contains for each field to reading the number of the associated dataset.

For I = 1, nbnoch
v (I): number of the dataset associated the iième field with reading

ex: 55, 57, 2414

.

".FID_PAR": S V I

This object contains for each field to reading the characteristics of the heading [§4] of the dataset sought. This heading is composed to the maximum of 20 "records" made up of 40 "fields" each one.

V (1): 1ier field of record 1 of field 1
...
V (48): 8th field of record 2 of field 1
...
V (800): 40ième field of record 20 of field 1
...
V (6401): 1ier field of record 1 of field 9
...

V (7200): 40ième field of record 20 of field 9
...

v ((ich-1) *800+ (irec-1) *40+ifield): value associated with the field ich located at the recording irec and for the field ifield

.
".FID_LOC": S V I

This object contains for each field, 5 couples of whole values making it possible to locate with the interior of the dataset, the sequence number, the moment, the frequency… the first value indicates it n° of the recording where is stored information and the second value indicates its position.

v (1) = N° of the recording
Sequence number

v (2) = Position
v (3) = N° of the recording
Moment
Field n°1
v (4) = Position
v (5) = N° of the recording
Frequency

v (6) = Position

v (7) = N° of the recording
Nume_mode

v (8) = Position

v (9) = N° of the recording
Mass_gene

v (10) = Position

v (11) = N° of the recording
Sequence number

v (13) = Position
v (13) = N° of the recording
Moment

Field n°2
v (14) = Position
...

.
".FID_CMP": S V K8

This object contains for each field the components of the size to reading.

v (1) = DX
v (2) = DY
v (3) = DZ
Field n°1
v (4) = DRX
v (5) = DRY
v (6) = DRZ

v (1001) = EPXX
v (1002) = EPYY
v (1003) = EPZZ
Field n°2
v (1004) = EPXY

v (1005) = EPXZ
v (1006) = EPYZ

...

...

.
“.FID_NBC”: S V I

This object contains for each field the component count to reading.

4
Structure of the datasets

The general structure of the datasets results 55, 57 and 2414 exploited by order LIRE_RESU is made up of 2 parts:

- *Part a: heading containing of general information,*
- *Part b: contains the values.*

-1
Part A
55% VALUES WITH THE NODES
Record 1
ASTER 5.01.00 CONCEPT TEMPLE CALCULATES - FIELD WITH THE NODES OF…
Record 2
FIELD WITH THE NODES OF REFERENCE SYMBOL TEMP - TEMP
Record 3
ASTER 5.01.00 CONCEPT TEMPLE CALCULATES THE 18/12/98 A 15:19: 49 OF…
Record 4
FIELD WITH THE NODES OF REFERENCE SYMBOL TEMP
Record 5
SEQUENCE NUMBER: 0 INST: 0.00000E+00
Record 6
2 4 1 5 2 1
Record 7
2 1 1 0
Record 8
0.00000E+00
1% N1 NODE

Part B

205% N205 NODE
1.00000E+02
-1

Appear 4-a: Dataset n°55 (example)
Data-processing handbook of Description
D4.02 booklet: -
HT-66/05/003/A

-1
Part A
57% VALUES WITH THE NODES OF THE ELEMENTS
Record 1
ASTER 3.05.30 CONCEPT U CALCULATES IT - FIELD BY ELEMENT WITH…
Record 2
FIELD BY ELEMENT WITH THE NODES OF REFERENCE SYMBOL VARI_ELNO_ELGA -…
ASTER 3.05.30 CONCEPT U CALCULATES THE 29/12/95 A 09:56: 55 OF TYPE…
Record 3
FIELD BY ELEMENT WITH THE NODES OF REFERENCE SYMBOL VARI_ELNO_ELGA
Record 4
Record 5
Record 6
1 4 3 0 2 6
Record 7

2 1 1 1
Record 8
0.15000E+02
1 1 8.6% MESH MA2
Part B
2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.

2.07919E-05 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.
-1

Appear 4-b: Dataset n°57 (example)

-1
Part A
2414
Record 1
1
Record 2
B.C. 1, TEMPERATURE_1, LOAD SET 1
1
Record 3
/users/lebouv/SGI/Code_Aster/TPLS100B/tpls100_coque.mf1
Record 4
Record 5
MODEL_SOLUTION_SOLVE
Record 6
LOAD SET 1
Record 7
Analysis time was Jan 6 - 99 11:11: 25
Record 8
NONE
Record 9
2 1 1 5 2 1
Record 10
-10 0 1 1 1 0 0 0
Record 11
2 0
Record 12
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.
0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.
Record 13
1

Part B
-6.10352E-06

205
-6.10352E-06
-1

Appear 4-c: Dataset n° 2414 (Example)

5
Default values

The default values stored in the structure of data FORMAT_IDEAS are presented in document of use [U4.26.03].

6 Examples

In this paragraph, we present two examples:

- *Example a: use of the search criteria per defect to read the results,*
- *Example b: use of the search criteria user to read the results.*

For each one of these examples we give the syntax of order LIRE_RESU as well as *contents of the structure of data FORMAT_IDEAS.*

Example a: search criteria per defect

INIT = LIRE_RESU (GRID = m,
UNIT
=
19
,
FORMAT
=
“IDEAS”,
TYPE_RESU
=
“EVOL_NOLI”,
NOM_CHAM
=
(“DEPL”),
INST
=
15.
,

)
Appear 6-a: Syntax of order LIRE_RESU

FORMAT_IDEAS (1) (2) (3) (4) (5) (6)

. “FID_NOM”
DEPL

. “FID_NUM”
55

. "FID_PAR" (1) 9999 9999 9999 9999 9999 9999

(2) 9999 9999 9999 9999 9999 9999

(3) 9999 9999 9999 9999 9999 9999

(4) 9999 9999 9999 9999 9999 9999

(5) 9999 9999 9999 9999 9999 9999

(6)
1 4 3 8 2 6

(7) 9999 9999 9999 9999 9999 9999

(13) 9999 9999 9999 9999 9999 9999

(40)

“.FID_LOC”
7 4 8 1
9999 9999

“.FID_CMP”
“DX” “DY” “DZ” “DRX” “DRY” “DRZ”

Table 6-a: Contents of SD FORMAT_IDEAS (default values)

At the time of the research of the dataset, number 9999 is a joker making it possible to be unaware of the value read in
the heading.

Example a: search criteria defined by the user.

INIT = LIRE_RESU (MODEL
= Mo

,
UNIT
=
19
,
FORMAT
=
“IDEAS”,
TYPE_RESU
=
“EVOL_NOLI”,
NOM_CHAM
=
(“15”)
,
INST
=
15.
,
FORMAT_IDEAS
=
_F
(
NOM_CHAM
=
“DEPL”,
NUME_DATASET
=
55,
RECORD_6
=
(3,1,9999,4,2,3),
POSI_ORDRE =
(8,4),
POSI_INST
=
(7,9999),
CMP
=
(“DX”, “DY”, “DZ”),

)

)

Appear 6-b: Syntax of order LIRE_RESU

FORMAT_IDEAS (1) (2) (3) (4) (5) (6)

. “FID_NOM”
DEPL

. “FID_NUM”
55

. “FID_PAR” (1) 9999 9999 9999 9999 9999 9999

(2) 9999 9999 9999 9999 9999 9999

(3) 9999 9999 9999 9999 9999 9999

(4) 9999 9999 9999 9999 9999 9999

(5) 9999 9999 9999 9999 9999 9999

(6)
3 1
9999
4 2 3

(7) 9999 9999 9999 9999 9999 9999

(13) 9999 9999 9999 9999 9999 9999

(40)

".FID_LOC"

8 4 7 1
9999 9999

“.FID_CMP”
“DX” “DY” “DZ” “XXX” “XXX” “XXX”

Structure of Data tabl_trc

1 Goal

One explains in this document how to exploit a structure of data tabl_trc in one elementary routine of calculation te00ij.

2
The structure of data TABL_TRC is a TABLE

The structure of data tabl_trc is by means of computer a TABLE [D4.02.05]. It is made up columns of values (SPEED, PARA_EQ,…) cf [§5]. The goal of this document is not to describe an already described SD, but to show how one recovers in routines TE00IJ the values data by the user in order DEFI_TRC.

Order DEFI_TRC builds diagonal TABLE a “per blocks”. This TABLE “is linearized” in the total orders of mechanics via the routine tbexlr [D6.06.01] called by the routine of construction of coded material (rcmaco); i.e. it is transformed into a list of realities who is then accessible in routines TE00IJ.

3
Let us start from an example, the test hsnv101a

trc = DEFI_TRC (HIST_EXP: (VALE: (-1.106D+03 1.100D+01 8.563D+00 -2.760D-02
1.220D-04 -2.955D-07 3.402D-10 -1.517D-13
0.000D+00 0.000D+00 0.000D+00 8.360D+02
0.000D+00 0.000D+00 0.000D+00 6.001D+02
0.000D+00 0.000D+00 1.000D+00 3.450D+02)
)
(VALE: (-2.206D+03 1.100D+01 8.563D+00 -2.760D-02
1.220D-04 -2.955D-07 3.402D-10 -1.517D-13
0.000D+00 0.000D+00 0.000D+00 8.360D+02
0.000D+00 0.000D+00 0.000D+00 6.001D+02
0.000D+00 0.000D+00 1.000D+00 3.450D+02)
)
TEMP_MS: (P: 1.100D+01
THRESHOLD: 4.500D-01
AKM: -3.125D+01
BKM: 1.406D+01
TPLM: -3.497D+03
)
) ;

4
Explanations of the values given

4.1 word
key
factor
HIST_EXP

The first 8 values of key word VALE under the key word factor HIST_EXP define the history thermics:

· the first value is the value of derived from the function T (T) speed of cooling,
· the second value is the defining parameter of equivalence temps_température austenitization,
· the 6 following values define the coefficients of the monomists of degree 0 to 5 such as polynomial of a nature 5 thus built either the interpolation between AR3 and TMF within the

meaning of least
squares of the function F (T) deduced from the thermal history and such as F (T) = ln (T (T)).

The following values (necessarily by group of 4) define the respective proportions of
ferrite, pearlite and bainite present at a temperature given for the experimental thermal history defined by the first 8 values.

4.2
key word factor TEMP_MS

These 5 values define the sizes intervening in the law of evolution of temperature ms in
function of the conditions of austenitization and the quantities of ferrite, pearlite and bainite already formed.
This law is associated a diagram TRC.

5
Definition of the blocks

The table thus produced is "diagonal per blocks", i.e. the table breaks up into blocks.

1) The first 8 values of key word VALE under the key word factor HIST_EXP are found
in block 1 with quadrupled defining NB_POINT,
2) Block 2 defines the respective proportions of ferrite, pearlite and bainite, the following values
key word VALE under the key word factor HIST_EXP,
3) Block 3 defines the laws associated with each diagram TRC.

6
Impression of the table of the tabl_trc type

Block 1:

SPEED PARA_EQ COEF_0 COEF_1 COEF_2 COEF_3 COEF_4 COEF_5 NB_POINT
1.106E+03 1.100E+01 8.563E+00 -2.760E-02 1.220E-04 -2.955E-07 3.402E-10 -1.517E-13
3.
2.206E+03 1.100E+01 8.563E+00 -2.760E-02 1.220E-04 -2.955E-07 3.402E-10 -1.517E-13
3.

Block 2:
Z1 Z2 Z3
TEMP
0.00000E+00 0.00000E+00 0.00000E+00 8.36000E+02
0.00000E+00 0.00000E+00 0.00000E+00 6.00100E+02
0.00000E+00 0.00000E+00 1.00000E+00 3.45000E+02
0.00000E+00 0.00000E+00 0.00000E+00 8.36000E+02
0.00000E+00 0.00000E+00 0.00000E+00 6.00100E+02
0.00000E+00 0.00000E+00 1.00000E+00 3.45000E+02

Block 3:
P THRESHOLD
AKM BKM
TPLM
1.10000E+01 4.50000E-01 -3.12500E+01 1.40600E+01 -3.49700E+03

7
List realities produced by linearization of the table

3.00000E+00
9.00000E+00 2.00000E+00
1.10600E+03 1.10000E+01 8.56300E+00 -2.76000E-02 1.22000E-04 2.95500E-07 3.40200E-10
-1.51700E-13 3.00000E+00
2.20600E+03 1.10000E+01 8.56300E+00 -2.76000E-02 1.22000E-04 2.95500E-07 3.40200E-10
-1.51700E-13 3.00000E+00
4.00000E+00 6.00000E+00
0.00000E+00 0.00000E+00 0.00000E+00 8.36000E+02
0.00000E+00 0.00000E+00 0.00000E+00 6.00100E+02
0.00000E+00 0.00000E+00 1.00000E+00 3.45000E+02
0.00000E+00 0.00000E+00 0.00000E+00 8.36000E+02
0.00000E+00 0.00000E+00 0.00000E+00 6.00100E+02
0.00000E+00 0.00000E+00 1.00000E+00 3.45000E+02

5.00000E+00 1.00000E+00
1.10000E+01 4.50000E-01 -3.12500E+01 1.40600E+01 -3.49700E+03
Data-processing handbook of Description
D4.03 booklet: Parameterized operators
HI-75/01/005/A

3.00000E+00
the table is composed of 3 blocks
9.00000E+00 2.00000E+00
1ier block made up of 9 columns and 2 lines
. . .
values of the 1ier block line by line
4.00000E+00 6.00000E+00
2nd block made up of 4 columns and 6 lines
. . .
values of the 2nd block line by line
5.00000E+00 1.00000E+00
3rd block made up of 5 columns and 1 line

values of the 3rd block

8 *Examples to recover a value in the list of realities*

8.1 example 1

If one wants to recover the SPEED of the second key word factor HIST_EXP, it is necessary to shift:

quickly = ListR8 (1 + 2 + 9 + 1)
1: the number of blocks defines
2: 2 numbers to dimension the 1ier block
9: 9 values to define a key word factor

1: "SPEED" is in position 1

8.2 example
2

2 quantities which one finds in the te00ij:
NBHIST
a number of thermal stories experimental

equal to the number of key words factors HIST_EXP

equal to the number of lines of block 1

NBTRC
a number of laws of evolution of the temperature

equal to the number of key words factors TEMP_MS

equal to the number of lines of block 3

in our example, we find these values:
NBHIST = ListR8 (1 + 2) = 2
*NBTRC = ListR8 (1 + 2 + 9*2 + 2 + 4*6 + 2) = 1*

Data-processing handbook of Description
D4.04 booklet: -
Document: D4.04.01

Structure of Data CATA_ELEM

1 General information

The structure of data CATA_ELEM gathers all the information provided in the files of catalogues of finite elements [D3.02.01].

This SD is created by the procedure of update of code MAJNEW and is safeguarded in the base elements. This base is recopied in the base of the user during the ordering BEGINNING. objects which make this SD are then accessible in reading by all the operators from the code.

There is only one SD of the type CATA_ELEM; its name is "&CATA".

SD CATA_GRANDEUR contains
information of the catalogue
compelem/grandeur_simple__.cata

SD CATA_TYPE_MAILLE contains
information of the catalogue
compelem/type_maille__.cata

SD CATA_OPTION contains
information of the catalogues
options/.cata*

SD CATA_TYPE_ELEM contains information of the catalogues typelem/*.cata

SD CATA_PHEN_MODE contains information of the catalogue compelem/phenomene_modelisation__.cata

Note:

All objects described in this document (except the 4 &CATA.TE.DIM_GEOM objects, &CATA.TE.OPTTE, &CATA.TE.TAILLMAX and &CATA.TE.NBLIGCOL) are created by scripts python of Lecture_Cata_Ele/*.py. These scripts generate an ASCII file containing these objects which are then read again by routine FORTRAN lccata.f. This routine calculates the 4 missing objects then.

---

2 Tree structures

CATA_ELEM (K5):: =record
".CL": CATA_COM_LIBR
".GD": CATA_GRANDEUR
".TM": CATA_TYPE_MAILLE
".OP": CATA_OPTION
".TE": CATA_TYPE_ELEM

“$VIDE”: CATA_PHEN_MODE

CATA_COM_LIBR (K8)
:: =record
“.COMLIBR”
:

OJB
XC
V
K80
NAKED
LONG=1

CATA_GRANDEUR (K8)
:: =record
“.DESCRIGD”

: OJB XC V I
NO
LONG=7
“.NOMCMP”
:

OJB
XC
V
K8
NO
“.NOMGD”
:

OJB
S
NR
K8
“.TYPEGD”
:

OJB
S
V
K8

CATA_TYPE_MAILLE (K8)
:: =record
".NBNO"
:

OJB
XC
V
I
NO
LONG=1
NBOBJ=
nb_tm
".NOMTM"
:

OJB
S
NR
K8
LONG=
nb_tm
".TMDIM"
:

OJB
S
V
I
LONG=
nb_tm
".NOELRF"
:

OJB
S
NR
K8
LONG=
nb_elrefe
".NOFPG"
:

OJB
S
NR
K16
LONG=
nb_fam_pg
“.TMELRF”
:

OJB
S
V
I
LONG=
nb_elrefe
“.TMFPG”
:

OJB
S
V
I
LONG=
nb_fam_pg

CATA_OPTION (K8):: =record
“.DESCOPT”

: OJB XC V I

NO
“.NOMOPT”
:

OJB
S
NR
K16
“.OPTPARA”
:

OJB
XC

V
K8
NO

CATA_TYPE_ELEM (K8):: =record
".DIM_GEOM"

: OJB S V I
".MODELOC"

: OJB XC V I
NO
".NBLIGCOL"

: OJB S V I
".NOMMOLOC":

OJB
S
NR
K24
".NOMTE"
:

OJB
S
NR
K16
".OPTMOD"

: OJB XC V I
NAKED
".OPTNOM"

: OJB XC V K8
NAKED
".OPTTE"

: OJB S V I
“.TAILLMAX”

: OJB S V I
“.TYPEMA”
:

OJB
S
V
K8
“.NBELREFE”:

OJB
S
V
I
*LONG=2*nb_te*
“.NOELREFE”:

OJB
S
V
K8
“.PNLOCFPG”:

OJB
S
V
K32
LONG=nb_loc_fpg
“.NOLOCFPG”:

OJB
S
V
I
LONG=nb_loc_fpg
“.NOFPG_LISTE”

: OJB S NR K24

“.FPG_LISTE”

: OJB XC V K8
NAKED
“.CTE_ATTR”

: OJB S V K16

*LONG=2*nb_attributs*

CATA_PHEN_MODE (K5):: =record
“.PHENOMENE”:

OJB
S
NR
K16
“.ACOUSTIQUE .MODL”
: OJB S NR K16
“.ACOUSTIQUE”

: OJB XC V I NO
“.MECANIQUE .MODL”
: ...

...

3
Contents of the OJB

3.1 Notations, dimensions

nb_te
numbers type_element catalogue
nb_tm
type_maille catalogue numbers
nb_op
option of the catalogue numbers
nb_gd
size of the catalogue numbers

3.2 SD CATA_GRANDEUR: "&CATA.CL"

CATA_COM_LIBR (K8)
:: =record

".COMLIBR"

: OJB XC V K80

NAKED LONG=1

.COMLIBR:
This object contains the "free comments" which one can write in certain catalogues enters << blah. >>. Currently, one can write some in the catalogue grandeur_simple__ and in the catalogues of options.
A free comment is a contiguous continuation of K80 stored in object .COMLIBR. It is necessary then to store (elsewhere!) the number of lines and the number of the 1st line of the free comment.

3.3 SD CATA_GRANDEUR: "&CATA.GD"

CATA_GRANDEUR (K8)
:: =record

“.DESCRIGD”

: OJB XC V I

NO LONG=7

“.NOMCMP”

: OJB XC V K8

NO

“.NOMGD”

: OJB S NR K8

“.TYPEGD”

: OJB S V K8

.NOMGD:

Pointer of name allowing to associate all the sizes (simple or elementary) one number. It is this number which we will identify thereafter with the size.

Note:

Collections .DESCRIGD and .NOMCMP are numbered in the same way that .NOMGD.

.NOMCMP:

Collection of V (K8). One reaches it by the number of the size: Gd, or by its name. All the simple sizes have all their named CMP. One thus finds opposite Gd, the list of all the names of the Gd CMP. If the size is elementary, there is nothing opposite of Gd.

.TYPEGD: V (K8).
Gd ---> K8: type_scalaire (size) (R, I, C, K8, K16, K24)
Data-processing handbook of Description

.DESCRIGD: contiguous collection of V (I) length 7.
Gd ---> V (I): descriptor of the size Gd.

V (1): code_gd
1: simple size
3: elementary size (vector)
4: elementary size (matrice_sym)
5: elementary size (matrice_rectangle)
V (3): n_ec
numbers the entier_codés necessary ones to describe the CMP of
size.
V (4): gd_ligne
size "line" for the elementary sizes "vector" and
"matrix".
V (5): gd_colonne
size "column" for the elementary sizes "stamps".
V (6): nblcom
A number of lines of the free comment associated the size Gd
V (7): indcom
Index in `&CATA.CL.COMLIBR `of the 1st line of the comment
free associated the size Gd

3.3 SD
CATA_TYPE_MAILLE: "&CATA.TM"

This catalogue contains the information contained in the catalogue type_maille__.cata

That is to say:
nb_tm: type_maille numbers
nb_elrefe: a number of ELREFE
nb_fam_pg: a number of families of points of Gauss

CATA_TYPE_MAILLE (K8)
:: =record
".NBNO"

: OJB XC V I NO LONG=1 NBOBJ= nb_tm
".NOMTM"

: OJB S NR K8 LONG=
nb_tm
".TMDIM"

: OJB S V I LONG= nb_tm
".NOELRF"

: OJB S NR K8 LONG=
nb_elrefe
".NOFPG"

: OJB S NR K16
LONG= nb_fam_pg
".TMELRF"

: OJB S V I
LONG= nb_elrefe

".TMFPG"

: OJB S V I
LONG= nb_fam_pg

.NOMTM: This pointer of name contains the names of the type_maille (K8)
.NOELRF: This pointer of name contains the names of the ELREFE (K8)
.NOFPG: This pointer of name contains the names of the families of points of Gauss.
The name of a family of points of Gauss (K16) is obtained by concaténant the name of the ELREFE (K8) and
the surname in this ELREFE (K8). For example: "HE8 FPG1"
.NBNO: NBNO (i_tm): a number of nodes for the type_maille i_tm
.TMDIM: TMDIM (i_tm): topological dimension of the type_maille (0/1/2/3)
.TMELRF: TMELRF (i_elrf): number of the type_maille associated the ELREFE i_elrf.
.TMFPG: TMFPG (i_fpg): a number of points of Gauss for the i_fpg family.

3.4 SD
CATA_OPTION: "&CATA.OP"

CATA_OPTION (K8)
:: =record

".DESCOPT"

: OJB XC V I

NO
".NOMOPT"

: OJB S NR K16
".OPTPARA"

: OJB XC V K8 NO

.NOMOPT:

Pointer of name (K16) making it possible to associate has all the options a number that one will confuse with the option: opt.

.DESCOPT:

Contiguous collection of V (I).
opt ---> DESCOPT (opt) = V
The length of V is 6+3 (nbin+nbou) with:*
nbin: a number of parameters "in" option
nbou: a number of parameters "out" of the option

V (1): 1
useless
V (2): nbin
a number of parameters "in"
V (3): nbou
a number of parameters "out"
V (4): 1
useless
V (4+1): Gd (in, 1)
size associated with the parameter "in" 1
V (4+2): Gd (in, 2)
size associated with the parameter "in" 2
...

V (4+nbin+1): Gd (out, 1)
size associated with the parameter "out" 1

...

V (4+nbin+nbou):
size associated with the last parameter "out"

Gd (out, nbou)

V (4+nbin+nbou+1): nblcom
A number of lines of the free comment general associated with
the option.
V (4+nbin+nbou+2): indcom
Index in `&CATA.CL.COMLIBR `of the 1st line of
free comment general associated with the option

Then the free comments associated with the various parameters ("in" or "out come") with the option:

V (6+nbin+nbou+1): nblcom
A number of lines of the free comment associated 1st
parameter "in"
V (6+nbin+nbou+2): indcom
Index in `&CATA.CL.COMLIBR `of the 1st line of
free comment associated the 1st parameter "in"
...

V (6+3 (nbin+nbou) - 1): nblcom*
A number of lines of the free comment associated
last parameter "out"
V (6+3 (nbin+nbou)): indcom*
Index in `&CATA.CL.COMLIBR `of the 1st line of
free comment associated the last parameter "out"

.OPTPARA

Contiguous collection of V (K8).
opt ---> NOMPARA (opt) = V

V (1): will nom_para (in, 1)
name of the parameter "in" number 1
V (2): will nom_para (in, 2)
name of the parameter "in" number 2
...

V (nbin+nbou): will nom_para (out, nbout)
name of the last parameter "out"

3.5 SD
CATA_TYPE_ELEM: "&CATA.TE"

CATA_TYPE_ELEM (K8)
:: =record
".DIM_GEOM"

: OJB S V I
".MODELOC"

: OJB XC V I
NO
".NBLIGCOL"

: OJB S V I
".NOMMOLOC"

: OJB S NR K24
".NOMTE"

: OJB S NR K16
“.OPTMOD”

: OJB XC V I
NAKED
“.OPTNOM”

: OJB XC V K8
NAKED
“.OPTTE”

: OJB S V I
“.TAILLMAX”

: OJB S V I
“.TYPEMA”

: OJB S V K8
“.NBELREFE”

: OJB S V I
*LONG=2*nb_te*
“.NOELREFE”

: OJB S V K8
“.PNLOCFPG”

: OJB S V K32
LONG=nb_loc_fpg
“.NOLOCFPG”

: OJB S V I
LONG=nb_loc_fpg
“.NOFPG_LISTE”

: OJB S NR K24

“.FPG_LISTE”

: OJB XC V NAKED K8

“.CTE_ATTR”

: OJB S V K16
*LONG=2*nb_attributs*

3.5.1 Dimensions

.NBLIGCOL: vector of entireties length 6: V.

V (1)
nb_op: a number of options
V (2)
nb_te: numbers type_element
V (3)
nb_te: numbers type_element
V (4)
nb_gd: a number of sizes
V (5)
nb_te: numbers type_element
V (6)
nb_gd: a number of sizes

3.5.2 Name, TYPE_MAILLE, geometrical dimension, families of integration of TYPE_ELEMENT

.NOMTE: Pointer of name allowing to associate type_element a number (of 1 to N) which allows to identify it: you.

.TYPEMA: vector (K8) length nb_te: V
V (you): name of the type_maille associated type_element.

.NBELREFE: vector (I) length 2*nb_te: V
V (2* (you-1) +1): a number of ELREFE for type_element you.

V (2 (you-1) +2): address in .NOELREFE of the 1st ELREFE for type_element you.*

.NOELREFE: vector (K8: names of the ELREFE of all type_element.
V (.NBELREFE (2 (you-1) +2+k-1)) : name of the kth ELREFE of type_element you.*
Data-processing handbook of Description
D4.04 booklet: -
HT-66/05/003/A

.PNLOCFPG:
Pointer of name allowing to associate a "local family of points of Gauss" a number which as index in the object "&CATA.TE.NOLOCFPG will be used". A "local family of points of Gauss" is identified by a name (K32) obtained while concaténant: the name of type_element (K16), the name of the ELREFE (K8) and the surname (K8).
For example:
ENTETE__ ELEMENT__ THER_PENTA6_D MAILLE__ PENTA6
ELREFE__ PE6 GAUSS__ RIGI=FPG1
The "local family of points of Gauss" will be called: "THER_PENTA6_D PE6 RIGI"

Caution:

1) the pointers of names JEVEUX being limited to K24, object .PNLOCFPG is not a truth pointer of names. It is simply about a vector of K32. To make the equivalent of JENUNO, it is necessary to traverse the vector until finding the name sought. The index of the name in vector is the sought number.
2) Certain families are "simple here" (: RIGI) of others are "lists" (see paragraph

below).

.NOLOCFPG:
Vector of entireties allowing "to point" towards the .TM.NOFPG objects and .TM.TMFPG
For a "simple" family: .NOLOCFPG > 0
For a family "lists": .NOLOCFPG = 0

In short, the use of objects .PNLOCFPG and .NOLOCFPG will be done in FORTRAN (for one "simple" family) by:
NOFLPG=TYPELE//ELREFE//FAMILL ("local" name of a family of PG (K32))
NUFLPG=INDK32 ("&CATA.TE.PNLOCFPG", NOFLPG)
NUFGPG=&CATA.TE.NOLOCFPG (NUFLPG)
NOFGPG=&CATA.TM.NOFPG (NUFGPG) ("total" name of the family (K16))
NBPOIN=&CATA.TM.TMFPG (NUFGPG) (a number of points of the family)

.DIM_GEOM: vector (I) length nb_te: V
V (you): geometrical dimension associated type_element
/0: type_element does not know size GEOM_R
/1: type_element knows CMP DX of size GEOM_R
/2: type_element knows the CMP DY of size GEOM_R
/3: type_element knows CMP DZ of size GEOM_R

3.5.2.1 Families of PG "lists"

One can define in the catalogues of type_element families which are lists of families existing ("simple").

For example:
ENTETE__ ELEMENT__ MAILLE__ HEXA20
ELREFE__ H20 GAUSS__ RIGI=FPG27 MASS=FPG8 FPG_LISTE__ MATER= (RIGI, MASS)

For type_element the, the family called MATER is a family of 35 items (27+8). 3rd not RIGI is the 3rd point of MATER the 3 point of MASS is the 30ème not MATER.

One stores this information in the 2 following objects:

.NOFPG_LISTE: OJB S NR K24
It is a pointer of names making it possible to point in 2nd object (.FPG_LISTE)
The name of a family "lists" (NOFPGL2) is K24:
NOFPGL2=NOMTE (1: 16) //NOFPGL (1: 8) if NOFPGL is the name given to the family "lists" (MATER in our example).
Data-processing handbook of Description
D4.04 booklet: -
HT-66/05/003/A

.NOFPG_LISTE (NOFPGL2) - > KFPGL

.FPG_LISTE: OJB XC V NAKED K8 ()
The access to this collection is done thanks to the preceding object (.NOFPG_LISTE).
.FPG_LISTE (KFPGL) = V (K8)
This vector of K8 is dimensioned with nb_fam +1
V (ifam): surname ifam of the list.
V (nb_fam +1): name of the elrefe.

For our example: V= ("RIGI", "MASS", "H20")

3.5.3 Modes
buildings

One decides type_element to identify the local modes of all by an entirety: moloc. This entirety is single for each couple (type_element, definition of local mode)

.NOMMOLOC:
Pointer of name. (K24)
With each made up name: nom_te//nom_mode one can associate a number: moloc.
ex: "DKT"/"NGEOMER" <----> 67.
moloc varies from 1 with nb_mode_locaux (total on all type_element). moloc serves as
pointer of access to collection .MODELOC

.MODELOC:
Contiguous collection of V (I).
moloc ---> V (I)

V (1): code
1: ELEM__
2: ELNO__
3: ELGA__
4: VECTEUR__
5: MATRICE__
V (2): Gd
size associated with the mode_local
V (3): nb_scal
a number of scalars (I, R.) representing the local mode
(i.e length of the local field).

If code = EL. : _ _
V (4): nb_pt

nb_pt is the number of points of localization of the field on the element:
·
for 1 local mode of type ELEM__, nb_pt = 1,
·
for 1 local mode of type ELNO__, nb_pt is the number of nodes of the element,
·
for 1 local mode of type ELGA__, nb_pt is the number of points of Gauss of the element.

One adds 10000 to the absolute value of nb_pt to possibly state that them various points of the field do not have same representation (ELNO__/DIFF__).

If ELNO__/DIFF__:
V (4+1)
beginning of the descripteur_grandor of item 1
....

V (4+n_ec (i-1) +1)*
beginning of the descripteur_grandor of item I
If not:
V (4+1) and the continuation are the descripteur_grandor (Gd).

if code = ELGA__:
/
V (4+n_ec+1): +NUFGPG if this family is "simple".
/
V (4+n_ec+1): - KFPGL if this family is "list".
Data-processing handbook of Description

NUFGPG is the number of the family "simple" partner with the mode_local. Pointer in the object "&CATA.TM.NOFPG".

KFPGL is the number of the family "lists" associated with the mode_local. Pointer in the object "&CATA.TE.FPG_LISTE".

If code = VECTEUR__ or MATRICE__
V (4): moloc (line)
If code = MATRICE__
V (5): moloc (column)

.TAILLMAX: vector (I) length nb_te: V

V (you): Max (.MODELOC (3)) for all the local modes of type_element you

3.5.4 Options calculated by type_element

.OPTTE: Object simple V (I).

*V ((you-1) *nb_op+op) ---> i_optte: number of the optte (OPTion-Type_Element) associated CALCULATION (opt, you).*
This number i_optte is used to point in collections .OPTMOD and .OPTNOM.

.OPTMOD: Contiguous collection of V (I).

This collection describes the local modes of the elementary options.

i_optte ---> V (I)

V (1)
num_calc
number of elementary calculation
V (2)
nbin
parameter numbers "in"
V (3)
nbout
parameter "out numbers"
V (3+1)
moloc_in_1
local mode of the first parameter "in"
V (3+2)
moloc_in_2
local mode of the second parameter "in"
...

V (3+nbin+1)
moloc_ou_1
local mode of the first parameter "out"
...

V (3+nbin+nbou)
moloc_ou_nbout
local mode of the last parameter "out"

.OPTNOM:

Contiguous collection of V (K8). This collection describes the names of parameters of the options elementary.

i_optte ---> V (K8)

V (1)
will nom_para (in, 1)
...

V (nbin+1)

will nom_para (out, 1)
...

V (nbin+nbou)
will nom_para (out, nbout)

Data-processing handbook of Description
D4.04 booklet: -
HT-66/05/003/A

3.5.5 Object
".CTE_ATTR"

.CTE_ATTR: Collection of V (K16) length nb_te. This collection contains the attributes of all type_element.

*.CTE_ATTR (you): V (K16) LONG=2*nb_attribut*
V (2 (iattr-1) +1): name of the attribute of number iattr*
V (2 (iattr-1) +2): value of the attribute of number iattr*

Note:

To find the value of an attribute of name nom_attr, one must traverse this vector until to find this name with an odd index.

3.6 SD
CATA_PHEN_MODE: "&CATA"

CATA_PHEN_MODE (K5):: =record
".PHENOMENE"

: OJB S NR K16
".ACOUSTIQUE .MODL"

: OJB S NR K16
".ACOUSTIQUE"

: OJB XC V I NO
“.MECANIQUE .MODL”

: OJB S NR K16
“.MECANIQUE”

: OJB XC V I NO
“.THERMIQUE .MODL”

: OJB S NR K16
“.THERMIQUE”

: OJB XC V I NO

.PHENOMENE: S NR K16

This pointer of names contains all the names of phenomenon read in the catalogue:

Today:

.
“MECHANICAL”
.
“THERMAL”
.
“ACOUSTIC”

It is not used to point in a collection.

“.ACOUSTIQUE .MODL”: Names of modelings of the ACOUSTIC phenomenon.
“.MECANIQUE .MODL”: Names of modelings of the MECHANICAL phenomenon.
“.THERMIQUE .MODL”: Names of modelings of the THERMAL phenomenon.

Other objects:

The other objects of the structure of donnéées CATA_PHEN_MODE “are not suffixés” “into hard” in documentation. It is an exception (historical!) with the principles of the tree structure. One creates as many additional objects of phenomena read. These objects have as complete names:

"&CATA. " //nom_de_phenomene

Let us take the example of:

".MECANIQUE"

: OJB XC V I

NO LONG= nb_tm + 2

It is a collection of V (I), named by possible modelings for this phenomenon.
With a given modeling, a vector of entireties V corresponds.

For i_tm of 1 with nb_tm:
V (i_tm): number of type_élément associated the type nets i_tm, for modeling.
If V (i_tm) =0: the type_maille i_tm type_element did not associate for
modeling.
V (nb_tm +1): toplogic dimension of the "principal" elements of modeling: 0/1/2/3
V (nb_tm +2): dimension of physical space bathing modeling: 2/3

Data-processing handbook of Description
D4.04 booklet: -
Document: D4.04.02

Documentation of the sizes of Code_Aster

Summary:

We describe in this document the sizes described in the catalogue of the sizes. The "photo one" was catch on version 8.1.20 (October 2005).

For each size, we give on the 1st line: its scalar type (R/I/C/…) as well as a comment general for this size. Then us listels all its components and we give a comment on each one of them.
Data-processing handbook of Description
D4.04 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Description of the sizes

Date:
01/12/05
Author (S):
J. Key PELLET
:

ABSC_R
Type: R
Curvilinear X-coordinate along a telegraphic grid
ABSC_R
ABSC
curvilinear X-coordinate
ABSC_R
ABSC1
curvilinear X-coordinate of the 1st node of a SEG2
ABSC_R
ABSC2
curvilinear X-coordinate of the 2nd node of a SEG2
ADRSJEVE
Type: I
Size reserved for coded material.
Caution: not to add to him of component
ADRSJEVE
I1
address coded material
ADRSJEVN
Type: I
Size whose CMPS are addresses of objects JEVEUX
ADRSJEVN
I1
address 1st object
ADRSJEVN
I2
address 2nd object
ADRSJEVN
I3
address 3rd object
CAARPO
Type: R
characteristics of the curved beams
CAARPO
RCOURB
radius of curvature
CAARPO
ORIE_ARC

orientation of the arc
CAARPO
C_FLEX
coefficient of flexibility. The moments of inertia I y, Z are divided by
coefficient of flexibility cf I
= I
/Cf
y, Z
y, Z

CAARPO
I_SIGM
MF
1
coefficient of intensification of constraints (Ci)
I
I
Z
=
***
I
with
v
v

C
v
I
CAARPO
C_FLEX_Y
anisotropic coefficient of flexibility in the plan (X, Y)
CAARPO
I_SIGM_Y
anisotropic index of intensification of constraints in the plan (X, Y)
CAARPO
C_FLEX_Z
anisotropic coefficient of flexibility in plan (X, Z)
CAARPO
I_SIGM_Z
anisotropic index of intensification of constraints in plan (X, Z)
CACABL
Type: R
Wire specifications

CACABL
SECT
section of the cable
CACABL
TENS
initial tension
CACOQU
Type: R
Characteristics of the hulls
CACOQU
EP
thickness of the hull
CACOQU
SECT_L
summon sections of reinforcements in direction 1
CACOQU
ALPHA
1st angle of definition of the reference axis in the transverse plan
CACOQU
BETA
2nd angle of definition of the reference axis in the transverse plan
These two angles define compared to the reference mark of reference the vector to
to project on the tangent level of the element in order to define in it the reference mark (T, NR, L).
direction L is perpendicular to the tangent plan
CACOQU
KAPPA
transverse coefficient of shearing
CACOQU
C_METR
metric coefficient of modification for the element hull
CACOQU
X
1ére coordinated of a point of the reference axis
CACOQU
Y
2éme punctual coordinate of the reference axis
CACOQU
Z
3éme punctual coordinate of the reference axis
CACOQU
PAS_T
dimension in the direction T of the rectangular cell of reference
CACOQU

PAS_N
dimension in the direction NR of the rectangular cell of reference
CACOQU
COEF_ECH
coefficient of scale allowing to transform the cell of reference into
real cell of periodicity
CACOQU
K_DRL
local rigidity of rotation according to the direction L applied in a node
CACOQU
K_DRT
local rigidity of rotation according to the direction T applied in a node
CACOQU
K_DRN
local rigidity of rotation according to the direction NR applied in a node
CACOQU
KT_DL
differential rigidity of translation according to the direction L applied to a connection
between two nodes located in the direction T
CACOQU
KT_DT
differential rigidity of translation according to the direction T applied to a connection
between two nodes located in the direction T
CACOQU
KT_DN
differential rigidity of translation according to the direction NR applied to a connection
between two nodes located in the direction T
CACOQU
KT_DRL
differential rigidity of rotation according to the direction L applied to a connection
between two nodes located in the direction T
CACOQU
KT_DRT
differential rigidity of rotation according to the direction T applied to a connection

---

between two nodes located in the direction T
CACOQU
KT_DRN
differential rigidity of rotation according to the direction NR applied to a connection between two nodes located in the direction T
CACOQU
KN_DL
differential rigidity of translation according to the direction L applied to a connection between two nodes located in the direction NR
CACOQU
KN_DT
differential rigidity of translation according to the direction T applied to a connection between two nodes located in the direction NR
CACOQU
KN_DN
differential rigidity of rotation according to the direction NR applied to a connection between two nodes located in the direction NR
CACOQU
KN_DRL
differential rigidity of rotation according to the direction L applied to a connection between two nodes located in the direction NR
CACOQU
KN_DRT
differential rigidity of translation according to the direction T applied to a connection between two nodes located in the direction NR
CACOQU
KN_DRN
differential rigidity of rotation according to the direction NR applied to a connection between two nodes located in the direction NR
CACOQU
ANGL_L
direction of the reinforcements compared to a reference axis belonging to the plan tangent

CACOQU
P_CENT_L
percentages of reinforcements in direction 1
CACOQU
P_CENT_T
percentages of reinforcements in direction 2
CACOQU
DIST_N
offsetting of the tablecloth of reinforcements compared to the mesh support
CACOQU
CTOR
multiplicative constant of rigidity in rotation associated with the direction perpendicular in the tangent plan of the element
CADISA
Type: R
Stamp damping of the discrete elements 12 X 12 [U4.24.01]
CADISA
A1
coefficient (1, 1) of the matrix
CADISA
A2
coefficient (1, 2) of the matrix
CADISA
A3
coefficient (2, 2) of the matrix
CADISA
A77
coefficient (11,12) of the matrix
CADISA
A78
coefficient (12, 12) of the matrix
CADISA
REPA
index of the type of reference mark (=1 total reference mark, =2 local reference mark)
CADISK
Type: R
Stamp rigidity of the discrete elements 12 X 12 [U4.24.01]
CADISK
K1
coefficient (1, 1) of the matrix
CADISK
K2
coefficient (1, 2) of the matrix

CADISK
K3
coefficient (2, 2) of the matrix
CADISK
K77
coefficient (11,12) of the matrix
CADISK
K78
coefficient (12, 12) of the matrix
CADISK
REPK
index of the type of reference mark (=1 total reference mark, =2 local reference mark)
CADISK
ETA
damping coefficient hysteretic
CADISM
Type: R
Stamp of mass of the discrete elements 12 X 12
CADISM
M1
coefficient (1, 1) of the matrix
CADISM
M2
coefficient (1, 2) of the matrix
CADISM
M3
coefficient (2, 2) of the matrix
CADISM
M77
coefficient (11,12) of the matrix
CADISM
M78
coefficient (12, 12) of the matrix
CADISM
REPM
index of the type of reference mark (=1 total reference mark, =2 local reference mark)
CAGEBA
Type: R
Geometrical characteristics of the bars with rectangular or circular section
(cf [U4.24.01])
CAGEBA
HY1
dimension according to GY of the rectangle (node 1)

CAGEBA
HZ1
dimension according to GZ of the rectangle (node 1)
CAGEBA
EPY1
thickness according to GY of a hollow rectangle (node 1)
CAGEBA
EPZ1
thickness according to GZ of a hollow rectangle (node 1)
CAGEBA
HY2
dimension according to GY of the rectangle (node 2)
CAGEBA
HZ2
dimension according to GZ of the rectangle (node 2)
CAGEBA
EPY2
thickness according to GY of a hollow rectangle (node 2)
CAGEBA
EPZ2
thickness according to GZ of a hollow rectangle (node 2)
CAGEBA
R1
radius of the circle (node 1)
CAGEBA
EP1
thickness of the hollow circle (node 1)
CAGEBA
R2
radius of the circle (node 2)
CAGEBA
EP2
thickness of the hollow circle (node 2)

Date:
01/12/05
Author (S):
J. Key PELLET
:
D4.04.02-B Page
: 4/16

CAGEBA
TSEC
type of the section (general, right-angled, circle)
CAGEPO
Type: R
Geometrical characteristics of the beams with rectangular or circular section
(cf [U4.24.01])
CAGEPO
HY1
dimension according to GY of the rectangle (node 1)
CAGEPO
HZ1
dimension according to GZ of the rectangle (node 1)
CAGEPO
EPY1
thickness according to GY of a hollow rectangle (node 1)
CAGEPO
EPZ1
thickness according to GZ of a hollow rectangle (node 1)
CAGEPO
HY2
dimension according to GY of the rectangle (node 2)
CAGEPO
HZ2
dimension according to GZ of the rectangle (node 2)
CAGEPO
EPY2
thickness according to GY of a hollow rectangle (node 2)
CAGEPO
EPZ2
thickness according to GZ of a hollow rectangle (node 2)
CAGEPO
R1
radius of the circle (node 1)

CAGEPO
EP1
thickness of the hollow circle (node 1)
CAGEPO
R2
radius of the circle (node 2)
CAGEPO
EP2
thickness of the hollow circle (node 2)
CAGEPO
TSEC
type of the section (general, right-angled, circle)
CAGE_R
Type: R
Geometrical characteristics
CAGE_R
CAGE
Geometrical characteristic
CAGNBA
Type: R
Geometrical characteristics of a section of bar (cf [U4.24.01])
CAGNBA
A1
surface of the cross section
CAGNPO
Type: R
Geometrical characteristics of a section of beam (cf [U4.24.01])
CAGNPO
A1
surface of the section (node 1)
CAGNPO
IY1
principal moment of inertia compared to GZ (node 1)
CAGNPO
IZ1
principal moment of inertia compared to GY (node 1)
CAGNPO
AY1
coefficient of shearing in the direction Gy (node 1)
CAGNPO
AZ1
coefficient of shearing in the direction Gz (node 1)
CAGNPO

EY1
eccentricity of the center of torsion, component of CG following GY (node 1)
CAGNPO
EZ1
eccentricity of the center of torsion, component of CG following GZ (node 1)
CAGNPO
JX1
constant of torsion (node 1)
CAGNPO
RY1
distance from an external fibre measured according to Y (node 1)
CAGNPO
RZ1
distance from an external fibre measured according to Z (node 1)
CAGNPO
RT1
effective ray of torsion (node 1)
CAGNPO
AI1
surface of the interior section, case of the tubes for example (node 1)
= 0 for a full section
CAGNPO
JG1
constant of warping (node 1)
CAGNPO
IYR21
y (y2 z2
+

) ds (node 1)
S
CAGNPO
IZR21
Z (y2 z2
+

) ds (node 1)
S
CAGNPO
A2
surface of the section (node 2)
CAGNPO
IY2

principal moment of inertia compared to GZ (node 2)
CAGNPO
IZ2
principal moment of inertia compared to GY (node 2)
CAGNPO
AY2
coefficient of shearing in the direction Gy (node 2)
CAGNPO
AZ2
coefficient of shearing in the direction Gz (node 2)
CAGNPO
EY2
eccentricity of the center of torsion, component of CG following GY (node 2)
CAGNPO
EZ2
eccentricity of the center of torsion, component of CG following GZ (node 2)
CAGNPO
JX2
constant of torsion (node 2)
CAGNPO
RY2
distance from an external fibre measured according to Y (node 2)
CAGNPO
RZ2
distance from an external fibre measured according to Z (node 2)
CAGNPO
RT2
effective ray of torsion (node 2)
CAGNPO
AI2
surface of the interior section, case of the tubes for example (node 2)
= 0 for a full section
CAGNPO
JG2
constant of warping (node 2)
CAGNPO
IYR22
y (y2 z2
+

) ds (node 2)
S
CAGNPO

IZR22
Z (y2 z2
+

) ds (node 2)
S

CAGNPO
TVAR
type of section (constant, variable)
CAMASS
Type: R
Geometrical characteristics of the solid elements
CAMASS
C
index of definition of the reference mark of orthotropism (=1 definition by 3 angles nautical, = -1 definition by an axis and a point on this axis)
CAMASS
ALPHA
1st nautical angle
CAMASS
BETA
2nd nautical angle
CAMASS
KAPPA

3rd nautical angle
CAMASS
X
no one if C=1, if not 1st co-ordinate of the point of the axis
CAMASS
Y
no one if C=1, if not 2ère coordinated point of the axis
CAMASS
Z
no one if C=1, if not 3ère coordinated point of the axis
CAORIE
Type: R
Orientation of a segment in 3D. Nautical angles [U4.24.01]
CAORIE
ALPHA
1st nautical angle
CAORIE
BETA
2nd nautical angle
CAORIE
GAMMA
3rd nautical angle
CAPOUF
Type: R
Geometrical characteristics of the elements fluid beam
CAPOUF
B_T
transverse correct term
CAPOUF
B_N
normal correct term
CAPOUF
B_TN
correct term couples
CAPOUF
A_FLUI
fluid surface associated at the end corrector
CAPOUF
A_CELL
surface of the cell of reference
CAPOUF
COEF_ECH
enlarging compared to the real period of the cell of reference

CARCRI
Type: R
Criterion of convergence of a material non-linear problem (for 1 point of Gauss)
CARCRI
ITECREL
numbers max of iteration (ITER_INTE_MAXI)
CARCRI
MACOMP
type of Jacobien (speed or incremental) (TYPE_MATR_COMP)
CARCRI
RESCREL
tolerance for convergence (RESI_INTE_RELA)
CARCRI
THETA
parameter of the theta_méthode for THM
CARCRI
ITEDEC
indicator for recutting (ITER_INTE_PAS)
CARCRI
INTLOC
algorithm of resolution (IMPLICIT or RUNGE_KUTTA_2)
CASECT
Type: K8
Name of an object of the cara-beam type containing the characteristics of a section of beam
CASECT
NAME
name of a cara-beam object
CHLI_R
Type: R
Charge limit
CHLI_R
CHLI1
1st elementary term
CHLI_R
CHLI2
2nd elementary term
CHLI_R
CHLI3
3éme elementary term
CHLI_R
CHAR0

elementary term of with the permanent loading
COEH_F
Type: K8
Coefficient of heat exchange (function) = H (T ext.
*
- T)
COEH_F
H
coefficient of exchange (continuous problem)
COEH_F
H_INF
coefficient of exchange on the lower face of a hull
COEH_F
H_SUP
coefficient of exchange on the higher face of a hull
COEH_R
Type: R
Coefficient of heat exchange (real) = H (T ext.
*
- T)
COEH_R
H
coefficient of exchange (continuous problem)
COEH_R
H_INF
coefficient of exchange on the lower face of a hull
COEH_R
H_SUP
coefficient of exchange on the higher face of a hull
COEH_R
H_LAM
conductivity of the medium located between the two walls
COEH_R
H_JEU
coefficient of roughness
COEH_R
H_PRES
contact pressure (not currently used)
COEH_R
TYPE_H
flag (if =0 then H=cste, if =1 then H variable)
COMPOR
Type: K16

Material behavior
COMPOR
RELCOM
relation of behavior: “ELAS”, “VMIS_ISOT_LINE”,…
COMPOR
NBVARI
a number of variables intern for the behavior
COMPOR
DEFORM
type of deformations: “SMALL”,…
COMPOR
INCELA
great displacement (E. Lorentz)? ? ?
COMPOR
LGPG
length of a point of Gauss (qnté which allows IMPR_RESU the writing pts

---

of Gauss without mixing them)
COMPOR
NBCOUCH
A number of layers
DBEL_R
Type: R
Acoustic decibel
DBEL_R

DB
decibel
DDLI_C
Type: C
Value (complex) imposed on 1 ddl (or to 1 linear relation): I
I U = C
I
DDLI_C
C
Second member of the type complexes relation

DDLI_F
Type: K8
Value (function) imposed on 1 ddl (or to 1 linear relation): I
I U = C
I
DDLI_F
C
Second member of the function type of the relation
DDLI_R
Type: R
Value (real) imposed on 1 ddl (or to 1 linear relation): I
I U = C
I
DDLI_R
C
Second member of the real type of the relation
DDLM_C
Type: C
Coefficients (complex) for a node of a linear relation:
mechanics: A1.UX + A2.UY +… + B3.DRZ = C
DDLM_C
A1
coefficient complexes for the ddl Ux
DDLM_C
A2
coefficient complexes for the ddl Uy
DDLM_C
A3
coefficient complexes for the ddl Uz
DDLM_C
B1
coefficient complexes for the ddl DRx

DDLM_C
B2
coefficient complexes for the ddl Dry
DDLM_C
B3
coefficient complexes for the ddl DRz
DDLM_R
Type: R
Coefficients (real) for a node of a linear relation:
mechanics: A1.UX + A2.UY +… + B3. DRZ = C
thermics: A1.T + A2 T
+ A3 T
= C
inf
sup
for the elements of hull
thermics
DDLM_R
A1
Mechanics: real coefficient for ddl UX
Thermics: real coefficient for the ddl Tmoyen
DDLM_R
A2
Mechanics: real coefficient for ddl UY
Thermics: real coefficient for the ddl Tinf
DDLM_R
A3
Mechanics: real coefficient for ddl UZ
Thermics: real coefficient for the ddl Tsup
DDLM_R
B1
Mechanics: real coefficient for ddl DRX
DDLM_R
B2
Mechanics: real coefficient for ddl DRY
DDLM_R
B3
Mechanics: real coefficient for ddl DRZ
DEPL_C
Type: C
Displacement (complex)
DEPL_C
DX

translation according to OX
DEPL_C
DY
translation according to OY
DEPL_C
DZ
translation according to OZ
DEPL_C
DRX
rotation around OX
DEPL_C
DRY
rotation around OY
DEPL_C
DRZ
rotation around OZ
DEPL_C
GRX
warping (for an element of beam)
DEPL_C
NEAR
ddl of pressure
DEPL_C
TEMP
ddl of temperature
DEPL_C
PHI
angle of cracking
DEPL_C
DH
hydraulic diameter
DEPL_C
LAGR
parameter of lagrange of to the dualisation of the boundary conditions
DEPL_F
Type: K8
Displacement (function)
DEPL_F
DX
translation according to OX
DEPL_F
DY
translation according to OY

DEPL_F
DZ
translation according to OZ
DEPL_F
DRX
rotation around OX
DEPL_F
DRY
rotation around OY
DEPL_F
DRZ
rotation around OZ

DEPL_F
GRX
warping (for an element of beam)
DEPL_F
NEAR
ddl of pressure
DEPL_F
TEMP
ddl of temperature
DEPL_F
PHI
angle of cracking

DEPL_F
DH
hydraulic diameter
DEPL_F
LAGR
parameter of lagrange of to the dualisation of the boundary conditions
DEPL_R
Type: R
Displacement (unknown factor for the mechanical phenomenon)
DEPL_R
DX
translation according to OX
DEPL_R
DY
translation according to OY
DEPL_R
DZ
translation according to OZ
DEPL_R
DRX
rotation around OX
DEPL_R
DRY
rotation around OY
DEPL_R
DRZ
rotation around OZ
DEPL_R
GRX
warping (for an element of beam)
DEPL_R
NEAR
ddl of pressure
DEPL_R
TEMP
ddl of temperature
DEPL_R
PHI
angle of cracking
DEPL_R
DH
hydraulic diameter
DEPL_R

LAGR
parameter of Lagrange of to the dualisation of the boundary conditions
DEPL_R
for X-FEM: DCX, DCY, DCZ: TRADITIONAL DDLS X-FEM
DEPL_R
for X-FEM: H1X, H1Y, H1Z: DDLS HEAVYSIDE X-FEM
DEPL_R
for X-FEM: E1X, E1Y, E1Z: DDLS NOUVEAU RICHES (CRACKTIP1) X-FEM
E2X, E2Y, E2Z: DDLS NOUVEAU RICHES (CRACKTIP2) X-FEM
E3X, E3Y, E3Z: DDLS NOUVEAU RICHES (CRACKTIP3) X-FEM
E4X, E4Y, E4Z: DDLS NOUVEAU RICHES (CRACKTIP4) X-FEM
DOMMAG
Type: R
field of damage on a structure
DOMMAG
DOMA
value of the damage
DURT_R
Type: R
Initialization of the calculation of hardness associated with the metallurgy
DURT_R
HV
value
ENER_R
Type: R
Energy
ENER_R
TOTAL
total energy of the element
ENER_R
TRAC_COM
energy in traction and compression
ENER_R
TORSION
energy in torsion
ENER_R
MEMBRANE
energy out of membrane
ENER_R
INFLECTION
energy in inflection
ENER_R
FLEX_Y

energy in inflection Y
ENER_R
FLEX_Z
energy in inflection Z
ENER_R
PLAN_XY
energy in plan XY
ENER_R
PLAN_XZ
energy in plan XZ
ENER_R
DX
energy according to DX
ENER_R
DY
energy according to DY
ENER_R
DZ
energy according to DZ
ENER_R
DRX
energy according to DRX
ENER_R
DRY
energy according to DRY
ENER_R
DRZ
energy according to DRZ
EPSI_F
Type: K8
Deformation (function)
EPSI_F
EPXX
xx
EPSI_F
EPYY
yy
EPSI_F
EPZZ
zz
EPSI_F
EPXY
xy

EPSI_F
EPXZ
xz
EPSI_F
EPYZ
yz
EPSI_R
Type: R
Deformation
EPSI_R
EPXX
xx deformation of a continuous medium

EPSI_R
EPYY
yy deformation of a continuous medium
EPSI_R
EPZZ
zz deformation of a continuous medium
EPSI_R
EPXY
xy deformation of a continuous medium
EPSI_R
EPXZ
xz deformation of a continuous medium

EPSI_R
EPYZ
yz deformation of a continuous medium
EPSI_R
EXX
hull: generalized deformations
EPSI_R
EYY
hull: generalized deformations
EPSI_R
EXY
hull: generalized deformations
EPSI_R
KXX
hull: generalized deformations
EPSI_R
KYY
hull: generalized deformations
EPSI_R
KXY
hull: generalized deformations
EPSI_R
GAX
hull: generalized deformations
EPSI_R
GAY
hull: generalized deformations
EPSI_R
EPX
Beam: elongation according to the axis of the beam
EPSI_R
KY
Beam: curve according to the axis Y
EPSI_R
KZ
Beam: curve according to axis Z
EPSI_R
INVA_2
second invariant of the tensor of deformation
EPSI_R
PRIN_1
principal deformation of the tensor direction 1
EPSI_R

PRIN_2
principal deformation of the tensor direction 1
EPSI_R
PRIN_3
principal deformation of the tensor direction 1
EPSI_R
INVA_2SG
second signed invariant of the tensor of deformation
EPSI_R
VECT_1_X
Component according to OX of the 1st principal vector ()
EPSI_R
VECT_1_Y
Component according to OY of the 1st principal vector ()
EPSI_R
...
...
EPSI_R
VECT_3_Z
Component according to OZ of the 3rd principal vector ()
ERROR
Type: R
Error analysis with the method of the residues
ERROR
ERREST
absolute error estimated on the element
ERROR
NUEST
relative error estimated on the element
ERROR
SIGCAL
energy of the constraints normalizes on the element
FELECR
Type: R
Application of the force of Laplace (FORCE_ELEC)
FELECR
X1
co-ordinate X of a point of driver 1
FELECR
Y1
co-ordinate Y of a point of driver 1
FELECR
Z1

co-ordinate Z of a point of driver 1
FELECR
X2
co-ordinate X of a point of driver 2
FELECR
Y2
co-ordinate Y of a point of driver 2
FELECR
Z2
co-ordinate Z of a point of driver 2
FELECR
CODE
= 10: components of the force Laplace (X1, Y1, Z1 and X2=Y2=Y3=0)
= 11: TRANS
= 12: DIST
= 2: INFI
= 3: FINISHED
FISS_R
Type: R
Node of the bottom of crack and its normal (calculation of K1, K2 in 2D)
FISS_R
TESTSTEMXÀ
co-ordinate 1 of the node of the bottom of crack
FISS_R
YA
co-ordinate 2 of the node of the bottom of crack
FISS_R
XNORM
component 1 of the normal to the crack
FISS_R
YNORM
component 2 of the normal to the crack
FLAPLA
Type: K24
Mutual inductance between 2 telegraphic circuits
FLAPLA
NOMAIL
name of the grid
FLAPLA
NOGEOM
field of geometry
FLUN_F
Type: K8

Heat flow leaving per unit of area = (- T) .n (it is a scalar)
FLUN_F
FLUN
value of outgoing flow
FLUN_F
FLUN_INF
value of outgoing flow by the lower face of a hull
FLUN_F
FLUN_SUP
value of outgoing flow by the higher face of a hull
FLUN_R
Type: R
Heat flow leaving per unit of area = (- T) .n (it is a scalar)

FLUN_R
FLUN
value of outgoing flow
FLUN_R
FLUN_INF
value of outgoing flow by the lower face of a hull
FLUN_R
FLUN_SUP
value of outgoing flow by the higher face of a hull
FLUX_F
Type: K8

Vectorial flow of heat in a material point of the continuous field: = - T
FLUX_F
FLOW
following component OX of
FLUX_F
FLUY
following component OY of
FLUX_F
FLUZ
following component OZ of
FLUX_R
Type: R
Vectorial flow of heat in a material point of the continuous field: = - T
FLUX_R
FLOW
following component OX of
FLUX_R
FLUY
following component OY of
FLUX_R
FLUZ
following component OZ of
FLUX_R
FLUX_SUP
flow on a point of the higher face of the hulls
FLUX_R
FLUY_SUP
flow on a point of the higher face of the hulls
FLUX_R
FLUZ_SUP
flow on a point of the higher face of the hulls
FLUX_R
FLUX_INF
flow on a point of the lower face of the hulls
FLUX_R
FLUY_INF
flow on a point of the lower face of the hulls
FLUX_R
FLUZ_INF
flow on a point of the lower face of the hulls
FORC_C
Type: C
Complex force (specific, linear, surface or voluminal) applied to

a mechanical model
FORC_C
FX
component according to OX of the force
FORC_C
FY
component according to OY of the force
FORC_C
FZ
component according to OZ of the force
FORC_C
MX
moment following OX
FORC_C
MY
moment following OY
FORC_C
MZ
moment following OZ
FORC_C
BX
Bi-moment for the elements of beam with warping
FORC_C
REFERENCE MARK
indicator of reference mark: if REFERENCE MARK = reference mark OXYZ is the total reference mark
if not it reference mark is the local reference mark related to the element
FORC_C
ALPHA
nautical angles
FORC_C
BETA
nautical angles
FORC_C
GAMMA
nautical angles
FORC_F
Type: K8
Force (specific, linear, surface or voluminal) applied to a model
mechanics
FORC_F
FX
component according to OX of the force

FORC_F
FY
component according to OY of the force
FORC_F
FZ
component according to OZ of the force
FORC_F
MX
moment following OX
FORC_F
MY
moment following OY
FORC_F
MZ
moment following OZ
FORC_F
BX
Bi-moment for the elements of beam with warping
FORC_F
REFERENCE MARK
indicator of reference mark: if REFERENCE MARK = reference mark OXYZ is the total reference mark
if not it reference mark is the local reference mark related to the element
FORC_F
ALPHA
nautical angles
FORC_F
BETA
nautical angles
FORC_F
GAMMA
nautical angles
FORC_R
Type: R
Force (specific, linear, surface or voluminal) applied to a model
mechanics
FORC_R
FX
component according to OX of the force
FORC_R
FY
component according to OY of the force
FORC_R

FZ
component according to OZ of the force
FORC_R
MX
moment following OX
FORC_R
MY
moment following OY
FORC_R
MZ
moment following OZ
FORC_R
BX
Bi-moment for the elements of beam with warping
Data-processing handbook of Description
D4.04 booklet: -
HT-66/05/003/A

FORC_R
REFERENCE MARK
indicator of reference mark: if REFERENCE MARK = reference mark OXYZ is the total reference mark if not it reference mark is the local reference mark related to the element
FORC_R
ALPHA
nautical angles
FORC_R
BETA
nautical angles
FORC_R
GAMMA
nautical angles
FREQ_R
Type: R
Frequency (real)
FREQ_R
FREQ
value of the frequency
FTHM_F
Type: K8
Thermohydraulic flow (modeling THM)
FTHM_F
PTHER
function of the heat flow
FTHM_F
PFLU
hydrailic function of flow

FTHM_R
Type: R
Thermohydraulic flow (modeling THM)
FTHM_R
PTHER
value of the heat flow
FTHM_R
PFLU
hydrailic value of flow
G
Type: R
Rate of refund of energy and coefficients of intensity of constraints
G
GTHETA
Rate of refund of energy
G
FIC1
? ? ?
G
FIC2
? ? ?
G
K1
Coefficient of K1 constraints
G
K2
Coefficient of K2 constraints
GEOM_R
Type: R
Geometry (of a node)
GEOM_R
X
co-ordinate according to OX
GEOM_R
Y
co-ordinate according to OY
GEOM_R
Z
co-ordinate according to OZ (0. If the model is 2D)
HARMON
Type: I
Harmonic of Fourier
HARMON

NH
number of harmonic of Fourier
IMPE_C
Type: C
Impedance
IMPE_C
IMPE
value complexes impedance
IMPE_F
Type: K8
Impedance
IMPE_F
IMPE
function of the impedance
IMPE_R
Type: R
Impedance
IMPE_R
IMPE
actual value of the impedance
INDIC
Type: R
The chart of the sizes of the new elements for a mending of meshes contains
INDIC
HNEW
= hold
p
/ /
1 where p is the degree of the polynomials of the element
INDIC
HOLD
sup of the 3 medians of the element
INDIC
X
co-ordinates of? ? ?
INDIC
Y
co-ordinates of? ? ?
INDIC
Z
co-ordinates of? ? ?
INDIC
BID

? ? ?
INDIC
KSI
relative error/precision if 1: one refines
INST_R
Type: R
Moment of calculation for a temporal evolution
INST_R
INST
value of time (urgent)
INST_R
DELTAT
increment of time for a calculation by `'not of time ''
INST_R
THETA
parameter of - method (0: clarify, 1: implicit,] 0 [
1
: semi-implicit)
INST_R
KHI
indicator for stationary or transitory calculation
INST_R
R
parameter of Lagrangian increased
INST_R
RHO
parameter of Lagrangian increased
LISTMA
Type: K16
Definition of the distributed loads on a zone of the grid (Force of LAPLACE)
LISTMA
LISTMA
List meshs affected by the loading
LISTMA
TRANS
Name of a vector of work describing the loading
MASS_R
Type: R
Characteristics
MASS_R
M
mass element
MASS_R

CDGX
co-ordinate according to X of the C.D.G
MASS_R
CDGY
co-ordinate following Y
MASS_R
CDGZ
co-ordinate according to Z
Data-processing handbook of Description
D4.04 booklet: -
HT-66/05/003/A

MASS_R
IXX
inertia y2

MASS_R
IYY
inertia x2

MASS_R
IZZ
inertia x2
y2
+ polar inertia
MASS_R
IXY
inertia xy

MASS_R
IXZ
inertia xz

MASS_R
IYZ
inertia yz

MASS_R
EX
moment of inertia principal X
MASS_R
EY
moment of inertia principal Y
MASS_R
EZ
moment of inertia principal Z
MATE_F
Type: K8
Material
MATE_F
MATT
name of affected material to a mesh
META_R
Type: R
Metallurgical sizes provided by user (AFFE_CARTE)
META_R
ZF
Proportion of ferrite
META_R
ZP
Proportion of pearlite
META_R
ZB
Proportion of bainite
META_R
ZM
Proportion of martensite
META_R
Ms
Temperature to describe the martensitic transformation
META_R
TPG

Temperature at the point of gauss
META_R
P
Cut austenitic grain
NEUT_K24
Type: K24
Neutral size `'' of the K24 type. This size is used in general for
to transmit names of objects JEVEUX. The significance of the components varies
from one option to another. This size `'pass key '' is only used to avoid
the introduction of many sizes without much interest.
NEUT_K24
Z1
component 1
NEUT_K24
Z2
component 2
NEUT_K24
Z3
component 3
NEUT_K24
Z4
component 4
NEUT_K24
Z5
component 5
NEUT_R
Type: R
Neutral size `'' of the real type (see NEUT_K24)
NEUT_R
X1
component 1
NEUT_R
X2
component 2
NEUT_R
X3
component 3
NEUT_R
X4
component 4
NEUT_R
X5
component 5

NEUT_R
X6
component 6
NEUT_R
X7
component 7
NEUT_R
X8
component 8
NEUT_R
X9
component 9
NEUT_R
X10
component 10
NEUT_R
X11
component 11
NEUT_R
X12
component 12
NUMC_I
Type: I
For the multi-layer hulls
NUMC_I
NUMC
Number of the layer
NUMC_I
ORDO
Level of layer (SUP=1, MOY=0, INF=-1)
ONDE_F
Type: K8
Term of damping in incidental wave imposed on a face of element
ONDE_F
NEAR
Value of the wave of the function type
ONDE_R
Type: R
Term of damping in incidental wave imposed on a face of element
ONDE_R
NEAR
Value of the wave of the real type
PESA_R

Type: R
Characteristics of a loading of gravity: intensity and direction
PESA_R
G
value of the acceleration of gravity
PESA_R
AG
component according to OX of the vector giving the direction of gravity

---

PESA_R
BG
component according to OY of the vector giving the direction of gravity
PESA_R
CG
component according to OZ of the vector giving the direction of gravity
POSI
Type: I
Option of calculation of the matrix of Mass
POSI
POS
= 1, option MASS_MECA
= 0, option MASS_MECA_DIAG
PREC
Type: R
precision

PREC
PREC
value of the precision
PRES_C
Type: C
.
Surface loading applied except for a mechanical model (, CISA)
.
Unknown factor of a problem of accoustics: (pressure, speed of the fluid)
PRES_C
NEAR
value of the pressure
PRES_C
CISA
shearing applied to the edge of a model 2D
PRES_C
VX
speed of the fluid following OX
PRES_C
VY
speed of the fluid following OY
PRES_C
VZ
speed of the fluid following OZ
PRES_C
LAGR
parameter of lagrange of to the dualisation of the boundary conditions
PRES_F
Type: K8
.
Surface loading applied except for a mechanical model (, CISA)
.
Unknown factor of a problem of accoustics: (pressure, speed of the fluid)
PRES_F
NEAR
value of the pressure
PRES_F
CISA
shearing applied to the edge of a model 2D
PRES_R
Type: R
.
Surface loading applied except for a mechanical model (, CISA)

.

Unknown factor of a problem of accoustics: (pressure, speed of the fluid)
PRES_R
NEAR
value of the pressure
PRES_R
CISA
shearing applied to the edge of a model 2D
PRES_R
VX
speed of the fluid following OX
PRES_R
VY
speed of the fluid following OY
PRES_R
VZ
speed of the fluid following OZ
PRES_R
LAGR
parameter of lagrange of to the dualisation of the boundary conditions
RICE_TRACEY
Type: R
Cham_elem of RICE_TRACEY
RICE_TRACEY
TRIAX
rate of triaxiality on the mesh
RICE_TRACEY
RSRO
growth rate on the mesh at the current moment
RICE_TRACEY
VOLU
volume taken into account
RICE_TRACEY
NUMEMA
number of the mesh
RICE_TRACEY
DEPSEQ
variation of the equivalent plastic deformation
ROTA_R
Type: R
Characteristics of a loading of rotation: speed and axis
ROTA_R
OME

speed ongulaire
ROTA_R
AR
component (according to OX) of the axis of rotation
ROTA_R
Br
component (according to OY) of the axis of rotation
ROTA_R
CR
component (according to OZ) of the axis of rotation
SECTION
Type: R
Characteristics of a section of beam
SECTION
SURFACE
surface of the section
SECTION
XG
X-coordinate of the centre of gravity (reference mark OXY of the grid of the section)
SECTION
YG
ordinate of the centre of gravity (reference mark OXY of the grid of the section)
SECTION
ALPHA
angle (OX, GX) if GX 1st principal axis of inertia
SECTION
IX
2
`'inertia '' around GX ((Y - YG)
S
)
SECTION
IY

`'inertia '' around G Y (X - XG)
S

2
SECTION
XC
X-coordinate of the center of torsion (reference mark GXY of inertia)

SECTION
YC
ordinate of the center of torsion (reference mark GXY of inertia)
SECTION
CF
? ? ?
SIEF_C
Type: C
State of stress (or of effort interns)
SIEF_C
SIXX
xx constraints in a continuous medium
SIEF_C
SIYY
yy forced in a continuous medium
SIEF_C
SIZZ
zz forced in a continuous medium

SIEF_C
SIXY
xy forced in a continuous medium
SIEF_C
SIXZ
xz forced in a continuous medium

SIEF_C
SIYZ
yz forced in a continuous medium
SIEF_C
NR
normal effort
SIEF_C
VY
shearing action according to CY efforts intern beams
SIEF_C
VZ
shearing action according to C Z efforts intern beams
SIEF_C
MT
torque according to C X
SIEF_C
MFY
bending moment according to GY
SIEF_C
MFZ
bending moment according to G Z
SIEF_C
BX
Bi-moment (beam with warping)
SIEF_C
NXX
efforts intern hulls
SIEF_C
NYY
efforts intern hulls
SIEF_C
NXY
efforts intern hulls
SIEF_C
MXX
efforts intern hulls
SIEF_C
MYY
efforts intern hulls
SIEF_C
MXY
efforts intern hulls
SIEF_C

QX
efforts intern hulls
SIEF_C
QY
efforts intern hulls
SIEF_C
FX
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_C
FY
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_C
FZ
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_C
MX
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_C
MY
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_C
MZ
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_C
SIGN
elements of contact: normal constraint
SIEF_C
SITX
elements of contact: tangential constraint
SIEF_C
SITY
elements of contact: tangential constraint
SIEF_C
SITZ
elements of contact: tangential constraint
SIEF_C
VMIS
constraint of Von Mises
SIEF_C
TRESCA
tresca
SIEF_C
PRIN_1

constraint principal direction 1
SIEF_C
PRIN_2
constraint principal direction 2
SIEF_C
PRIN_3
constraint principal direction 3
SIEF_C
VMIS_SG
constraint of Von Mises signed by the trace of sigma
SIEF_C
SN
constraint in the section of beam due to the normal effort
SIEF_C
SVY
constraint in the section of beam due to the shearing action VY
SIEF_C
SVZ
constraint in the section of beam due to the shearing action VZ
SIEF_C
SMT
constraint in the section of beam due to the torque MX
SIEF_C
SMFY
constraint in the section of beam due to the moment D inflection MY
SIEF_C
SMFZ
constraint in the section of beam due to the bending moment MZ
SIEF_R
Type: R
State of stress (or of effort interns) (cf [U?. ? ?. ? ?])
SIEF_R
SIXX
xx constraints in a continuous medium
SIEF_R
SIYY
yy forced in a continuous medium
SIEF_R
SIZZ
zz forced in a continuous medium
SIEF_R
SIXY
xy forced in a continuous medium

SIEF_R
SIXZ
xz forced in a continuous medium
SIEF_R
SIYZ
yz forced in a continuous medium
SIEF_R
NR
normal effort
SIEF_R
VY
shearing action according to CY efforts intern beams
SIEF_R
VZ
shearing action according to C Z efforts intern beams
SIEF_R
MT
torque according to C X
SIEF_R
MFY
bending moment according to GY
SIEF_R
MFZ
bending moment according to G Z

SIEF_R
BX
Bi-moment (beam with warping)
SIEF_R
NXX
efforts intern hulls
SIEF_R
NYY
efforts intern hulls
SIEF_R
NXY
efforts intern hulls
SIEF_R
MXX
efforts intern hulls
SIEF_R
MYY
efforts intern hulls
SIEF_R
MXY
efforts intern hulls
SIEF_R
QX
efforts intern hulls
SIEF_R
QY
efforts intern hulls
SIEF_R
FX
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_R
FY
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_R
FZ
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_R
MX
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_R
MY
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_R

MZ
efforts for the discrete ones, beams, bars, holds, pulleys
SIEF_R
SIGN
elements of contact: normal constraint
SIEF_R
SITX
elements of contact: tangential constraint
SIEF_R
SITY
elements of contact: tangential constraint
SIEF_R
SITZ
elements of contact: tangential constraint
SIEF_R
VMIS
constraint of Von Mises
SIEF_R
TRESCA
tresca
SIEF_R
PRIN_1
constraint principal direction 1
SIEF_R
PRIN_2
constraint principal direction 2
SIEF_R
PRIN_3
constraint principal direction 3
SIEF_R
VMIS_SG
constraint of Von Mises signed by the trace of sigma
SIEF_R
SN
constraint in the section of beam due to the normal effort
SIEF_R
SVY
constraint in the section of beam due to the shearing action Vy
SIEF_R
SVZ
constraint in the section of beam due to the shearing action Vz
SIEF_R
SMT

constraint in the section of beam due to the torque MX
SIEF_R
SMFY
constraint in the section of beam due to the moment D My inflection
SIEF_R
SMFZ
constraint in the section of beam due to moment Mz the bending
SIEF_R
TRIAX
rate of triaxiality
SIEF_R
SI_ENDO
equivalent constraint of damage
SIEF_R
MASF
*********** component used for the THM ************
SIEF_R
ENTR
to repeat: for the elements `'THM ''
SIEF_R
DISS
to repeat: for the elements `'THM ''
SIEF_R
FLHX
to repeat: for the elements `'THM ''
SIEF_R
FLHY
to repeat: for the elements `'THM ''
SIEF_R
FLHZ
to repeat: for the elements `'THM ''
SIEF_R
FLOW
to repeat: for the elements `'THM ''
SIEF_R
FLUY
to repeat: for the elements `'THM ''
SIEF_R
FLUZ
to repeat: for the elements `'THM ''
SIEF_R
VECT_1_X
Component according to OX of the 1st principal vector ()

SIEF_R
VECT_1_Y
Component according to OY of the 1st principal vector ()
SIEF_R
...
...
SIEF_R
VECT_3_Z
Component according to OZ of the 3rd principal vector ()
SIZZ_R
Type: R
Simple size being used to build elementary sizes MSIZ_R and
VSIZ_R for the calculation of estimator ZZ1
SIZZ_R
SIZZ
coefficient of the matrix or the vector
SOUR_F
Type: K8
Voluminal source of function type
SOUR_F
SOUR
value of the voluminal source applied to a mesh
key word SOURCE of order AFFE_CHAR_THER_F
SOUR_F
VNOR
value the normal speed applied to a face
key word VITE_FACE of order AFFE_CHAR_MECA_F
SOUR_R
Type: R
Voluminal source of real type
SOUR_R
SOUR
value of the voluminal source applied to a mesh
key word SOURCE of order AFFE_CHAR_THER

Description of the sizes

Date:
01/12/05
Author (S):
J.** Key **PELLET
:
D4.04.02-B Page
: 15/16

SOUR_R
VNOR
value the normal speed applied to a face
key word VITE_FACE of order AFFE_CHAR_MECA
SPMA_R
Type: R
Min and max values of a real field + localization. The “min” and “max” are taken on the sequence numbers of the SD_RESULTAT (in general temporal)
SPMA_R
MAX
value max
SPMA_R
MIN
value min
SPMA_R
NCOUMAX
number of the layer where the max is reached
SPMA_R
NCOUMIN
number of the layer where the min is reached
SPMA_R
NSECMAX
number of the sector where is reached the max (pipes)
SPMA_R
NSECMIN
number of the sector where the min is reached (pipes)
SPMA_R
NPCOUMAX
number of the point on the layer where the max is reached
SPMA_R
NPCOUMIN
number of the point on the layer where the min is reached
SPMA_R

NPSECMAX
number of the point in the sector where the max is reached
SPMA_R
NPSECMIN
number of the point in the sector where the min is reached
STAOUDYN
Type: R
Parameters of Newmark if calculation dynamic
STAOUDYN
STAOUDYN
= 0: statics
= 1: dynamics
STAOUDYN
ALFNMK
parameter of Newmark ALPHA
STAOUDYN
DELNMK
parameter of Newmark DELTA
TEMP_C
Type: C
Unknown temperature of the thermal phenomenon
TEMP_C
TEMP
temperature
TEMP_C
TEMP_INF
temperature on the lower face (hulls)
TEMP_C
TEMP_SUP
temperature on the higher face (hulls)
TEMP_C
LAGR
parameter of lagrange of to the dualisation of the boundary conditions
TEMP_F
Type: K8
Unknown temperature of the thermal phenomenon
TEMP_F
TEMP
temperature
TEMP_F
TEMP_INF
temperature on the lower face (hulls)
TEMP_F

TEMP_SUP
temperature on the higher face (hulls)
TEMP_R
Type: R
Unknown temperature of the thermal phenomenon
TEMP_R
TEMP
temperature
TEMP_R
TEMP_INF
temperature on the lower face (hulls)
TEMP_R
TEMP_SUP
temperature on the higher face (hulls)
TEMP_R
LAGR
parameter of lagrange of to the dualisation of the boundary conditions
VAR2_R
Type: R
Components of the CHAM_NO of the variable type internal (passage of one CHAM_ELEM with a CHAM_NO)
VAR2_R
VARI_1
variable interns 1
VAR2_R
VARI_2
variable interns 2
VAR2_R
VARI_3
variable interns 3
VAR2_R
VARI_29
variable interns 29
VAR2_R
VARI_30
variable interns 30
VARI_R
Type: R
Variables intern for the non-linear laws of behavior
Caution: the number of variables interns being able to strongly vary according to models of behavior (from 1 to 50 for example), they are not distinguished components of this size. The real number of components is given in outside catalogues.

VARI_R
VARI
conventional name of the components: VARI_1, VARI_2,…
VNOR_C
Type: C
Normal speed applied to a face of mesh (acoustic)
VNOR_C
VNOR
value normal speed
VNOR_F
Type: K8
Normal speed applied to a face of mesh (acoustic)
VNOR_F
VNOR
function normal speed
NEIGHBOR
Type: I
Elements close and their types for the estimator to error
NEIGHBOR
V0
number of the close element 1
NEIGHBOR
V1
number of the close element 2
NEIGHBOR
V2
number of the close element 3
NEIGHBOR
V3
number of the close element 4
NEIGHBOR
V4
number of the close element 5
NEIGHBOR
V5
number of the close element 6

Titrate:
Description of the sizes

Date:
01/12/05
Author (S):
J.** Key **PELLET
:
D4.04.02-B Page
: 16/16

NEIGHBOR
V6
number of the close element 7
NEIGHBOR
T0
type of the close element 1
NEIGHBOR
T1
type of the close element 2
NEIGHBOR
T2
type of the close element 3
NEIGHBOR
T3
type of the close element 4
NEIGHBOR
T4
type of the close element 5
NEIGHBOR
T5
type of the close element 6
NEIGHBOR
T6
type of the close element 7
WEIBULL
Type: R
Constraint of Weibull
WEIBULL
DSIGWB
value of the constraint

Code_Aster ®
Version
4.0

Titrate:
Structure of Data grid

Date:
05/05/97
Author (S):
J.** Key **PELLET
:
D4.06.01-A Page
: 1/4

Organization (S): EDF/IMA/MMN

Data-processing handbook of Description
D4.06 booklet:
D4.06.01 document

Structure of Data grid

1 General information

A grid is a whole of meshs of the predefined type: HEXA8, TRIA3,…

These meshs are defined by a list of nodes which have co-ordinates. These are the nodes which connect the meshs between them. The whole of the co-ordinates of the nodes of the grid forms one field with the nodes of the size "geometry" (cham_no_GEOM_R).

A grid also contains groups of meshs and groups of named nodes. These groups are unspecified: a mesh (for example) can belong to 0, 1, 2,…, N groups.

Let us announce that for the static under-structuring (and it only), a grid can contain super-meshs (meshs having an unspecified number of nodes).

When the grid is made of elements linéïques, it can contain a CHART containing for each mesh the curvilinear X-coordinate of each node of the mesh.

Basic concepts: node, mesh, groups of meshs (or nodes), grid are more detailed in the document [D3.01.01].

2
Tree structure of the Structure of Data

GRID
(K8)
:: = record

O
“.DIME”:
OJB
S V I
“.NOMNOE”
:
OJB
S NR K8
“.COORDO”
:
CHAM_NO_GEOM_R

F
“.NOMGNO”
:
OJB
S NR K8
“.GROUPENO”
:
OJB
XD
V I NO (“$.NOMGNO”) VARI

% if the grid contains meshs:

I O
“.NOMMAI”
:
OJB
S NR K8
“.TYPMAIL”:
OJB
XC
E I NO (“$.NOMMAI”) IDIOTS

“.CONNEX”
:
OJB
XC
V I NO (“$.NOMMAI”) VARI

F
“.NOMGMA”
:
OJB
S NR K8
“.GROUPEMA”
:
OJB
XD
V I NO (“$.NOMGMA”) VARI

% if the grid contains super-meshs (under-structuring

statics)
:

I O
“.NOMACR”
:
OJB
S V K8
“.PARA_R”
:
OJB
S V R
“.SUPMAIL”:
OJB
XD
V I NO () VARI
F
“.TYPL”:
OJB
S V I

% if the grid contains a chart of abcisse curvilinear:
F
".ABS_CURV"
:
CARTE_ABSC_R

F
"$vide":
TITRATE

3
Contents of objects JEVEUX

.
".DIME": S V I LENGTH = 6
V (1): nb_no: a number of physical nodes of the grid
V (2): nb_nl: a number of nodes of Lagrange of the grid
V (2) > 0 there exist super-meshs: static under-structuring (sss)
V (3): nb_ma: a number of meshs of the grid
V (4): nb_sm: a number of super-meshs of the grid
V (4) > 0 sss
V (5): nb_sm_mx: raising number of super-meshs
V (6): dim_coor: /2 (if grid 2D)
/3 (if grid 3D)

.
".NOMNOE": S NR LONG K8 = nb_no
It is the pointer of names giving the correspondence:
nom_de_noeud numero_de_noeud

.
".NOMGNO": S NR LONG K8 = nb_gno
It is the pointer of names giving the correspondence:
nom_de_group_no numero_de_group_no

Caution:

The number of group_no of a grid can change: one can modify a grid (order DEFI_GROUP) to add group_no to him.

.

“.GROUP_NO”: XD V I NO (“$.NOMGNO”) VARI NB_OJB = nb_gno

That is to say V = “.GROUPENO” (nom_gno)

V (1): number of the 1° node of the group_no of name: nom_gno
V (2): number of the 2° node of the group_no of name: nom_gno
...
V (N): number of the last node of the group_no of name: nom_gno
-
group_no nb_gno = NMAXOC numbers (“.GROUPND”)
-
numbers node N of nom_gno = LONMAX (V)

.
“.NOMMAI”: S V LONG K8 = nb_ma
It is the pointer of names giving the correspondence:
nom_de_maille numero_de_maille

.
“.TYPMAIL”: XC E I NO (“$.NOMMAI”) NB_OJB = nb_ma

That is to say E = “.TYPMAIL” (nom_mail)

E: number of the type of mesh associated with the mesh with name: nom_mail
-

the type of mesh is a name defined in the /compelem/typmail catalogue:
-
SEG2, TRIA3, QUAD4,…, HEXA20
-
the known types of mesh of Aster are described in [U3.01]
- correspondence: nom_de_type_de_maille numero_de_type_de_maille is accessible by the pointer from name: “&CATA.TM.NOMTM” cf [D4.04.01].

.
“.CONNEX”: XC V I NO (“$.NOMMAI”) NB_OJB = nb_ma

That is to say V = “.CONNEX” (nom_mail)

V (1): number of the 1° node of the mesh of name: nom_mail
...
V (N): number of the last node of the mesh of name: nom_mail
-
N = a number of nodes of nom_mail = LONMAX (V)
-
the number of nodes of a mesh is always the number of nodes associated with the type of mesh who is attached to him (see object “&CATA.TM.NBNO” [D4.04.01])

.
“.NOMGMA”: S NR LONG K8 = nb_gma
It is the pointer of names giving the correspondence:
nom_de_group_ma numero_de_group_ma
Caution:

The number of GROUP_MA of a grid can change: one can modify a grid (order DEFI_GROUP) to add group_ma to him.

.
“.GROUPEMA”: XD V I NO (“$.NOMGMA”) VARI NB_OJB = nb_gma

That is to say V = “.GROUPEMA” (nom_gma)

nb_ma_gma = a number of meshs of nom_gma = LONMAX (V)
for I = 1, nb_ma_gma
V (I): number of the ième mesh of nom_gma
- a number
of
group_ma of the grid = NMAXOC (“.GROUPEMA”)

.
“.NOMACR”: S V LONG K8 = nb_sm

for I = 1, nb_sm
V (I): name of the MACR_ELEM_STAT associated with super-mesh I
Data-processing handbook of Description
D4.06 booklet:
HI-75/97/011/A

.
*“.PARA_R”: S V R LENGTH = 14*nb_sm*
for I = 1, nb_sm:
V (14 (i-1) +1)*
:
TX
V (14 (i-1) +2)*
:
TY
V (14 (i-1) +3)*
:
TZ

V (14 (i-1) +4)*
:
alpha
V (14 (i-1) +5)*
:
beta
V (14 (i-1) +6)*
:
gamma
V (14 (i-1) +7)*
:
PX
V (14 (i-1) +8)*
:

PY
V (14 (i-1) +9)*
:
PZ

V (14 (i-1) +13)*
:
dmini
V (14 (i-1) +14)*
:
dmaxi

-
(TX, TY, TZ) are the values of translation of the geometrical transformation associated super-mesh I
- (alpha, beta, gamma) are the nautical angles (in radians) defining rotation geometrical transformation,
-
(PX, PY, PZ) defines the preceding centre of rotation.
That is to say macrost the MACR_ELEM_STAT associated with super-mesh I, the position of super-mesh I is defined by isometry of the nodes of macrost. The isometry is composition in the order: rotation then translation,
-
dmini: minimal distance between 2 nodes of mesh I,
-
dmaxi: maximum distance between 2 nodes of mesh I.

.
".SUPMAIL": XD V I NO () VARI NB_OJB = nb_sm

That is to say V = ".SUPMAIL" (nom_sma)

V is a vector containing the numbers of the nodes of the super-mesh nom_sma the nodes of one super-mesh can be of type "physical" or type "Lagrange".

That is to say:
inop a "physical" number of node of the super-mesh nom_sma
inol a number of node "Lagrange" of the super-mesh nom_sma

1 · inop · nb_no

nb_no + 1 · inol · nb_no + nb_nl

V is the shape of recopy of object ".CONX" of the MACR_ELEM_STAT [D4.08.01].

V defines the connectivity of the super-meshs. The super-meshs "are restuck" by nodes "physical". The nodes of "Lagrange" inherited the MACR_ELEM_STAT are never common to several super-meshs.

-
numbers total nodes ("physics" + "Lagrange") of nom_sma = LONMAX (V)
-
the pointer of names (intern) of object ".SUPMAIL" gives the correspondence:
numero_super_maille nom_super_maille

.
".TYPL": S V I LENGTH = nb_nl
for I = 1, nb_nl
V (I): /1 if the node of "Lagrange" I is of type "before"
/2 if the node of "Lagrange" I is of type "after"

Data-processing handbook of Description
D4.06 booklet: -
Document: D4.06.02

Structures of Data ligrel and model

Summary

Description of the SD ligrel and model.

A model represents the association of the types of finite elements on the meshs of a grid.

A ligrel is a standard list of groups of elements in the same way.

1
Structures of data in some lines

These 2 structures of Data are presented in the document [D0.04.02]

In short:

- ***a ligrel contains a list of finite elements. A finite element being the couple made of one net and of a type of finite element (type_elem),***
- ***the meshs supporting the finite elements can be meshs of the grid or additional meshs (or late),***
- ***a model contains a ligrel; but it can also exist a ligrel in a load [D4.06.04],***
- ***in the ligrel of the model, a mesh of the grid can support only one finite element with more (objects .MAILLE and .REPE),***
- ***in the ligrel of the model, a node of the grid can support only one finite element with more (object .NOEU),***
- ***with a model is associated a PHENOMENON and only one: "MECHANICAL", "THERMAL" or "ACOUSTIC",***
- ***with each PHENOMENON is associated a fundamental size: "DEPL_R", "TEMP_R" or "PRES_C",***
- ***the two preceding limitations make it possible to know which are the degrees of freedom carried by the nodes affected by finite elements (objects. PRNM and. PRNS),***
- ***a model can contain static substructures: "activation" of super-meshs of grid [D4.06.01 §2],***
- ***a ligrel can not contain any finite element. In this case it is a ligrel of model, it model must contain static substructures,***
- ***a ligrel (as a model) is always associated a single grid.***

---

2 Tree structure

ligrel (K19).: =record

".NBNO" OJB
S
E
I
".NOMA" OJB
S
E
K8
indirect (1)
(1)
:
grid

% if the ligrel contains finite elements

".LIEL" OJB
XC
V
I
NAKED ()
".REPE" OJB
S
V
I

% if the ligrel contains late meshs:

".NEMA" OJB
XC
V
I
NAKED ()

% if the ligrel contains late nodes:

".PRNS" OJB
S
V
I
".LGNS" OJB
S
V
I

% if it is a ligrel of model
".PRNM" OJB
S
V
I

model (K8).: =record

“.MODELE”
: ligrel

|
% if the model contains finite elements
“.MAILLE”
OJB
S
V
I
“.NOEUD”
OJB
S
V
I

|
% if the model contains static substructures
'.SSSA
OJB
S
V
I

|
% if the model contains a crack XFEM
'.FISS
OJB
S
V
K8

LONG=1

3
Contents of objects JEVEUX of the ligrel

3.1 “.LIEL”: XC V I NAKED ()

Collection .LIEL contains the numbers of the meshs supporting of the finite elements. The elements are
gathered by groups of elements in the same way standard (GREL) [D0.04.02]. An object of the

collection
corresponds to a GREL.

That is to say I the ième GREL of the LIGREL,

V =.LIEL (I); N = LONG (V) = l+nbre of meshs of the ième GREL

V (1)
number of the mesh associated with the 1st element with the GREL
V (2)
number of the mesh associated with the 2nd element with the GREL
V (n-1)
number of the mesh associated with the last element with the GREL
V (N)
number of the type of element associated with the GREL I
(&CATA.TE.NOMTE object [D4.04.01])

Important conventions:

.
if the mesh is a mesh of the grid its number is stored just as it is.
.
if the mesh is a late mesh, its number is stored with the minus sign (cf object .NEMA),
.

the number of GREL of a ligrel is worth: NUTOC (“.LIEL”)

3.2 “.REPE”: S V I

Either V = “.REPE”

That is to say:
nbma = no. of meshs of the grid associated with the ligrel
*LENGTH (V) = 2*nbma*

for I = 1, nbma
V (2 (i-1) +1): number of the GREL associated with mesh I with the grid
V (2 (i-1) +2): positon in the GREL of mesh I of the grid

This object is I " 'opposite " of object .LIEL concerning the meshs of the grid

If I is a nonaffected mesh: V (2 (i-1) +1) = V (2 (i-1) +2) = 0

3.3 “.NEMA”: XC V I NAKED ()

This collection describes the late meshs of the ligrel. II exists an object of collection by late mesh.

That is to say:
nbmas = a number of late meshs of the ligrel
nbmas = NUTIOC (.NEMA)

for I = 1, nbmas

V = .NEMA (I); n= LONG (V) = (a number of nodes of the mesh) + 1

V (L)
number of the 1st node of mesh I
V (2)
number of the 2nd node of mesh I
V (n-1)
number of the last node of mesh I
V (N)
number of the type of mesh I

If the number of a node is negative, it is the opposite of the number of a late node of the ligrel (cf object .NBNO).

3.4 “.PRNS”: S V I

This object describes the ddls carried by the late nodes of the ligrel.

That is to say:
V = .PRNS; nbnos = a number of late nodes of the ligrel
nec = a number of coded entireties necessary to the fundamental size associated the ligrel
nec = (a number of CMP (fundamental size)/30) +1

for I = 1, nbros
V (nec (i-1) +1)*
: 1st coded entirety of node I
V (nec (i-1) +2)*
: 2nd coded entirety of node I
V (nec (i-1) +nec)*
: last coded entirety of node I

"The small" piece of .PRNS concerning node I is what one calls a Descriptor-Size [D4.06.05].
Handbook of Data-processing Description
D4.06 booklet: - HT-66/05/003/A

3.5 ".LGNS": S V I

This object is dimensioned with the number of late nodes of the ligrel.

V (INO): numbers indicating how the late node of Lagrange INO must be numbered (see SD NUME_DDL).

V (INO): 0
node INO is not a node of Lagrange

V (INO): +1
node INO is a node of Lagrange of the type "1". II must be numbered before the physical ddls that it constrained.
V (INO): - 2
node INO is a node of Lagrange of the type "2". II must be numbered after the physical ddls that it constrained.
V (INO): - 1
node INO is a node of Lagrange of the type "1". II must be numbered after the physical ddls that it constrained.
V (INO): +2
node INO is a node of Lagrange of the type "2". II must be numbered before the physical ddls that it constrained.

3.6 ".NBNO": S.E.I

The number of late nodes of the ligrel contains

3.7 ".NOMA": S.E.K8

The name of the grid associated with the ligrel contains

3.8 ".PRNM": S V I

This object describes the ddls carried by the nodes of the grid.

That is to say:
V = .PRNM; nbno = a number of nodes of the grid
nec = a number of coded entireties necessary to the fundamental size associated the ligrel
nec= (a number of CMP (fundamental size)/30) +1

for I = 1, nbro
V (nec (i-1) +1)*
:
1st coded entirety of node I
V (nec (i-1) +2)*
:
2nd coded entirety of node I
V (nec (i-1) +nec):*
last coded entirety of node I

8.1

Titrate:
SD ligrel and model

Date:
01/12/05
Author (S):
J. Key PELLET
:
D4.06.02-B Page
: 6/8

4
Contents of objects JEVEUX of the model

4.1 ".MAILLE": S V I

That is to say V = ".MAILLE"
LENGTH (V) = a number of meshs of the grid = nbma

for I = 1, nbma
V (I): number of the type of element carried by mesh I
(= 0 if the mesh is not affected by a finite element)

4.2 ".NOEUD": S V I

That is to say V = ".NOEUD"
LENGTH (V) = a number of nodes of the grid = nbno

for I = 1, nbno
V (I): number of the type of element carried by node I
(= 0 if the node is not affected by a finite element)

Note:

A node "does not carry" not really of finite element. But to simplify the work of the user, order AFFE_MODELE allows to affect finite elements "specific" on nodes of the grid. That avoids with the user thinking of creating meshs POI1 in its grid.

In this case, the specific element is carried by a late mesh of type POI1 created by AFFE_MODELE.

4.3 ".SSSA": S V I

That is to say V = "SSSA"

nb_sm = a number of super-meshs of the grid

LENGTH (V) = nb_sm + 3

for I = 1, nbsm

V (I)
=
/1 if super-mesh I is affected ("active" under-structuring)
/0 if not
V (nbsm+l)
=
a number of super-meshs of the grid
V (nbsm+2)
=
a number of active substructures
V (nbsm+3)
=
a number of nodes of Lagrange of the grid

Note:

V (nb_sm+1) and V (nb_sm+3) should not be used any more, information existing already in grid (object. DIME).

They were introduced at one time when the grid could be enriched by new super-meshs with any moment. This possibility was removed.

4.4 ".FISS": S V K8 LONG=1

.FISS (1): name of SD FISS_XFEM (produced by DEFI_FISS_XFEM)

Date:
01/12/05
Author (S):
J. Key PELLET
:
D4.06.02-B Page
: 7/8

5 Examples

5.1 SD model

MOTH=AFFE_MODELE (MAILLAGE=MAIL,
AFFE=_F (ALL = “YES”, MODELING = “AXIS”, PHENOMENON = “THERMAL”))

--
SEGMENT IMPRESSION OF VALUES >MOTH .MAILLE <
1 - 289 289 300 300 300
--
IMPRESSION OF THE COLLECTION: MOTH .MODELE .LIEL
OBJECT IMPRESSION OF COLLECTION CONTIGUE>MOTH .MODELE .LIEL< OC: 1
1 - 1 2 289
OBJECT IMPRESSION OF COLLECTION CONTIGUE>MOTH .MODELE .LIEL< OC: 2
1 - 3 4 5 300
--
SEGMENT IMPRESSION OF VALUES >MOTH .MODELE .NBNO <
1 - 0
--
SEGMENT IMPRESSION OF VALUES >MOTH .MODELE .NOMA <
1 - >MAIL <
--
SEGMENT IMPRESSION OF VALUES >MOTH .MODELE .PRNM <
1 - 2 2 2 2 2
6 - 2 0 0 0 0
11 - 0 0 0 0 0
16 - 0 0 0 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0

31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 2 2 0
46 - 0 0 0 0 0
51 - 0 0 0 0 0
56 - 0 0 0 0 0
61 - 0 0 0
--
SEGMENT IMPRESSION OF VALUES >MOTH .MODELE .REPE <
1 - 1 1 1 2 2
6 - 1 2 2 2 3
--
SEGMENT IMPRESSION OF VALUES >MOTH .NOEUD <
1 - 0 0 0 0 0
6 - 0 0 0 0 0
11 - 0 0 0 0 0
16 - 0 0 0 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0
46 - 0 0 0 0 0
51 - 0 0 0 0 0
56 - 0 0 0 0 0
61 - 0 0 0

5.2 SD
ligrel (load)

CHTH=AFFE_CHAR_THER (MODELE=MOTH,
TEMP_IMPO=_F (NODE = “N4”, TEMP = 100.0))

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.LGNS <
1 - 1 -2 0 0 0
6 - 0 0 0

IMPRESSION OF THE COLLECTION: CHTH .CHTH.LIGRE.LIEL
OBJECT IMPRESSION OF COLLECTION CONTIGUE>CHTH .CHTH.LIGRE.LIEL< OC: 1
1 - -1 92

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.NBNO <
1 - 2

IMPRESSION OF THE COLLECTION: CHTH .CHTH.LIGRE.NEMA
OBJECT IMPRESSION OF COLLECTION CONTIGUE>CHTH .CHTH.LIGRE.NEMA< OC: 1
1 - 4 -1 -2 4

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.PRNM <
1 - 0 0 0 2 0
6 - 0 0 0 0 0
11 - 0 0 0 0 0
16 - 0 0 0 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0
46 - 0 0 0 0 0
51 - 0 0 0 0 0
56 - 0 0 0 0 0
61 - 0 0 0

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.PRNS <
1 - 16 16
--

Code_Aster ®

Version
4
Titrate:
Structure of Data cara_elem
Date:
28/01/1999
Author (S):
J. PELLET
Key:
D4.06.03
Page: 1/2
Mechanical department and Digital Models
Index:
With
Diffusion:
Users - Developers
Data-processing handbook of Description
D4.06 booklet:
D4.06.03 document
Structure of Data cara_elem

1 General

The structure of data cara_elem is a whole of charts [D4.06.05] which contain information assigned to the finite elements of the model.
In general, this information relates to the elements of structure: hulls, beams,… it are by example the thickness of the hulls, characteristics of inertia of the beams,…
One also uses the CARA_ELEM to assign an orientation (a local reference mark) to Iso elements parametric. This orientation is necessary if for example the material is not isotropic.

2
Tree structure

CARA_ELEM (K8):: =record
| ' .CARARCPO'! CHART
| ' .CARCABLE'! CHART
| ' .CARCOQUE'! CHART
| ' .CARDISCA'! CHART
| ' .CARDISCK'! CHART
| ' .CARDISCM'! CHART
| ' .CARGENBA'! CHART
| ' .CARGENPO'! CHART
| ' .CARGEOBA'! CHART
| ' .CARGEOPO'! CHART
| ' .CARMASSI'! CHART
| ' .CARORIEN'! CHART
| ' .CARPOUFL'! CHART

3
Description of the charts composing the CARA_ELEM

name of the chart
name of
Description
size
“.CARARCPO”
CAARPO
characteristics of the curved elements of beam
“.CARCABLE”
CACABL
characteristics of the elements of cable
“.CARCOQUE”
CACOQU
characteristics of the elements of hull
“.CARDISCA”
CADISA

characteristics of damping of the discrete elements
“.CARDISCK”
CADISK
characteristics of rigidity of the discrete elements
“.CARDISCM”
CADISM
characteristics of mass of the discrete elements
“.CARGENBA”
CAGNBA
surface of the section of the elements of bar
“.CARGENPO”
CAGNPO
inertial characteristics of the sections of the elements of beam
“.CARGEOBA”
CAGEBA
geometrical characteristics of the elements of bar with rectangular or circular section
“.CARGEOPO”
CAGEPO
geometrical characteristics of the elements of beam with rectangular or circular section
“.CARMASSI”
CAMASS
locate orthotropism for the solid elements 3D or 2D
“.CARORIEN”
CAORIE
orientation: nautical angles of the local reference mark
“.CARPOUFL”
CAPOUF
characteristics of the elements of modeling
3D_FAISCEAU
Data-processing handbook of Description
D4.06 booklet:
Index A

Code_Aster ®
Version
4
Titrate:
Structures of Data char_meca, char_ther and char_acou
Date:
28/01/1999
Author (S):
X. DESROCHES

Mechanical department and Digital Models
Index:
With
Diffusion:
Developers
Data-processing handbook of Description
D4.06 booklet:
D4.06.04 document
Structures of Data char_meca,
char_ther and char_acou
Summary:
Presentation of the structures of data describing the mechanical, thermal loadings and acoustics in Aster.
EDF
Direction of the Studies and Research
Electricity of France
Project Codes of Mechanics

---

1 Structure of data CHAR_MECA

1.1 General

A concept of the char_meca type contains: one or more loadings and/or one or more conditions with the limits affected on a mechanical model.

Note:
This concept is produced by one of the two operators: AFFE_CHAR_MECA or AFFE_CHAR_MECA_F.

1.2 Relations with the other SD

A concept char_meca is always associated a grid, via the entities NODE,

GROUP_NO, MESH, GROUP_MA on which are defined the loadings.
On this grid must have been affected a mechanical model.
A concept char_meca can contain a LIGREL [D4.06.02] which one will call "Ligrel of load" and of CHARTS [D4.06.05].

1.3

Tree structure of the SD char_meca

char_meca (K8)
: : =
record
".chme.model.nomo"
:
S
E
K8

".chme.ligre"
:
ligrel

".chme.cimpo"
:
chart

".chme.cmult"
:
chart

".chme.dpgen"
:
chart

".chme.epsin"
:
chart

".chme.f1D2D"
:
chart

".chme.f1D3D"
:
chart

".chme.f2D3D"
:
chart

“.chme.fco2D”
:
chart

“.chme.fco3D”
:
chart

“.chme.felec”
:
chart

“.chme.fl101”
:
chart

“.chme.fl102”
:
chart

“.chme.forno”
:
chart

“.chme.impe”
:
chart

“.chme.pesan”
:
chart

“.chme.press”
:
chart

“.chme.rotat”
:
chart

“.chme.tempe.temp”
:
S
E
K8

“.chme.vnor”

:
chart
".type"
:
S
E
K8

".lisma01"
:
S
V
I

".lisma02"
:
S
V
I

".trans01"
:
S
V
R

".trans02"
:
S
V
R

".chme.coni"
:
S
V
I

".chme.conr"
:
S
V
I
Data-processing handbook of Description
D4.06 booklet:
Index A

1.4 Contents of basic objects JEVEUX

".chme.model.nomo":
name of the model associated with the load (K8)
".chme.tempe.temp":
name of the field of temperature in the case of a loading
thermics (K8)
".type"
:
type of the load (K8)
It contains one of the 2 character strings
"meca_re" --> real for operator AFFE_CHAR_MECA
"meca_fo" --> function for operator AFFE_CHAR_MECA_F
".lisma01"
:
S V I (dim = 2* NBMAIL1)
where NBMAIL1 is the number of meshs introduced behind the words

key ALL or NETS or GROUP_MA in the key word factor
INTE_ELEC.
This object contains the list of the nodes of the linear elements
defining the principal driver
“.lisma02”
:
S V I (dim = 2* NBMAIL2)
even thing for the secondary driver (if there exists)
“.trans01”
:
S V I (dim = 6)
(1) tx
x0
(tx, ty, tz) are the components of a translation of
(2) ty
y0
principal driver with the secondary driver (case where the key word
(3) tz or
z0
TRANS of FORCE_ELEC is present)
(4) 0.
N
(X
(5) 0.
X
0, y0, z0) are the punctual coordinates
N
(6) 0.
y
(nx, ny, nz) components of the normal common to
nz
principal driver and with the secondary driver (case where the word
key SYME of FORCE_ELEC is present)
“.chme.coni”
:
S V I (dim = 2* NBCOUPLE+1)
where NBCOUPLE is the intervening number of couples of meshs
in the key word factor LIAISON_UNIL_NO (conditions of contact
unilateral
(1) NBCOUPLE
Buckle I = 1, NBCOUPLE
(2i) = number of the first node
(2i+1) = number of the second node in glance
fine I buckles
Data-processing handbook of Description

".chme.conr"
:
S V R (dim = NBCOUPLE* (2*NDIM+1))
where NDIM is the dimension of space (2 or 3)
This object contains the normals with the nodes defined in
".chme.coni" and initial play
Buckle I = 1, NBCOUPLE
Buckle J = 1, NDIM
((i-1) * (2*NDIM+1) +j): j-ème component of
normal with the first node
fine J buckles
Buckle J = 1, NDIM
((i-1) * (2*NDIM+1) +NDIM+j): j-ème component of
the normal with the second node
fine J buckles
((2*NDIM+1) *i): initial play
fine I buckles

1.5
Description of the charts of a CHAR_MECA

Name chart
Name size
Description
.CIMPO
DDLI_R, _C, _F
second member of the equations of boundary conditions
kinematics
.CMULT
DDLM_R, _C
coefficients of the equations of boundary conditions
kinematics
.DPGEN

NEUT_R
generalized efforts
.EPSIN
EPSI_R, _F
initial deformation
. F1D1D
FORC_R, _C, _F
force divided linear into 1D
. F1D2D
FORC_R, _C, _F
force divided linear into 2D
. F1D3D
FORC_R, _C, _F
force divided linear into 3D
. F2D3D
FORC_R, _C, _F
force divided surface into 3D
.FCO2D
FORC_R, _C, _F
force distributed for the hulls “2D”
.FCO3D
FORC_R, _C, _F
force distributed for the hulls “3D”
.FELEC
FELECR
positional parameters of the drivers
.FL101
LISTMA
forces of Laplace
.FL102
.FLUX
FTHM_R, _F
flow “THM”
.FORNO
FORC_R, _F
nodal forces
.IMPE
IMPE_R, _F
impedance (acoustic)
.ONDE
ONDE_R, _F
amplitude of pressure of incidental wave (acoustic)
.PESAN
PESA_R
gravity

.PRESS
PRES_R, PRES_F
pressure distributed
.ROTAT
ROTA_R
acceleration of rotation
.VNOR
SOUR_R, SOUR_F
normal speed of a mesh (acoustic)
Data-processing handbook of Description
D4.06 booklet:
Index A

```
%
% LOADING IN PRESSURE WITH BLOCKING OF DDLS BY DDL_IMPO AND FACE_IMPO
%
BEGINNING ();
m = LIRE_MAILLAGE ();
Mo = AFFE_MODELE (GRID: m
VERIF: "MESH"
AFFE: (ALL: "YES"
PHENOMENON: "MECHANICAL"
MODELING: "c_plan"));
CH = AFFE_CHAR_MECA (model: Mo
DDL_IMPO: (GROUP_NO: (A B) Dy: 0. )
FACE_IMPO: (MESH: M266 dnor: 0. )
PRES_REP: (GROUP_MA: GRMA13 near: 60.));
IMPR_CO (Co: CH);
END ();
====> IMPR_CO OF THE STRUCTURE OF DATA: CH????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 27
==========================================================================
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
```

SEGMENT IMPRESSION OF VALUES >CH .CHME.CIMPO.DESC <
1 - 24 6.6 -3 1
6 - -3 2 -3 3 -3
11 - 4 -3 5 -3 6
16 - 2 2 2 2 2
21 - 2

IMPRESSION OF THE COLLECTION: CH .CHME.CIMPO.LIMA
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CIMPO.LIMA< OC: 1
1 - -1
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CIMPO.LIMA< OC: 2
1 - -2
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CIMPO.LIMA< OC: 3
1 - -3
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CIMPO.LIMA< OC: 4
1 - -4
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CIMPO.LIMA< OC: 5
1 - -5
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CIMPO.LIMA< OC: 6
1 - -6

SEGMENT IMPRESSION OF VALUES >CH .CHME.CIMPO.NOLI <
1 - >CH .CHME.LIGRE.LIEL<>CH .CHME.LIGRE.LIEL<
3 - >CH .CHME.LIGRE.LIEL<>CH .CHME.LIGRE.LIEL<
5 - >CH .CHME.LIGRE.LIEL<>CH .CHME.LIGRE.LIEL<

SEGMENT IMPRESSION OF VALUES >CH .CHME.CIMPO.NOMA <
1 - >M <

SEGMENT IMPRESSION OF VALUES >CH .CHME.CIMPO.VALE <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00

SEGMENT IMPRESSION OF VALUES >CH .CHME.CMULT.DESC <
1 - 26 6.6 -3 1
6 - -3 2 -3 3 -3
11 - 4 -3 5 -3 6
16 - 2 2 2 2 2
21 - 2

IMPRESSION OF THE COLLECTION: CH .CHME.CMULT.LIMA
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CMULT.LIMA< OC: 1
1 - -1
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CMULT.LIMA< OC: 2

1 - -2
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CMULT.LIMA< OC: 3
1 - -3
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CMULT.LIMA< OC: 4
1 - -4
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CMULT.LIMA< OC: 5
1 - -5
OBJECT IMPRESSION OF COLLECTION >CH .CHME.CMULT.LIMA< OC: 6
1 - -6

SEGMENT IMPRESSION OF VALUES >CH .CHME.CMULT.NOLI <
1 - >CH .CHME.LIGRE.LIEL<>CH .CHME.LIGRE.LIEL<
3 - >CH .CHME.LIGRE.LIEL<>CH .CHME.LIGRE.LIEL<
5 - >CH .CHME.LIGRE.LIEL<>CH .CHME.LIGRE.LIEL<

SEGMENT IMPRESSION OF VALUES >CH .CHME.CMULT.NOMA <
1 - >M <

SEGMENT IMPRESSION OF VALUES >CH .CHME.CMULT.VALE <

---

1 - 1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - 0.00000E+00 0.00000E+00 -7.07107E-01 0.00000E+00 0.00000E+00
16 - 0.00000E+00 0.00000E+00 0.00000E+00 7.07107E-01 0.00000E+00
21 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 -7.07107E-01
26 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
31 - 7.07107E-01 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - 0.00000E+00

SEGMENT IMPRESSION OF VALUES >CH .CHME.DPGEN.DESC <
1 - 58 1 1 1 9999

6 - 8190

IMPRESSION OF THE COLLECTION: CH .CHME.DPGEN.LIMA
OBJECT IMPRESSION OF COLLECTION >CH .CHME.DPGEN.LIMA< OC: 1
1 - 0

SEGMENT IMPRESSION OF VALUES >CH .CHME.DPGEN.NOLI <
1 - > <

SEGMENT IMPRESSION OF VALUES >CH .CHME.DPGEN.NOMA <
1 - >M <

SEGMENT IMPRESSION OF VALUES >CH .CHME.DPGEN.VALE <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 1.00000E+00
11 - 1.00000E+00 1.00000E+00

IMPRESSION OF THE COLLECTION: CH .CHME.LIGRE.LIEL
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.LIEL< OC: 1
1 - -1 -2 -4 -6 39
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.LIEL< OC: 2
1 - -3 -5 38

SEGMENT IMPRESSION OF VALUES >CH .CHME.LIGRE.NBNO <
1 - 8

IMPRESSION OF THE COLLECTION: CH .CHME.LIGRE.NEMA
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.NEMA< OC: 1
1 - 1 -1 -2 4
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.NEMA< OC: 2
1 - 119 -3 -4 4
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.NEMA< OC: 3
1 - 57 -5 -6 4
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.NEMA< OC: 4
1 - 57 -5 -6 4
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.NEMA< OC: 5
1 - 41 -7 -8 4
OBJECT IMPRESSION OF COLLECTION >CH .CHME.LIGRE.NEMA< OC: 6
1 - 41 -7 -8 4

SEGMENT IMPRESSION OF VALUES >CH .CHME.LIGRE.NOMA <
1 - >M <

SEGMENT IMPRESSION OF VALUES >CH .CHME.LIGRE.PRNS <
1 - 4096 4096 4096 4096 4096

6 - 4096 4096 4096

SEGMENT IMPRESSION OF VALUES >CH .CHME.MODEL.NOMO <
1 - >MO <

SEGMENT IMPRESSION OF VALUES >CH .CHME.PRESS.DESC <
1 - 63 2 2 1 9999
6 - 2 3 6.6

IMPRESSION OF THE COLLECTION: CH .CHME.PRESS.LIMA
OBJECT IMPRESSION OF COLLECTION >CH .CHME.PRESS.LIMA< OC: 1
1 - 0
OBJECT IMPRESSION OF COLLECTION >CH .CHME.PRESS.LIMA< OC: 2
1 - 0

SEGMENT IMPRESSION OF VALUES >CH .CHME.PRESS.NOLI <
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CH .CHME.PRESS.NOMA <
1 - >M <

SEGMENT IMPRESSION OF VALUES >CH .CHME.PRESS.VALE <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 6.00000E+01 0.00000E+00 0.00000E+00 0.00000E+00
11 - 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >CH .TYPE <
1 - >MECA_RE <
====> FINE IMPR_CO OF STRUCTURE OF DATA: CH????????????????
Data-processing handbook of Description
D4.06 booklet:
Index A

2

Structure of data CHAR_THER

2.1

General

A concept of the char_ther type contains one or more loadings and/or one or more conditions with the limits affected on a thermal model.

Note:

This concept is produced by one of the two operators: AFFE_CHAR_THER or AFFE_CHAR_THER_F.

2.2

Relations with the other SD

A concept char_ther is always associated a grid, via the entities NODE, GROUP_NO, MESH, GROUP_MA on which are defined the loadings.
On this grid must have been affected a thermal model.
A concept char_ther can contain a ligrel, known as ligrel of load.

2.3

Tree structure of the SD char_ther

char_ther (K8)
: : =
record
".chth.model.nomo"
:
S
E
K8

".chth.ligre"
:
ligrel

".chth.cimpo"
:
chart

".chth.cmult"
:
chart

".chth.coefh"
:
chart

".chth.flunl"
:
chart

".chth.flure"

:
chart

“.chth.grain”
:
chart

“.chth.hechp”
:
chart

“.chth.soure”
:
chart

“.chth.t_ext”
:
chart

“.chth.conve.vale”
:
S
V
K8
'.type
'
:
S
E
K8
2.4
Contents of basic objects JEVEUX
“.chth.model.nomo”:
name of the model associated with the load (K8)
“.chth.conve.vale”:
vector of dimension 1 container the name of the field speed of
transport in the case of the equation of diffusion-convection
“.type”
:
type of the load (K8)
It contains one of the 2 character strings
“ther_re” --> real for operator AFFE_CHAR_THER
“ther_fo” --> function for operator AFFE_CHAR_THER_F

```
%
% LOADING THERMAL WITH FLOW IMPOSES, EXCHANGE AND TEMPERATURE IMPOSEE
%
BEGINNING ();
EMAIL =LIRE_MAILLAGE ();
MOTH =AFFE_MODELE (GRID: EMAIL
AFFE: (ALL: “YES” MODELING: “AXIS”
PHENOMENON: “THERMAL”));
CHTH =AFFE_CHAR_THER (MODEL: MOTH
FLUX_REP: (GROUP_MA: GRMA13 FLUN: 0.0)
(GROUP_MA: GRMA14 FLUN: 1729.9091)
EXCHANGE: (GROUP_MA: GRMA12 COEF_H: 500. TEMP_EXT: 17.034444)
TEMP_IMPO: (GROUP_NO: GRNM15 TEMP: 100.0)
);
IMPR_CO (CO: CHTH);
END ();
====> IMPR_CO OF THE STRUCTURE OF DATA: CHTH????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 32
==========================================================================
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CIMPO.DESC <
1 - 24 2 2 -3 1
6 - -3 2 2 2
--------------------------------------------------------------------------
IMPRESSION OF THE COLLECTION: CHTH .CHTH.CIMPO.LIMA
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.CIMPO.LIMA< OC: 1
1 - -1
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.CIMPO.LIMA< OC: 2
1 - -2
--------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CIMPO.NOLI <
```

1 - >CHTH .CHTH.LIGRE.LIEL<>CHTH .CHTH.LIGRE.LIEL<

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CIMPO.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CIMPO.VALE <
1 - 1.00000E+02 1.00000E+02

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CMULT.DESC <
1 - 26 2 2 -3 1
6 - -3 2 2 2

IMPRESSION OF THE COLLECTION: CHTH .CHTH.CMULT.LIMA
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.CMULT.LIMA< OC: 1
1 - -1
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.CMULT.LIMA< OC: 2
1 - -2

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CMULT.NOLI <
1 - >CHTH .CHTH.LIGRE.LIEL<>CHTH .CHTH.LIGRE.LIEL<

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CMULT.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.CMULT.VALE <
1 - 1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.COEFH.DESC <
1 - 18 2 2 3 1
6 - 3 2 14 14

IMPRESSION OF THE COLLECTION: CHTH .CHTH.COEFH.LIMA
Data-processing handbook of Description
D4.06 booklet:
Index A

OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.COEFH.LIMA< OC: 1
1 - 1 2 3 4 5
6 - 6 7 8 9 10
11 - 11 12 13 14 15
16 - 16 17 18 19 20
21 - 21 22 23 24 25
26 - 26 27 28 29 30
31 - 31 32 33 34 35
36 - 36 37 38 39 40
41 - 41 42 43 44 45
46 - 46 47 48 49 50
51 - 51 52 53 54 55
56 - 56 57 58 59 60
61 - 61 62 63 64 65
66 - 66 67 68 69 70
71 - 71 72 73 74 75
76 - 76 77 78 79 80
81 - 81 83 84 85 86
86 - 87 88 89 90 91
91 - 92 93 94 95 96
96 - 97 98 99.100.101
101 - 102 104
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.COEFH.LIMA< OC: 2
1 - 82 103

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.COEFH.NOLI <
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.COEFH.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.COEFH.VALE <
>>>>>

1 - 0.00000E+00 0.00000E+00 0.00000E+00 5.00000E+02 0.00000E+00
6 - 0.00000E+00

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.FLURE.DESC <
>>>>>
1 - 37 2 2 3 1
6 - 3 2 14 14

IMPRESSION OF THE COLLECTION: CHTH .CHTH.FLURE.LIMA
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.FLURE.LIMA< OC: 1
1 - 1 2 3 4 5
6 - 6 7 8 9 10
11 - 11 12 13 14 15
16 - 16 17 18 19 20
21 - 21 22 23 24 25
26 - 26 27 28 29 30
31 - 31 32 33 34 35
36 - 36 37 38 39 40
41 - 41 42 43 44 45
46 - 46 47 48 49 50
51 - 51 52 53 54 55
56 - 56 57 58 59 60
61 - 61 62 63 64 65
66 - 66 67 68 69 70
71 - 71 72 73 74 75
76 - 76 77 78 79 80
81 - 81 82 84 85 86
86 - 87 88 89 90 91
91 - 92 93 94 95 96
96 - 97 98 99.100.101
101 - 102.103.104
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.FLURE.LIMA< OC: 2
1 - 83

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.FLURE.NOLI <
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.FLURE.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.FLURE.VALE <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 1.72991E+03 0.00000E+00
6 - 0.00000E+00

IMPRESSION OF THE COLLECTION: CHTH .CHTH.LIGRE.LIEL

OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.LIGRE.LIEL< OC: 1
1 - -1 -2 45

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.NBNO <
1 - 4

IMPRESSION OF THE COLLECTION: CHTH .CHTH.LIGRE.NEMA
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.LIGRE.NEMA< OC: 1
1 - 1 -1 -2 4
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.LIGRE.NEMA< OC: 2
1 - 2 -3 -4 4

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.LIGRE.PRNS <
1 - 16 16 16 16

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.MODEL.NOMO <
1 - >MOTH <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.T_EXT.DESC <
1 - 74 2 2 3 1
6 - 3 2 14 14

IMPRESSION OF THE COLLECTION: CHTH .CHTH.T_EXT.LIMA
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.T_EXT.LIMA< OC: 1
1 - 1 2 3 4 5
6 - 6 7 8 9 10
11 - 11 12 13 14 15

16 - 16 17 18 19 20
21 - 21 22 23 24 25
26 - 26 27 28 29 30
31 - 31 32 33 34 35
36 - 36 37 38 39 40
41 - 41 42 43 44 45
46 - 46 47 48 49 50
51 - 51 52 53 54 55
56 - 56 57 58 59 60
61 - 61 62 63 64 65
66 - 66 67 68 69 70
71 - 71 72 73 74 75
76 - 76 77 78 79 80
81 - 81 83 84 85 86
86 - 87 88 89 90 91
91 - 92 93 94 95 96
96 - 97 98 99.100.101
101 - 102 104
OBJECT IMPRESSION OF COLLECTION >CHTH .CHTH.T_EXT.LIMA< OC: 2
1 - 82 103

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.T_EXT.NOLI <
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.T_EXT.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHTH .CHTH.T_EXT.VALE <
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 1.70344E+01
6 - 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >CHTH .TYPE <
1 - >THER_RE <
====> FINE IMPR_CO OF STRUCTURE OF DATA: CHTH????????????????

3
Structure of data char_acou
3.1
General

A concept of the char_acou type contains one or more loadings affected on an acoustic model.

Note:

This concept is produced by the operator: AFFE_CHAR_ACOU.

3.2
Relations with the other SD

A concept char_acou is always associated a grid, via the entities NODE,
GROUP_NO, MESH, GROUP_MA on which are defined the loadings.

On this grid must have been affected an acoustic model.
A concept char_acou can contain a ligrel, known as ligrel of load.

3.3
Tree structure of the SD char_acou

char_acou (K8)
: : =
record
“.chac.model.nomo”
:
S
E
K8

“.chac.ligre”
:
ligrel

“.chac.cimpo”
:
chart

“.chac.cmult”
:
chart

“.chac.imped”
:
chart

“.chac.vitfa”

:
chart
'.type
'
:
S
E
K8

3.4
Contents of basic objects JEVEUX

".chac.model.nomo":
name of the acoustic model associated the load (K8)
".type"
:
type of the load (K8)
The chains "acou_re contains" (assignment of reality)

```
%
% GUIDES ACOUSTIC WAVE IN E.F. TRADITIONAL
%
BEGINNING ();
EMAIL = LIRE_MAILLAGE ();
AIR = DEFI_MATERIAU (FLUID: (RHO: 1.3 CELE_C: IH 343. 0.));
CHAMPMAT = AFFE_MATERIAU (GRID: EMAIL,
AFFE: (ALL: "YES" MATER: AIR));
GUIDE = AFFE_MODELE (GRID: EMAIL, VERIF: "MESH",
AFFE: (ALL: "YES" MODELING: "3D"
PHENOMENON: "ACOUSTIC"));
CHARACOU = AFFE_CHAR_ACOU (MODEL: GUIDE
VITE_FACE: (GROUP_MA: ENTRY VNOR: IH 0.014 0.)
IMPE_FACE: (GROUP_MA: EXIT IMPE: IH 445.9 0.));
IMPR_CO (CO: CHARACOU);
END ();
```

D4.06.04
Page: 13/12
====> IMPR_CO OF THE STRUCTURE OF DATA: CHARACOU????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 12

==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.IMPED.DESC <
1 - 46 2 2 3 1
6 - 3 2 2 2

IMPRESSION OF THE COLLECTION: CHARACOU.CHAC.IMPED.LIMA
OBJECT IMPRESSION OF COLLECTION >CHARACOU.CHAC.IMPED.LIMA< OC: 1
1 - 1 2 3 4 9
6 - 10 11 12 13 14
11 - 15 16 17 18 19
16 - 20 21 22 23 24
21 - 25 26 27 28 29
26 - 30 31 32 33 34
31 - 35 36 37 38 39
36 - 40 41 42 43 44
41 - 45 46 47 48 49
46 - 50 51 52 53 54
51 - 55 56 57 58 59
56 - 60 61 62 63 64
61 - 65 66 67 68
OBJECT IMPRESSION OF COLLECTION >CHARACOU.CHAC.IMPED.LIMA< OC: 2
1 - 5 6 7 8

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.IMPED.NOLI <
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.IMPED.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.IMPED.VALE <
1 - (0.00000E+00, 0.00000E+00) (4.45900E+02, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.MODEL.NOMO <
1 - >GUIDE <

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.VITFA.DESC <
1 - 76 2 2 3 1
6 - 3 2 2 2

IMPRESSION OF THE COLLECTION: CHARACOU.CHAC.VITFA.LIMA
OBJECT IMPRESSION OF COLLECTION >CHARACOU.CHAC.VITFA.LIMA< OC: 1
1 - 1 2 3 4
OBJECT IMPRESSION OF COLLECTION >CHARACOU.CHAC.VITFA.LIMA< OC: 2
1 - 5 6 7 8 9
6 - 10 11 12 13 14
11 - 15 16 17 18 19
16 - 20 21 22 23 24
21 - 25 26 27 28 29
26 - 30 31 32 33 34
31 - 35 36 37 38 39
36 - 40 41 42 43 44
41 - 45 46 47 48 49
46 - 50 51 52 53 54
51 - 55 56 57 58 59
56 - 60 61 62 63 64
61 - 65 66 67 68

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.VITFA.NOLI <
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.VITFA.NOMA <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >CHARACOU.CHAC.VITFA.VALE <
1 - (1.40000E-02, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >CHARACOU.TYPE <
1 - >ACOU_RE <
====> FINE IMPR_CO OF STRUCTURE OF DATA: CHARACOU????????????????
Data-processing handbook of Description
D4.06 booklet:
Index A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data CHART, CHAM_NO, CHAM_ELEM and RESUELEM Dates
:
06/10/05
Author (S):
***J. PELLET**, O.BOITEAU*
Key: D4.06.05-D Page:

Organization (S): EDF-R & D /AMA, SINETICS

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
D4.06.05 document

Structures of Data CHART, CHAM_NO, CHAM_ELEM and RESUELEM

Summary:

Description of the data-processing objects allowing to represent the fields of sizes on a GRID or a MODEL.

Author (S):
J. PELLET, O.BOITEAU

***Key: D4.06.05-D Page:
2/16***

1 General information

The 4 structures of data chart, cham_no, cham_elem and resuelem represent the fields of sizes discretized on the meshs or the nodes of a grid or on the elements of a ligrel.

We call "size" a "vector" of components (CMP) of the field. For example, for one field of displacement: ("DX", "DY", "DZ"). A discretized field is a whole of sizes located on nodes, points of Gauss or meshs. All the sizes of a field do not have *not inevitably same components: for example, on certain parts of the grid, the nodes can have 6 CMPS of displacement (elements of beam) whereas on other parts, the nodes only 3 CMPS (voluminal elements) have. The components of a size are a subset CMPS declared in the catalogue of the sizes [D4.04.01]. To describe a size, in addition to its numerical values, it is necessary to know of which CMPS it act; for that, one uses the concept of "descripteur_grandor" who describes the presence (or not) of the whole of the CMPS of the catalogue. This concept is described with [§3.1].*

·
The charts are fields discretized on the meshs of a grid (or the meshs late of a ligrel). There exists 1 size by mesh,
·
the cham_no are fields discretized on the nodes of a grid (or the nodes late of a ligrel). There exists 1 size by node,
·
the cham_elem are fields discretized on the elements of a ligrel. It can exist several sizes by element (for example a size by point of Gauss or by node). The points of dicretisation (nodes or not of Gauss) can have under-points ; if it is the case, all the points have the same number of under-points [§3.4.1],
·
the resuelem are fields discretized on the elements of a ligrel. Sizes associated such fields are the sizes known as "elementary": elementary matrices or elementary vectors. The whole of the values of a resuelem can be bulky, it is why the object containing these values (.RESL) has a structure of dispersed collection.

Important remark:

The structures of data described here are not easy use. They are SD normally used in operations of low levels: elementary calculations, assemblies, resolutions…
When one wants to read or write in such SD, it is often preferable to transform them in more convenient SD to use beforehand (cham_no_S or cham_elem_S). Routines of ad hoc transformation are [D6.00.01]: CNOCNS, CNSCNO, CELCES, CARCES…

2 Tree structures

chart (K19):: =record

".NOMA"
:

OJB

S.E.K8

".NOLI"
:

OJB

S V K24

".DESC"
:

OJB

S V I

".LIMA"
:

OJB

XC V I

".VALE"
:

/

OJB

S V R

/OJB

S V C

/OJB

S V K8

cham_no (K19):: =record

".DESC"
:

OJB

S V I

".REFE"
:

OJB

S V K24

“.VALE”
:

/

OJB

S V R

/OJB

S V C

/OJB

S V K8

/ …
% if solvor FETI (REFE (3) = ' FETI') and CHAM_NO representing a second member or a vector solution

“.FETC”:
OJB

S V Indirect K24 (*) dim=nbsd
(a number of under-fields)
(*): CHAM_NO not FETI (i.e. FETC (K) .REFE (3) “FETI” and for the moment imposed on

"MULT_FRONT")

cham_elem (K19):: =record

".CELK"
:

OJB

S V K24

".CELD"
:

OJB

S V I

".CELV"
:

/

OJB

S V R

/OJB

S V C

/OJB

S V K8

/ …

resuelem (K19):: =record

“.NOLI”
:

OJB

S V K24

“.DESC”
:

OJB

S V I

“.RESL”
:

/

OJB

XD V R

/OJB

XD V C

3
Contents of objects JEVEUX

3.1
DESCRIPTEUR_GRANDEUR

It is a vector of entireties. It describes the CMPS present indeed in a size.

All the possible CMPS of a size are described in the catalogue of the SIZES. They y are ordered. To describe the CMPS indeed present in a size one decides to keep a vector of Boolean which answers the following question: the ième CMP (in the order of sizes) is it catalogues presents in the size which one wants to describe? To save place memory (and disc), one decides "to code" this vector of Boolean on a vector entireties: on each entirety (called entier_codé), one codes 30 Boolean.

Example:

If size "DEPL_R" were described in the catalogue by:

DX DY DZ DRX
DRY
DRZ
LAGR

On an element of the beam type the descripteur_grandor is worth 126. Indeed:

DX DY DEZ DRX DRY DRZ LAGR
1 1 1 1 1 1 0
126 =
21 +
22 +
23 +
24 +
25 +
26

On element of a voluminal type the descripteur_grandor is worth 14. Indeed:

DX DY DZ DRX
DRY
DRZ
LAGR
1 1 1 0 0 0 0
14 =
21 +
22 +
23

On an additional node creates for the kinematic introduction of condition by dualisation, it descripteur_grandor is worth 128. Indeed:

DX DY DZ DERX
DRY
DRZ
LAGR
0 0 0 0 0 0 1
128
=
27

A descripteur_grandor is a vector of entier_codés: V of dimension n_ec where n_ec is it numbers the entier_codés necessary ones to the description of the size described in the catalogue.

n_ec
a number of CMPS in the catalogue
1
1 to 30
2
31 to 60
... ...

Ième entier_codé informs about the presence (or not)
numbered CMPS of 30 (i-1) +1 ---> 30*i.*
V is of dimension

3.2 SD
chart

3.2.1 General

A chart is a field discretized by mesh. Each mesh can be "affected" of a size (with more). The charts are in general SD create starting from the data of the user. Its structure is made to store (with less possible volume) information concerning the assignment of sizes on "pieces" of the grid.

Note:

The selected structure is economic spaces some but it does not answer the question quickly : which size is affected on the M1 mesh? To answer this question, it is necessary "to extend" chart (that to create bulkier temporary objects); it is the object of routine ETENCA called by CALCULATION.
A chart is thus an ordered list of couples (size, zone_affectée). The order of the couples is important because it is used to take into account the principle of overload of the assignments: the last

assignment takes precedence over the preceding ones.

One zone_affectée can be:

.
the whole of the meshs of the grid (ALL: "YES"),
.
the whole of the late meshs of a ligrel,
.
a GROUP_MA of the grid,
.
a list of meshs of the grid,
.
a list of late meshs of a ligrel.

3.2.2 Object
.NOMA

Name of the grid associated with the chart.

3.2.3 Object
.DESC

".DESC"
S V I DIM= 3 + (2+n_ec) *n_gd_max

Field "DOCU" of object .DESC contains: "CART"

DESC (1)
Gd (number of the size associated with the chart)
DESC (2)
n_gd_max (raising number of zone_affectée)
DESC (3)
n_gd_edit (a real number of zone_affectée)
DESC (3+1)
code_1er_zone ("code" of the zone_affectée first)
DESC (3+2)
number of 1st zone_affectée
.....

*DESC (3+2*n_gd_max-1)*
code_der_ent (code of the zone_affectée last)
*DESC (3+2*n_gd_max)*
number of the zone_affectée last

The "code" of one zone_affectée can be worth:

code =
1
-->
the whole of the meshs of the grid (ALL: “YES”),
code =
-1
-->
the whole of the late meshs of a ligrel,
code =
2
-->
a GROUP_MA of the grid,
code =
3
-->
a list of meshs of the grid,
code =
-3
-->
a list of late meshs of a ligrel.

If code = 1 (or 1)
the number of zone_affectée corresponding is not used for nothing.
If code = 2
the number of zone_affectée corresponding is the number of the group_ma in the collection maillа.GROUPEMA
If code = 3 (or 3)
the number of zone_affectée corresponding is the number of the object of collection .LIMA [§3.2.5] which contains the numbers of the meshs composing zone_affectée.

In object .DESC a continuation comes then from descripteur_grandor [§3.1] describing them various affected sizes. That is to say n_ec the number of entier_codé necessary to describe the CMPS of the size Gd:

*DESC (3+2*n_gd_max+1)*
beginning of the first descripteur_grandor
....

*DESC (3+2*n_gd_max*
beginning of the last descripteur_grandor
*+ (n_gd_max-1) *n_ec +1)*

Note:

For a field constant (1 only size assigned to all the meshs of the grid). One has then:

DESC (2) = 1
DESC (3) = 1
DESC (4) = 1
DESC (5) = it does not matter
DESC (6) = beginning of the descripteur_grandor of zone_affectée (ALL: "YES")

In this case .LIMA and .NOLI are not allocated (saving of space).

3.2.4 Object
.NOLI

This object is present only if the chart relates to late meshs.

It is a vector of K24 of dimension nb_gd_max. Opposite izone one finds, if this zone_affectée is a list of late meshs, the name of the ligrel or are defined these meshs.

izone
--->
nom_ligrel

3.2.5 Object
.LIMA

It is a numbered contiguous family of vectors of entireties.

.LIMA (izone)
:
V (I)

V contains (if the code of zone_affectée the izone is worth 3 or 3) the numbers of the meshs constituting zone_affectée.

The numbers of meshs of the list are numbers relating to the ligrel referred in .NOLI (izone).

if a number of mesh is > 0, it is a mesh of the grid associated with the chart.
if a number of mesh is < 0, it is a mesh of additional ligrel.

3.2.6 Object .VALE

*It is a vector of scalars dimensioned with nb_gd_max * nb_cmp_max, if nb_cmp_max is it a number of CMPS in the catalogue for the size associated with the chart.*

The size associated with zone_affectée the izone starts in .VALE with the index:

*izone --> .VALE ((izone-1) *nb_cmp_max + 1)*

Caution:

Only the affected CMPS are stored (consecutively and in the order of the catalogue) in object .VALE
For example, for a chart of DEPL_R, if the 1st zone is affected by: (DX=2. and DZ=4.)
.VALE (1) = 2.
.VALE (2) = 4.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

3.3 SD
cham_no

3.3.1 Object
.DESC

Field "DOCU" of object .DESC contains: "CHNO"

DESC (1)
Gd (size associated with the cham_no)
DESC (2)
num
DESC (3),…,
descripteur_grandor of the size if
DESC (3 + n_ec - 1)
num is < 0

If num is negative num = "-" nb_cmp

If num is < 0, its absolute value is the number of CMP of the size for ALL the nodes of grid (e.g. the field of geometry). In this case the field relates to only the nodes of grid (not of late nodes) and one suppose that all the nodes have the same representation of size.

The descripteur_grandor is then stored DESC (3) with DESC (3 + n_ec - 1).
If num is positive, there is then a structure of the prof_chno type referred in object .REFE.

3.3.2 Object
.REFE

REFE (1) name of the GRID.
REFE (2) name of a prof_chno [D4.06.07] (if DESC (2) >0)
The SD prof_chno describes the CMPS carried by the nodes of the cham_no.
It is used to point in the object .VALE which contains the values.

If FETI, it acts of the prof_chno of the total field, then for each
under-field, it is of course that local with the under-field.
REFE (3)
If solvor FETI: “FETI”
REFE (4)
If solvor FETI:
name of the structure of data of the type SD_FETI (information coming
NUME_DDL.NUME.REFN (4)).

3.3.3 Object .VALE

This object contains the “values” of the field to the nodes on the nodes of the grid or the nodes late of the ligrel used in the prof_chno.

The description of object .VALE if the cham_no is not with “constant representation” is made in [D4.06.07 §3].

If the cham_no is with “constant representation”:

That is to say
nb_no:
the number of nodes of the grid.

ncmp:
the number of CMPS carried by all the nodes of the grid.

LENGTH (.VALE) = nb_no * ncmp

VALE (1)
value of the 1st CMP carried by the 1st node
VALE (2)
value of the 2nd CMP carried by the 1st node
...
...
VALE (ncmp)
value of the last CMP carried by the 1st node
VALE (ncmp+1)
value of the 1st CMP carried by the 2nd node
...

...

The order of the CMPS is that of the catalogue of the sizes (object “&CATA.GD.NOMGD” [D4.04.01]).
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

3.3.4 Object .FETC

S V Indirect K24 (*) DIM = nbsd (a number of under-fields)
(*): CHAM_NO not FETI
(i.e. FETC (K) .REFE (3) "FETI" and for the moment imposed on "MULT_FRONT")
Optional Objet JEVEUX (present only for total field if FETI, then absent for each under-field) listing specific SD CHAM_NO to each under-field.

3.3.5 Complements for FETI

In the case of method FETI, the structure of data CHAM_NO is recursive on two levels. One "Main" SD CHAM_NO, concerning the total field (.REFE (3) = ' FETI'), comprises the objects Usual JEVEUX supplemented by a specific object of the decomposition of fields: the .FETC.
It is in fact a pointer indicating SD CHAM_NO "slaves" associated with each under-fields buildings. These local SD CHAM_NO are consisted of same objects JEVEUX as a CHAM_NO usual mono-field.
For the moment, the implementation of FETI in *Code_Aster* presupposes that these under-fields use all the same linear solvor mono-field (.REFE (3) = `MULT_FRONT' imposed by defect). This homogeneity facilitates handling of the vectors solution and second members local.

under-field 1

SD CHAM_NO

"main"

(total field)
...

.FETC

under-field *I*
SD CHAM_NO

"slaves"

(under-fields)

...

Appear 3.3.5-a: Structure of recursive data CHAM_NO if solvor FETI

In the case of a solvor FETI, one arbitrarily chose the following rule of naming for the SD CHAM_NO slave related to a under-field *J*:
nom_de_la_SD_CHAM_NO_maître (1: 11)/"F" //chaîne_de_caractères_libre (2: 8)
The character string is generated by a call to routine GCNCON.

Example: The series of following orders (resulting from case-test FETI002A)

BEGINNING (CODE=_F (NAME = ' FETI002A', NIV_PUB_WEB=' INTRANET'))
MATER=DEFI_MATERIAU (
ELAS=_F (E = 180000. , NAKED = 0.30, ALPHA = 15.E-6, RHO = 7700. ,))
MAIL=LIRE_MAILLAGE ()
MODM=AFFE_MODELE (MAILLAGE=MAIL,
AFFE= (_F (GROUP_MA = "STRU", PHENOMENON = "MECHANICAL",
MODELING = "D_PLAN",),
_F (GROUP_MA = "POISCR",
MODELING = "2D_DIS_T", PHENOMENE=' MECANIQUE'),
_F (GROUP_MA = "POIACR",
MODELING = "2D_DIS_T", PHENOMENE=' MECANIQUE'),))
CHCAR=AFFE_CARA_ELEM (MODELE=MODM,
DISCRET= (
_F (GROUP_MA=' POIACR', CARA = "K_T_N", VALE = (0. , 0. , 0. ,),),
_F (GROUP_MA=' POISCR', CARA = "K_T_N", VALE =
(180000.,0.,180000.,),),))
CHMAT=AFFE_MATERIAU (MAILLAGE=MAIL,

```
AFFE= (_F (TOUT=' OUI', MATER=MATER, TEMP_REF=20.,),))
CH1=AFFE_CHAR_MECA (MODELE=MODM,
PRES_REP= (_F (GROUP_MA=' DDLI', CLOSE = 1000. ,),
_F (GROUP_MA=' DDLI1', CLOSE = 2000. ,),),)
SDFETI=DEFI_PART_OPS (NOM=' SD',
MODELE=MODM,
INFO=1,
DEFI= (_F (GROUP_MA = “FETI1”, GROUP_MA_BORD = “B1”,),
_F (GROUP_MA = “FETI2”, GROUP_MA_BORD = “B2”,),
_F (GROUP_MA = “FETI3”, GROUP_MA_BORD = “B3”,),
_F (GROUP_MA = “FETI4”, GROUP_MA_BORD = “B4”,),),);
RESU=MECA_STATIQUE (MODELE=MODM,
CARA_ELEM=CHCAR,
CHAM_MATER=CHMAT,
SOLVEUR=_F (METHODE=' FETI',
PARTITION=SDFETI),
EXCIT= (_F (CHARGE=CH1,),),)
```

Built a “main” SD CHAM_NO “&&MESTAT.2NDMBR_ASS”…

```
====> IMPR_CO OF THE STRUCTURE OF DATA: &&MESTAT.2NDMBR_ASS?????
ATTRIBUTE: F CONTENTS: T BASE: > <
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 4
=========================================================================
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >&&MESTAT.2NDMBR_ASS.DESC <
1 - 36 1
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >&&MESTAT.2NDMBR_ASS.FETC <
1 - >&&MESTAT.2.F0000022 <>&&MESTAT.2.F0000026 <
3 - >&&MESTAT.2.F0000028 <>&&MESTAT.2.F0000032 <
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >&&MESTAT.2NDMBR_ASS.REFE <
1 - >MAIL <>RESU .00000.NUME <
3 - >FETI <>SDFETI <
--------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >&&MESTAT.2NDMBR_ASS.VALE <
```

1 - 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00
6 - 0.00000D+00 0.00000D+00 0.00000D+00 1.50000D+03 1.12500D+03
11 - 1.00000D+03 7.50000D+02 5.00000D+02 3.75000D+02 0.00000D+00
16 - 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00
21 - 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00
26 - 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00
31 - 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00 1.00000D+03
36 - 7.50000D+02 2.00000D+03 1.50000D+03

and of SD CHAM_NO "slaves" "&&MESTAT.2.F00000…" of type

...
====> IMPR_CO OF THE STRUCTURE OF DATA: &&MESTAT.2.F0000022?????
ATTRIBUTE: F CONTENTS: T BASE: > <
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 3
===
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >&&MESTAT.2.F0000022.DESC <
1 - 36 1

SEGMENT IMPRESSION OF VALUES >&&MESTAT.2.F0000022.REFE <
1 - >MAIL <>RESU. F0000007.NUME <
3 - <

SEGMENT IMPRESSION OF VALUES >&&MESTAT.2.F0000022.VALE <
1 - 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00 1.00000D+03
6 - 7.50000D+02 1.00000D+03 7.50000D+02 0.00000D+00 0.00000D+00
11 - 0.00000D+00 0.00000D+00 2.00000D+03 1.50000D+03

During an execution in parallel mode MPI, a processor sees itself allotting a certain number of under-fields (cf objects annex "&FETI.LISTE…" structure of data SD_FETI [D4.06.21]). "Main" SD CHAM_NO is always built, but its pointer .FETC does not go to indicate that under-fields concerned with the processor running: .FETC (*jk*) will be valid K24 that if the *jk under*-field is in the perimeter of the processor *J*.

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data CHART, CHAM_NO, CHAM_ELEM and RESUELEM Dates
:

For the processor *J*

vacuum

SD CHAM_NO
j1 under-field

“main”

(total field)

.FETC

under-field *J*
SD CHAM_NO “slaves”
2

(under-fields concerned with

...
the processor *J*)

Appear 3.3.5-b: Structure of recursive data CHAM_NO if solvor FETI and parallelism MPI

3.4 SD cham_elem

3.4.1 cases

cham_elem having under-points

The number of points of discretization (node, not of Gauss,…) of a cham_elem on a mesh is determined a priori by the number of points defined in the catalogue of the type_elem associated the mesh (see .NOLI (1) and .NOLI (2)). For the elements of the type “structure”, one can want to store more sizes that points defined in the catalogue.

During a non-linear calculation on a hull (for example), the integration chosen for the behavior non-linear requires to store the state of stresses in several points in the thickness: it is necessary to discretize the thickness of the hull. For that, one will say that each point of Gauss positioned on surface element (their number is fixed in the catalogue of the type_elem), is composed of N under-points representing the discretization of the normal to the element in this point.

In the same way, a non-linear element of pipe, will be able to discretize its section (circular ring) by cutting out it in sectors and layers.

On a given element, **all the points** of discretization have obligatorily the **same number of under-points**.

For creating a cham_elem with under-points, it is necessary to say for all the elements the number of under-points desired. For that, it is necessary to use a cham_elem_s size DCEL_I (argument DCELZ of routine ALCHML). When one calls the routine of elementary calculations (CALCULATION), it passage of this argument is underground: the cham_elem_s must have the same name as it cham_elem (OUT) which it is used to dimension.

3.4.2 Case

cham_elem not having under-points

By means of computer, all the cham_elem have under-points. A cham_elem which does not need this concept is in fact a cham_elem for which each point of discretization has one under-point; one then confuses the point and his under-point.

3.4.3 Case

cham_elem of size VARI_R

Size VARI_R is the special size reserved to represent a size of which the number components (CMP) is unspecified on the level of the catalogues of type_elem.

One is useful oneself for example of this size to represent the internal variables of the laws of behavior, because each law can have a number different from such variables.

In the catalogue of the sizes, this size has only one CMP: VARI.

At the time of the creation of a cham_elem_VARI_R, one must say for each element, how much components will have size VARI_R. These components will be called then: “V1”, “V2”,…, “Vn. For that, one uses the same mechanism as to declare the number of the under-points [§3.4.1]”

3.4.4 Object
.CELK

CELK (1)
name of the ligrel associated the cham_elem.
CELK (2)
name of the option of calculation associated with the cham_elem.
CELK (3)
/“ELNO”: CHAM_ELEM with the nodes
/“ELGA”: CHAM_ELEM at the points of Gauss
/“ELEM”: Constant CHAM_ELEM by element
CELK (4)
nume_couche: number of the layer (tallied on the left) for a calculated CHAM_ELEM
on a layer of element of hull.
CELK (5)
nive_couche: position in the layer for a CHAM_ELEM calculated on one
lay down element of hull:
/“INF”/“MOY”/“SUP”
CELK (6)
Name of the parameter of the option associated with the cham_elem (CELK (2))

3.4.5 Object
.CELD

.CELD: vector of entireties. Field “DOCU” of object .CELD contains: “CHML”

This object is the descriptor of the object containing the values of the cham_elem (.CELV).

CELD (1)
Gd:
size associated with the cham_elem.
CELD (2)
nb_gr: numbers grel associated ligrel.
CELD (3) mxsp
:
maximum of the number of under-points for

elements of the ligrel
CELD (4) mxcmp
:
maximum of the number of CMP (size VARI_R)
for the elements of the ligrel.
0 if size different of VARI_R
CELD (4+1) debu_grel_1:
address in .CELD of the beginning of information
concerning the 1st GREL
...

CELD (4+nb_gr) debu_grel_n:
address in .CELD of the beginning of information
concerning the last GREL

then one stores end to end the description of the field for each GREL of the ligrel

CELD (debu_grel +1)
nel:
element of the GREL numbers
CELD (debu_grel +2)
modelo:
mode_local associated the local field
(or 0 if non-existent field on the GREL)
CELD (debu_grel +3)
lgcata:
length of the local field within sight of the catalogue. it is
with saying without taking account of the under-points and of
multiple components of VARI_R.
(or 0 if non-existent field on the GREL)
CELD (debu_grel +4)
lggrel:
length total of the segment containing all them
values of the field on the GREL
then
C iel = 1, nel

CELD (debu_grel +4 +4* (iel-1) +1)
nbsp: sous_points for the element numbers
iel
CELD (debu_grel +4 +4* (iel-1) +2)
ncdyn: a number of CMP (VARI_R) for
the element iel
CELD (debu_grel +4 +4* (iel-1) +3)
lgchel: a number of values of the local field
for the element iel
lgchel= lgcata * nbsp * ncdyn

CELD (debu_grel +4 +4* (iel-1) +4)
adiel: address in object .CELV of
1st value of the element iel

3.4.6 Object
.CELV

It is a vector containing end to end the values of the local fields of the various elements.

The description of the segment concerning an element is given by the definite mode_local for type_elem. This description is éventuellemnt supplemented by the data of the number of under-points and of the number of CMPS (VARI_R).

For a field of size (different from VARI_R) not having under-points, all the elements of one even grel having same the type_elem, their local fields have all the same length and the same one organization.

One moves in object .CELV thanks to object .CELD.

One can describe the organization of object .CELV by these definitions:

CELV (ligrel) =
continuation of CELV (GREL) put end to end
CELV (GREL) =
continuation of CELV (element) put end to end
CELV (element) =
continuation of CELV (not) put end to end
CELV (not) =
continuation of CELV (under-point) put end to end
CELV (under-point) =
continuation of CMP (scalar) put end to end

3.4.7 Some “formulas” frequently used in the programming

3.4.7.1 LIGREL

number of the size associated with the CHAM_ELEM:
NUMGD=ZI (JCELD-1+1)
a number of GREL of the LIGREL associated with the CHAM_ELEM:
NGREL=ZI (JCELD-1+2)
a maximum number. under-points of the elements of a CHAM_ELEM: (perhaps = 0)
MXSP=ZI (JCELD-1+3)
a maximum number. CMPS (VARI_R) of the elements of a CHAM_ELEM:
(/=0 <=> VARI_R)
MXCDY=ZI (JCELD-1+4)

3.4.7.2 GREL: IGR

a number of elements of a GREL (IGR):
NEL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +1)

mode_local of a GREL (IGR):
IMOLO=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +2)

cumulated length of the elements of a GREL (IGR):
LGGREL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4)

address (in .CELV) beginning of GREL IGR:
DEBUGR=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +8)
then: ZR (JCELV -1 +DEBUGR) =…

length (CATALOGUE) of an element of a GREL (IGR):
LGCATA=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +3)

**3.4.7.3 Element
IEL of GREL IGR**

**address (in .CELV) beginning of element IEL of GREL IGR:
ADIEL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +4)
then: ZR (JCELV -1 +ADIEL) =…**

**length of element IEL of GREL IGR:
LGIEL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +3)**

**a number of under-points of element IEL of GREL IGR:
returns: 0 if there are no under-points
NBSPT=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +1)**

**a number of CMPS (VARI_R) of element IEL of GREL IGR:
returns: 0 if the size is not VARI_R
NCDYN=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +2)**

**3.5 SD
resuelem**

**3.5.1 Object
.NOLI**

**NOLI (1)
name of the ligrel associated the resuelem.
NOLI (2)
name of the option of calculation having given birth to the resuelem.**

**3.5.2 Object
.DESC**

Field "DOCU" of object .DESC contains: "RESL"

**DESC (1)
Gd (size associated with the resuelem)
DESC (2)
nb_gr (a number of GREL of .NOLI (1))
DESC (2+1)
mode_1er_gr (mode_local of the local fields of the first
GREL)
...**

**DESC (2+nb_gr)
mode_der_gr (mode_local of the last GREL)**

3.5.3 Object
.RESL

It is a dispersed collection of vectors of R (or C or K8,…).
The access to this collection is done by the number of GREL: .RESL (IGREL) - > V

If ncmpel is the number of scalars representing the local field for an element of the GREL,

V (1,…, ncmpel)
: values of the field on the 1st element of the GREL
V (ncmpel+1,…, 2*ncmpel)
: values of the field on the 2nd element of the GREL

---

4 Examples

4.1 SD
chart

CHART = CREA_CHAMP (TYPE_CHAMP: “CART_META_R”, OPERATION: “AFFE”,
GRID: NETTED
AFFE: (ALL: “YES”
NOM_CMP: (“ZF” “ZP” “ZB” “ZM” “P”)
VALE: (0.0.0.0.0.0 0.0.0.0))
AFFE: (GROUP_MA: GM2
NOM_CMP: (“ZF” “ZP”)
VALE: (0.2.0.3))
AFFE: (MESH: T2
NOM_CMP: (“ZP” “ZM” “P”)
VALE: (0.4.0.5.0.6))

) ;
IMPR_CO (CO: CHART);

Note:

The contents of the objects printed below can surprise: it does not correspond to what is known as with [§3.2]. Indeed this chart "was finished" by a call to routine TECART this optional the purpose of action is to allow a "fine" overload of the values affected in order CREA_CHAMP (cf [D6.10.01]).

SEGMENT IMPRESSION OF VALUES >CARTE .DESC <
1 - 64 3 3 3 1
6 - 3 2 3 3 254
11 - 254 254

IMPRESSION OF THE COLLECTION: CHART .LIMA
OBJECT IMPRESSION OF COLLECTION CONTIGUE>CARTE .LIMA< OC: 1
1 - 1 3
OBJECT IMPRESSION OF COLLECTION CONTIGUE>CARTE .LIMA< OC: 2
1 - 2
OBJECT IMPRESSION OF COLLECTION CONTIGUE>CARTE .LIMA< OC: 3
1 - 4 5

SEGMENT IMPRESSION OF VALUES >CARTE .NOLI <
1 - > <> <
3 - > <

SEGMENT IMPRESSION OF VALUES >CARTE .NOMA <
1 - >MAILLA <

SEGMENT IMPRESSION OF VALUES >CARTE .VALE <
1 - 2.00000E-01 3.00000E-01 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 2.00000E-01 4.00000E-01 0.00000E+00
11 - 5.00000E-01 0.00000E+00 0.00000E+00 6.00000E-01 0.00000E+00
16 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
21 - 0.00000E+00

4.2 SD cham_no

cham_no = CREA_CHAMP (GRID: netted, TYPE_CHAMP: "NOEU_DEPL_R",
OPERATION: "AFFE",
AFFE: (GROUP_NO: gn1
nom_cmp: "DX" VALE_R: 1.0)
AFFE: (NODE: (N2, n7)
NOM_CMP: ("DX", "DZ") vale_r: (2. , 4.))
);

IMPR_CO (CO: cham_no);

SEGMENT IMPRESSION OF VALUES >CHAM_NO .DESC <
1 - 32 6
--
SEGMENT IMPRESSION OF VALUES >CHAM_NO .REFE <
1 - >MAILLA <>cham_no <
--
SEGMENT IMPRESSION OF VALUES >CHAM_NO .VALE <
1 - 2.00000E+00 4.00000E+00 1.00000E+00 1.00000E+00 2.00000E+00
6 - 4.00000E+00

4.3 SD
cham_elem

FLUXN=CALC_CHAM_ELEM (MODELE=MOTH, TEMP=T2,
CHAM_MATER=CHMAT, OPTION=' FLUX_ELNO_TEMP')

IMPR_CO (CO=FLUXN)

--
SEGMENT IMPRESSION OF VALUES >FLUXN .CELD <
>>>>>
1 - 47 2 1 0 6
6 - 18 2 6520 8 16
11 - 1 0 8 1 1
16 - 0 8 9 3 6857
21 - 6 18 1 0 6
26 - 17 1 0 6 23

31 - 1 0 6 29

SEGMENT IMPRESSION OF VALUES >FLUXN .CELV <
>>>>>
1 - -8.78595D-12 -4.27645D-12 -8.78595D-12 -4.08919D-12 6.96696D-12
6 - -4.07954D-12 6.96696D-12 -4.77838D-12 4.96957D-12 -4.15161D-12
11 - 4.96957D-12 -4.26679D-12 -1.33159D-12 -4.54543D-12 -1.33159D-12
16 - -3.57760D-12 7.27596D-12 -8.41283D-12 7.27596D-12 -8.41283D-12
21 - 7.27596D-12 -8.41283D-12 0.00000D+00 -8.86757D-12 0.00000D+00
26 - -8.86757D-12 0.00000D+00 -8.86757D-12 0.00000D+00 -8.86757D-12
31 - 0.00000D+00 -8.86757D-12 0.00000D+00 -8.86757D-12

SEGMENT IMPRESSION OF VALUES >FLUXN .CELK <
>>>>>
1 - >MOTH .MODELE <>FLUX_ELNO_TEMP <
3 - >ELNO <> <
5 - > <>PFLUX_R

4.4 SD
resuelem

CHTH= AFFE_CHAR_THER (MODEL: MODEL TEMP_IMPO: (NODE: N8 TEMP: 3.4)
SOURCE: (ALL: “YES” SOUR: 7.));
VECTEL=CALC_VECT_ELEM (LOAD: CHTH OPTION: “CHAR_THER”);
IMPR_CO (CO: VECTEL);

The resuelem is extracted from VECT_ELEM VECTEL: “VECTEL .VE001”

SEGMENT IMPRESSION OF VALUES >VECTEL .VE001 .DESC <
1 - 105 3 5781 5648 0

SEGMENT IMPRESSION OF VALUES >VECTEL .VE001 .NOLI <
1 - >MODEL .MODELE <>CHAR_THER_SOUR_R <

IMPRESSION OF THE COLLECTION: VECTEL .VE001 .RESL
OBJECT IMPRESSION OF COLLECTION >VECTEL .VE001 .RESL< OC: 1
1 - 3.50000E+00 3.50000E+00 3.50000E+00 4.66667E+00 4.66667E+00
6 - 4.66667E+00
OBJECT IMPRESSION OF COLLECTION >VECTEL .VE001 .RESL< OC: 2
1 - 4.08333E+00 4.66667E+00 4.66667E+00 4.08333E+00

Intentionally white left page.

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
Document: D4.06.06

Structures of data cham_no_s and cham_elem_s

Summary

This document describes the structures of data cham_no_s and cham_elem_s One also gives the list of

---

1 Objective

2 new SD are defined: cham_no_s and cham_elem_s which contains same information as the SD cham_no and cham_elem but which is more "simple" to handle in FORTRAN. There exists utilities allowing to transform a cham_no into cham_no_s (and reciprocally) (in the same way for cham_elem).

These SD will be thus in general temporary SD making it possible to work more simply

Important remark:

The SD cham_no_s and cham_elem_s are not as general as the SD cham_no and cham_elem.
For the cham_no_s, one describes only the fields carried by the nodes of the grid (and not possible late nodes),
For the cham_elem_s, one does not describe that the fields carried by the finite elements of which net support is a mesh of the grid (and not a late mesh

2 SD
cham_no_s

cham_no_s (K19)

:: = record

“.CNSK”:
OJB
S
V
K8
dim=2

“.CNSD”:
OJB
S
V
I
dim=2

“.CNSC”:
OJB
S
V
K8
dim=nb_CMP

“.CNSV”
: OJB S V R/C/I/…
dim=nb_NOEUD*nb_CMP

“.CNSL”:
OJB
S
V
L
dim=nb_NOEUD*nb_CMP

Contents of the OJB:

This SD is used to describe a field of sizes carried by the nodes of a grid.

“.CNSK” (1):
netted: name of the grid subjacent with the cham_no_s.

“.CNSK” (2):
nomgd: name of the size associated with cham_no_s (“DEPL R”,
“SIEF_R”,…)

“.CNSD” (1): nb_NOEUD: a number of nodes of the subjacent grid.
“.CNSD” (2): nb_CMP: maximum CMPS carried by the nodes number.

“.CNSC” (ICMP): cmp_i: name of the ième CMP of the cham_no_s

Note:

The order of the CMPS in .CNSC can be unspecified. One is not obliged to respect the order catalogue of the sizes. On the other hand, the CMPS must belong to the CMPS of size nomgd.

“.CNSV”:

This object contains the values of the cham_no_s.
The type of this vector JEVEUX (R/C/I/K8,…) is that of the size nomgd.
*Its dimension is nb_NOEUD*nb_CMP; i.e. all the nodes of grid can carry all CMPS described in .CNSC.*

One reaches ICMP-ème CMP of INO-ème NODE by the formula:

*VALUE (INO, ICMP) = .CNSV ((INO-1) *nb_CMP + ICMP)*

Note:

The presence (or the absence) of a CMP on a NODE is indicated via object .CNSL (see below). During the creation of a cham_no_s, its nonaffected values are in settings with "undef" for better detecting their illicit use.

".CNSL":

This object contains the Boolean ones indicating the presence (or the absence) of the values of the cham_no_s.
*Its dimension is nb_NOEUD*nb_CMP; one moves there in the same way that in object .CNSV*

One examines the presence of ICMP-ème CMP of INO-ème NODE by the formula:

*EXIST (INO, ICMP) = .CNSL ((INO-1) *nb_CMP + ICMP)*

3 SD
cham_elem_s

3.1
Description of the SD

This structure of data makes it possible to represent the values of the fields discretized on the meshs of a grid.

More precisely, the access to an actual value (or complex,…) field is done while specifying:

· the number of the mesh supporting the finite element (IMA),
· the number of the point in mesh (IPT),
· the number of the under-point in point (ISP) (ISP=1 in general),
· the number of the component of the size associated with field (ICMP),

cham_elem_s (K19)

:: = record

".CESK":
OJB
S
V

K8
dim=3

“.CESD”:
OJB
S
V
I
dim=5
+
4*nb_MAILLE

“.CESC”:
OJB
S
V
K8
dim=nb_CMP

“.CESV”

: OJB S V.
dim=nbval

“.CESL”:
OJB
S
V
L
dim=nbval

3.1.1 object
.CESK

.CESK (1):
netted: name of the grid subjacent with the cham_elem_s.
.CESK (2):
nomgd: name of the size associated with the cham_elem_s (“DEPL_R”, “SIEF_R”,…)
.CESK (3):
/“ELNO”: field known with the nodes of the elements,
/“ELGA”: field known at the points of Gauss of the elements,
/“ELEM”: constant field by element (one will say whereas it is known in the center of

---

3.1.2 object
.CESD

.CESD (1):
nb_MAILLE: a number of meshs of the subjacent grid.
.CESD (2): nb_CMP
: a number of CMPS carried by the points. It is the dimension of the object
.CESC
.CESD (3): nbptmx: maximum of the number of points carried by the meshs
.CESD (4): nbspmx: maximum of the number of under-points carried by the points of the meshs
.CESD (5): nucmpmx: highest sequence number of the possible CMP of the cham_elem_s
(in the order of object .CESC)

.CESD (5+4 (ima-1) +1): nbpt (ima): a number of points of the mesh ima.*
.CESD (5+4 (ima-1) +2): nbsp (ima): a number of under-points of the mesh ima.*
.CESD (5+4 (ima-1) +3): nbcmp (ima): maximum number of the CMPS carried by under*
points of the points of the mesh ima.
.CESD (5+4 (ima-1) +4):*
IAD (ima):
IAD+1 is the address in objects .CESL and .CESV of the 1st CMP
1st under-point of the 1st point of the mesh ima (if they exist)

3.1.3 object

.CESC

.CESC (ICMP): cmp_i: name of the ième CMP of the cham_elem_s

Note:

The order of the CMPS in .CESC can be unspecified. One is not obliged to respect the order catalogue of the sizes. On the other hand, the CMPS must belong to the CMPS of size nomgd (except size VARI_R).

3.1.4 objects .CESL and .CESV

These objects contain the values of the cham_elem_s (.CESV) and Boolean (.CESL) indicating if these values were affected (or if they are unspecified).

Type JEVEUX (R/C/I/K8,…) object .CESV is that of the size nomgd.

The dimension of these 2 vectors is nbval.
*nbval is the sum on all the meshs ima of nbpt (ima) * nbsp (ima) *nbcmp (ima)*

---

to reach ICMP-ème CMP of the ISP-ème under-POINT of IPT-ème NOT of IMA-ème NET of a cham_elem_s, one uses utility routine CESEXI:

CALL CESEXI (JCESD, JCESL, IMA, IPT, ISP, ICMP, IAD) *where JCESD and JCESL are the addresses of objects .CESD and .CESL of the cham_elem_s.*

IAD is the "exit" of this routine.

if IAD > 0, that wants to say that the required component exists in the cham_elem_s. One can then to recover by: VALUE = ZR (JCESV-1+IAD) (if the field is real).

if IAD < 0, that wants to say that the required component has a possible place in the cham_elem_s but that it not affected currently. One can then affect a value in the cham_elem_s in making:
ZR (JCESV-1+IAD) = VALUE
ZR (JCESL-1+IAD) = .TRUE.

if IAD = 0, that wants to say that the required component does not have a possible place in cham_elem_s. I.e. that one at least of the following conditions is checked:
IMA > nb_MAILLE
IPT > nbpt (IMA)
ISP > nbSP (IMA)
ICMP > nbcmp (IMA)

3.2
Example of loop on the values of a cham_elem_s

CALL JEVEUO (THESE ".CESD", "It, JCESD)
CALL JEVEUO (THESE ".CESL", "It, JCESL)
CALL JEVEUO (THESE ".CESV", "It, JCESV)

NBMA = ZI (JCESD-1+1)

C 40, IMA = 1, NBMA
NBPT = ZI (JCESD-1+5+4 (IMA-1) +1)*
NBSP = ZI (JCESD-1+5+4 (IMA-1) +2)*
NBCMP = ZI (JCESD-1+5+4 (IMA-1) +3)*

C 30, IPT = 1, NBPT
C 20, ISP = 1, NBSP
C 10, ICMP = 1, NBCMP
CALL CESEXI (JCESD, JCESL, IMA, IPT, ISP, ICMP, IAD)
IF (IAD.GT.0) VALUE = ZR (JCESV-1+IAD)

4
Utility routines

CARCES
to transform a chart into a cham_elem_s
CELCES
to transform a cham_elem into cham_elem
CESCES
to change the discretization of a cham_elem (ELNO/CART/ELGA)
CESCNS
to transform a cham_elem_s into a cham_no_s
CESCRE
to create a cham_elem_s
CESEXI
to test the existence of a CMP of a point of a mesh of a cham_elem_s
CESRED
"to reduce" a cham_elem_s on a list of meshs and/or a list of CMPS.
CESTAS
"retasser" contents of a cham_elem_s
CNOCNS
to transform a cham_no into cham_no_s
CNSCES
to transform a cham_no_s into cham_elem
CNSCNO
to transform a cham_no_s into cham_no
CNSCRE
to create a cham_no_s

CNSPRJ
to project a cham_no_s on another grid
CNSRED
"to reduce" a cham_no_s on a list of nodes and/or a list of CMPS.
COPISD
to copy a cham_no_s or a cham_elem_s
DETRSD
to destroy a cham_no_s or a cham_elem_s
IMPRSD
to print on listing a cham_no_s or a cham_elem_s

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
D4.06.07 document

Structures of Data NUME_DDL, NUME_EQUA, STORAGE, PROF_CHNO

Summary:

This document describes the structures of data NUME_DDL, NUME_EQUA, STORAGE and PROF_CHNO used
to define the classification of the unknown factors of the linear systems and the storage of the assembled matrices and
vectors solution and second member.

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

---

1 General information

A NUME_DDL is used to define the classification of the unknown factors (and the equations) of a linear system.
It is pointed out that the unknown factors of such a system are CMPS carried by nodes of
GRID (or of the late nodes of LIGREL).

The matrices which one wants to describe are square; it is thus enough to describe for example their lines.
The classification of the unknown factors of a system is similar to that of the CMPS of a CHAM_NO. Moreover SD PROF_CHNO described in this document is that referred by SD CHAM_NO [D4.06.05].
SD NUME_DDL also contains the description of the tables of storage of the values of assembled matrices (SD MATR_ASSE [D4.06.10]). One calls STORAGE this part of the structure data. Possible storages are storage "MORSE" and "LIGN_CIEL".
The SD STORAGE describes how the terms of a symmetrical half-matrix in sound are arranged object ".VALE". When a MATR_ASSE is not symmetrical, it is supposed that his filling (or its "topology") remains symmetrical. STORAGE is unchanged but object ".VALE" "is doubled": 1st part to store the higher half, the 2nd part to store the lower half.

Caution:

The SD STORAGE "MORSE" is bulky (object .HCOL).

The relations of dependence between these objects can be represented by:

LIGREL
GRID
CHARGE
LIGREL
LIGREL

CHARGE
MODEL
LIGREL
CHARGE
PROF_CHNO
STORAGE
NUME_EQUA
NUME_DDL
MATR_ASSE
CHAM_NO

Appear 1-a: Bonds of the NUME_DDL, NUME_EQUA, STORAGE and PROF_CHNO with the other structures of data

---

2 Tree structures

NUME_DDL (K14):: =record
/% if solvor LDLT, MULT_FRONT or GCPC (i.e. NUME.REFN (3) = "LDLT", "MULT_FRONT" or
"GCPC"):
".NUME": NUME_EQUA
"$VIDE": STORAGE
/% if solvor MUMPS (i.e. NUME.REFN (3) = "MUMPS"):
".NUME": NUME_EQUA
"$VIDE": STORAGE

".NSLV": OJB S V K24
/% if solvor FETI (i.e. NUME.REFN (3) = "FETI"):
".NUME": NUME_EQUA
".FETN": OJB S V indirect K24 (*) dim = nbsd (a number of under-fields)
(*): NUME_DDL not FETI
(i.e. FETN (K) .NUME.REFN (3) "FETI" and for the moment imposed on "MULT_FRONT")

NUME_EQUA (K19):: =record
"$VIDE": PROF_CHNO
".DELG": OJB S V I
".NEQU": OJB S V I
".REFN": OJB S V K24

PROF_CHNO (K19):: =record
".DEEQ": OJB S V I
".LILI": OJB S NR K24
".LPRN": OJB S V I
".NUEQ": OJB S V I
".PRNO": OJB XC V I NAME ($.LILI) LONG ($.LPRN)

STORAGE (K14):: =record
/% if solvor "LDLT"
".SLCS": STOC_LCIEL
/% if solvor "MULT_FRONT"
".SMOS": STOC_MORSE
".MLTF": MULT_FRONT
/% if solvor "GCPC" or "MUMPS"
".SMOS": STOC_MORSE

STOC_LCIEL (K19):: =record
".ABLO": OJB S V I
".ADIA": OJB S V I
".DESC": OJB S V I
".HCOL": OJB S V I
".IABL": OJB S V I
".REFE": OJB S V K24
Data-processing handbook of Description

D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

STOC_MORSE (K19):: =record
".ABLO": OJB S V I
".ADIA": OJB S V I
".DESC": OJB S V I
".HCOL": OJB S V I
".IABL": OJB S V I
".REFE": OJB S V K24

MULT_FRONT (K19):: =record
".ADNT": OJB S V I
".GLOB": OJB S V I
".LOCL": OJB S V I
".PNTI": OJB S V I

Code_Aster ®
Version
8.1

Titrate:
Structure of Data NUME_DDL, NUME_EQUA, STORAGE, PROF_CHNO
Date:
29/09/05
Author (S):
J. PELLET*, O. BOITEAU Key*

3
PROF_CHNO

.LILI
S NR K24
It is the pointer of names of ".PRNO". It contains the name of the LIGREL of model and those of the LIGREL with meshs and/or late nodes. If it is about a LIGREL with meshs and with late nodes (DDL_IMPO, LIAISON_DDL…), that makes it possible to identify the nodes late implied in the PROF_CHNO. On the other hand, if it is about a LIGREL only with late meshs (FORCE_NODALE…), it does not point towards any object of collection of .PRNO.
The nodes of a PROF_CHNO are:
· is the nodes of the grid my (concerned with the model), subjacent with PROF_CHNO,
· is late nodes of one (or several) LIGREL which rest on my.
Collection ".PRNO" contains several objects:
.PRNO (1): nodes of the grid my
.PRNO (2): late nodes of the LIGREL whose name is in .LILI (J)
.PRNO (3): late nodes of the LIGREL whose name is in .LILI (K)
...
One stores in .LILI (1) the value "&MAILLA"

.LPRN
S V I
It is the pointer length of .PRNO

.PRNO
XC V I NAME ($.LILI) LONG ($.LPRN)
This collection describes which are the CMPS carried by the nodes (of the grid or late) implied in the PROF_CHNO.
If nec is the number of coded entireties of the size associated with the PROF_CHNO,
.PRNO (1) is length (nb_noeuds (my)) (2+nec)*
*.PRNO (ili) is length (nb_noeuds_tardifs ilième LIGREL) * (2+nec).*
Each node is described by 2 entireties and 1 vector of coded entireties length nec that one calls the descriptor-size [D4.06.05].
Let us take the example of the nodes of the grid my.
v = PRNO (1)
*v (ino-1) * (2+nec) +1) = ieq*

*v (ino-1) * (2+nec) +2) = nb_cmp*

*v (ino-1) * (2+nec) +2+1)*
...
descriptor-size of the node ino
*v (ino-1) * (2+nec) +2+nec)*

.nb_cmp is the number of CMPS carried by the node ino of the grid.
.ieq is the address in object .NUEQ of the 1st CMP carried by ino.

Note:

All the CMPS carried by the same node are consecutive in .NUEQ.
This is why one does not store that the address of 1st. The CMPS are ordered in the order of the catalogue of the sizes.
Unfortunately, the size associated with the .PRNO is not stored in PROF_CHNO.

.NUEQ
S V I
It is a vector of indirection between object .PRNO and object .VALE of the CHAM_NO which this PROF_CHNO refers. This vector of indirection makes it possible to be freed from the rule according to which the CMPS of a node are followed in the order of the catalogue of sizes.

That is to say for example a grid containing 3 nodes: N1, N2, N3
That is to say a size Gd having 2 CMPS: With and B (nec = 1)

If N1 carries A and B, N2 carries B and N3 carries A.

Note:

For FETI, one will check that this vector of indirection is equal to the identity bus if not that disturbs the algorithm of rebuilding of the total field solution (CHAM_NO) starting from the fields solutions buildings (CHAM_NO) with each under-field.
Normally this case of figure should arise only with the functionalities under-structuring which is proscribed with FETI. But one never knows!

1
2
6
3
1
4
4
1
2
.PRNO
N1
N2
N3
4 7 2
10
.NUEQ
N1/A N1/B N2/B N3/A
.VALE
1 2 3 4
5
6
7
8
9 10

.DEEQ
*S V I DIM = 2*neq if neq is the number of equations of the PROF_CHNO*
It is an "opposite" vector of .PRNO which describes (partially) the equations.

If nueq is a number of equation (i.e addresses in object .VALE).

*V ((nueq-1) *2+1): ino*
*V ((nueq-1) *2+2): ICMP*

· If ino > 0 and ICMP > 0
nueq is the equation associated with the icmpième CMP carried by the node ino with grid.
· If ino > 0 and ICMP < 0
nueq is one of the 2 equations which dualisent the blocking of the icmpième CMP of node ino of the grid.
· If ino = 0 and ICMP = 0
nueq is an equation of dualisation of a linear relation between several CMP.

4
NUME_EQUA

.REFN
S V K24 dim = 4
***.REFN (1)** = name of the grid subjacent with the NUME_DDL.*
***.REFN (2)** = name of the size simple partner NOMGDS TEMP R, DEPL R, PRES_C…*
***.REFN (3)** = standard of resolution of the linear solvor: “LDLT”, “GCPC”, “MULT_FRONT” “MUMPS” or “FETI” (information coming from SOLVEUR.SLVK (1)).*
***.REFN (4)** = name of the structure of data of the type SD_FETI (information*

coming from SOLVEUR.SLVK (6)).

.NEQU
NEQU (1) a total number of equations.
NEQU (2) unutilised

.DELG
S V I dim = neq
this object described (a little) the equations of the type “Lagrange”.
DELG (ieq) =/-1
if the equation ieq corresponds to CMP “LAGR” carried by the 2nd node of one net SEG3 carrying an element of Lagrange (Lagrange 1)
DELG (ieq) =/-2
if the equation ieq corresponds to CMP “LAGR” carried by the 3rd node of one net SEG3 carrying an element of Lagrange (Lagrange 2)
DELG (ieq) =/0 if not
.FETN
S V Indirect K24 (*) dim = nbsd (a number of under-fields)
(*): NUME_DDL not FETI
(i.e. FETN (K) .NUME.REFN (3) “FETI” and for the moment imposed on “MULT_FRONT”)
Optional Objet JEVEUX (present only for total field if FETI, then absent for each under-field) listing SD NUME_DDL suitable for each under-field.

5 Complement for MUMPS

.NSLV
S V K24 dim = 1
.NSLV (1) *= name of SD SOLVEUR.*

6
Complements for FETI

6.1
Structure of recursive data NUME_DDL

In the case of method FETI, the structure of data NUME_DDL is recursive on two levels.
A “main” SD NUME_DDL, concerning the total field (.NUME.REFN (3) = ' FETI'), comprises objects JEVEUX usual supplemented by a specific object of the decomposition of fields:
.FETN.
It is in fact a pointer indicating SD NUME_DDL “slaves” associated with each under local fields. These local SD NUME_DDL are consisted of same objects JEVEUX as one NUME_DDL usual mono-field.
For the moment, the implementation of FETI in Code_Aster presupposes that these under-fields use all the same linear solvor mono-field (.NUME.REFN (3) = `MULT_FRONT' imposed by defect). This homogeneity facilitates handling of the matrices and second members local.
Of course, during factorization symbolic system, one created the structures of data STORAGE of under-fields but not that of the total field. Only the NUME_EQUA of this last is born, for to be able to handle a total field solution and… to point towards the local fields.

under-field 1

SD NUME_DDL

“main”

(total field)

...

.FETN

under-field I
SD NUME_DDL

“slaves”

(under-fields)

...

Appear 6.1-a: Structure of recursive data NUME_DDL if solvor FETI

6.2
Regulate naming

In the case of solvor FETI, one chose the following rule of naming for SD NUME_DDL
slave related to a under-field J:
nom_de_la_SD_NUME_DDL_maître (1: 6)/“F” //chaîne_de_caractères_libre (2: 8)
The character string is generated by a call to routine GCNCON.

In addition, them pre-necessary of the routines of constitution of the NUME_DDL (slaves) impose creation
of LIGRELs temporary named
“&F”/“chaîne_de_caractères_libre (3: 8)/” .MODELE '

6.3 Checking

.NUEQ

In the case of solvor FETI, one checks that the vector of indirection .NUEQ of the main NUME_DDL is equal to the identity because this assumption facilitates the operation of rebuilding of the field total result with
to leave the fields local results. Normally, this case of figure should occur only for
under-structuring which is in any event contra-indicated, for other reasons, with FETI.

6.4
Particular case of parallelism MPI

During an execution in parallel mode MPI, a processor sees itself allotting a certain number of under-fields (cf objects annex "&FETI.LISTE…" structure of data SD_FETI [D4.06.21]). "Main" SD NUME_DDL is always built, but its pointer .FETN does not go to indicate that under-fields concerned with the processor running: .FETN (jk) will be valid K24 that if the jk under-field is in the perimeter of the processor J.

For the processor J

vacuum

SD NUME DDL
j1 under-field

"main"

(total field)

.FETN

under-field J
SD NUME_DDL "slaves"
2

(under-fields concerned with

...

the processor J)

Appear 6.4-a: Structure of recursive data NUME_DDL if solvor FETI and parallelism MPI

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

7
STOC_LIGN_CIEL

.REFE
(1)
name of classification supporting this storage.

.DESC
(1)
a number of equations: neq

(2)
cut blocks of the matrix: t_bloc

(3)
a number of blocks necessary to the storage of the values
matrix: n_bloc

(4)

maximum height of the columns of the matrix

.HCOL

S V I dim = neq
.HCOL
(I)
height of the ième column

.ADIA

S V I dim = neq
.ADIA
(I)
address diagonal term of the ième column in its block

.ABLO

S V I dim = n_bloc + 1
.ABLO
(1)
0

(K+1) number of the last column of the block K.
note: a column can belong only to one block

.IABL

S V I dim = neq
.IABL
(I)
number of the block which contains the ième column of the matrix

Example:

1st block
2nd block
X X
X X
X
X X
HCOL (1) = 1
X X O O

HCOL (2) = 2
X O O X X
HCOL (3) = 1
X X X X
HCOL (4) = 2
X X X X X
HCOL (5) = 5
X X X X
HCOL (6) = 6
X O O
HCOL (7) = 5
X X
HCOL (8) = 6
X
HCOL (9) = 5
HCOL (10) = 6

ADIA (1) =
1
ADIA (6) =
17
ADIA (2) =
3
ADIA (7) =
22
ADIA (3) =
4
ADIA (8) =
6
ADIA (4) =
6
ADIA (9) =
11
ADIA (5) =
11
ADIA (10)
=
17

ABLO (1) =
0
IABL (1 to 7)
=

1
ABLO (2) =
7
IABL (7 to 10) = 2
ABLO (3) =
10

---

8 *STOC_MORSE*

.REFE
(1)
name of classification supporting this storage.

.DESC
(1)
a number of equations: neq

(2)
cut block of the matrix: t_bloc

(3)
block numbers: 1

(4)

a number of terms stored in the matrix: n_termes

.HCOL

S V I dim = n_termes
.HCOL
(I)
number of line of the ième stored term

.ADIA

S V I dim = neq
.ADIA
(I)
address diagonal term of the ième column in the block
It is necessary thus that all the diagonal terms are stored in the matrix

.ABLO
(1) 0

(2) neq

.IABL

S V I dim = neq
.IABL
(I) 1

Note:

Today (August 2004), the filling of the matrix (object .HCOL) is such as all DDLs carried by a node are connected to all DDLs carried by the other nodes being next to it node via a finite element. The matrix is thus formed small full rectangles corresponding to the connectivity of these nodes.

Example:

1 2
7 10
1
3
8 11
2

HCOL (1) = 1
3
4 5
14 19
HCOL (2) = 1
4
6
15 20
HCOL (3) = 2…
5
9 12 16 21 25 28
HCOL (4) = 3 HCOL (28)
= 5
6
13 17 22 26 29
HCOL (5) = 3 HCOL (29)
= 6
7
18 23
HCOL (6) = 4 HCOL (30)
= 9
8
24
HCOL (7) = 1 HCOL (31)
= 10
9
27 30
HCOL (8) = 2
10
31
HCOL (9) = 5

ABLO (1) =
0
ABLO (2) =
10

IABL (1 to 10) = 1

ADIA (1)
=
1
ADIA (6) =

13

ADIA (2)
=
3
ADIA (7) =
18

ADIA (3)
=
4
ADIA (8) =
24
ADIA (4)
=
6
ADIA (9) =
27
ADIA (5)
=
9
ADIA (10)
=
31

9
MULT_FRONT

That is to say lgind, the sum of the number of the neighbors of the super-nodes.

.GLOB:
S V I
dimension = lgind
This vector gives the whole of the neighbors of the super-nodes

.LOCL:
S V I
dimension = lgind
This vector establishes for the numbers of lines of the supernoeuds, the correspondence between the local classification of the son and the local classification of the father.

.ADNT:
S V I
dimension = size of the initial matrix (Morse)
It is the vector of the addresses of the initial terms in the factorized matrix.

.PNTI:
S V I
*dimension = 19*neq+10*
Office plurality in the same vector of a succession of work tables.

10 Examples

An example of NUME_DDL associated with the 4 linear solveurs is given in:

“LDLT”
[D4.06.10 §5.2]
“MULT_FRONT”
[D4.06.10 §5.3]
“GCPC”
[D4.06.10 §5.4]
“FETI”
[D4.06.10 §5.5]

Intentionally white left page.

Code_Aster ®
Version
8.1

Titrate:
Description of the Structure of Data SD_RESULTAT

Date:
01/12/05
Author (S):
J.** Key **PELLET
:
D4.06.08-C Page
: 1/8

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D4.06 booklet: -
D4.06.08 document

Description of the Structure of Data
SD_RESULTAT

Summary:

One describes the contents of the representatives objects here the structure of data SD_RESULTAT.

Data-processing handbook of Description
D4.06 booklet: - HT-66/05/003/A

---

1 Introduction

The results of a calculation by finite elements are fields of scalars, vectors or tensors, but also of the parameters attached to these fields. For example, modal analysis of a structure conduit to calculate the clean vectors (fields of displacement) and the Eigen frequencies associated.

When calculation is not reduced to the resolution of only one linear system, the operators produce a whole of fields and parameters which are gathered in the structure of Résultat data "made up" described in this document and which is also called to be more precise SD_RESULTAT.

2 Typing of one SD_RESULTAT

The results (fields and parameters) likely to be stored in a SD_RESULTAT are enough variables. For example, the results of a transitory dynamic calculation can be fields of speed or of acceleration, which is not the case for a quasi-static calculation, results of a calculation thermics can be fields of temperatures or heat flow.

To distinguish all the possible situations the SD_RESULTAT are typified. One will speak for example SD_RESULTAT of the dyna_tran type for the results of a transitory dynamic calculation,

evol_noli for a nonlinear quasi-static calculation, evol_ther for a thermal calculation. These various types are described in the Instruction manual [U5], moreover, the SD_RESULTAT are create by a single routine RSCRSD [D6.05.01] whose source is relatively explicit.

All the SD_RESULTAT some are their types are accessible to the programming from same routines [D6.05.01].

3 Structure of the SD_RESULTAT

3.1 Parameters and variables of access

Information (fields and parameters) of a SD_RESULTAT is indexed by an entirety. This index sequence number or NUME_ORDRE is called. This index does not vary inevitably from 1 to N. Numbers of order can be negative or null; they can not be consecutive. For a RESULT of the type transient for example the sequence number 0 corresponds at the initial state, the sequence number 1 corresponds at the first moment of calculation. This NUME_ORDRE corresponds one or more parameters which allow also to reach information. For example, the moment of an evolutionary calculation or the frequency or it sequence number of a clean mode. These parameters of a a little particular kind are called variable of access. The other parameters (generalized mass of a mode, for example) are simply called parameters without another distinction.

A parameter (variable of access or other) can be of the whole, real, complex type or character.

3.2 Reference symbol of the fields

With a given NUME_ORDRE several fields can correspond which one distinguishes by what one call a reference symbol, for example, DEPL for the fields of displacements, SIGM_ELNO_DEPL for the constraints by elements with the nodes calculated starting from the field of displacement. These fields are of cham_no type, cham_elem or chart.

To simplify the reference symbols retained for the various types of SD_RESULTAT [U5] are names of sizes or the names of options of calculation.

Schematically a SD_RESULTAT is thus presented in the following form (example of SD_RESULTAT of the type mode_meca cf [U5.01.23]).

Number
1
2
3
...
4
of order
Reference symbol
"DEPL"
Clean vector
Clean vector
Constraints with the nodes by
Constraints with the nodes by
"SIGM_ELNO_DEPL"
"SIGM_ELNO_DEPL"
elements calculated to leave
elements calculated to leave
displacements
displacements
"EPSI_ELNO_DEPL"
Deformations with the nodes by
Deformations with the nodes by
"EPSI_ELEM_DEPL"
elements calculated to leave
elements calculated to leave

displacements
displacements
...
...
...
...
...
Variables of access
NUME_ORDRE
1
2
3
...
N
FREQ
Eigen frequency
NUME_MODE
Number of the clean mode
Others
parameters
NORMALIZES
Normalizes clean mode
OMEGA2
Square of the pulsation
...

In the first two-dimensional board (reference symbol and sequence number), one finds names of fields (K19 stored in a vector of K24).

In 2nd and 3rd tables, one finds the value (scalar) variables of access (or parameters).

4
SD_RESULTAT

Concepts SD_RESULTAT are typified.

The various concepts SD_RESULTAT whose contents are described in the booklet [U5.01] are the following (this nonrestrictive list, could be supplemented progressively with the developments of Aster; it is not, however not desirable that it lengthens too much):

EVOL_ELAS
: Result resulting from a quasi-static calculation with evolution with time
EVOL_NOLI
: Result resulting from a quasi-static or dynamic calculation non-linear
DYNA_TRANS
: Result resulting from a transitory linear dynamic calculation in space
physics
DYNA_HARMO
: Result resulting from a harmonic dynamic calculation in physical space
HARM_GENE
: Result resulting from a harmonic dynamic calculation in modal space
(generalized size)
ACOU_HARMO
: Result resulting from a harmonic acoustic calculation in physical space
MODE_MECA
: Result resulting from a calculation of research of eigenvalues and vectors
proper mechanics
MODE_GENE
: Result resulting from a calculation of research of eigenvalues and vectors
proper mechanics starting from generalized sizes
MODE_ACOU
: Result resulting from a calculation of research of eigenvalues and vectors
proper mechanics starting from acoustic sizes
MODE_STAT
: Result resulting from a calculation of static modes
EVOL_THER
: Result resulting from a transitory thermal calculation

BASE_MODALE: Result resulting from a regrouping from mechanical modes and modes statics

5 Tree structure

SD_RESULTAT
(K19):: ==
record

“.DESC” OJB
S NR K16
“.TACH” OJB
XC
V
K24
“.NOVA” OJB
S NR K16
“.TAVA” OJB
XC
V
K8
“.ORDR” OJB
S V I

% if MODE_MECA, MODE_GENE, MODE_STAT, BASE_MODALE, DYNA_TRAN,
%
DYNA_HARM, HARM_GENE, TRAN_GENE:

“.REFD” OJB
S V K24

LONG=6
Data-processing handbook of Description
D4.06 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

6
Contents of basic objects JEVEUX

".DESC":
This object is a pointer of name containing the reference symbols of these fields.
".TACH":
Contains the names of the fields contained in the RESULT.
This object is a collection of vectors constant length pointed by .DESC.
".NOVA":
This object is a pointer of name containing the names of the variables of access and parameters of calculation.
".TAVA":
Described the variables of access and the parameters of calculation.
This object is a collection of vectors length equal to 4 pointed by .NOVA.
For a given name, one finds:

- *in the first element of the vector, the name of the suffix of the OJB where is stored the value (K5),*
- *in the second, the characters associated with the row with the parameter allowing to find its value when a number is associated to him of order,*
- *in the third, the total number of different parameters contained in the OJB,*
- *in the fourth, one indicates if it is a variable of access or one parameter.*

".ORDR":

This object is a vector of entireties. It contains the sequence numbers stored in the SD.
That is to say for example: .ORDR = (0,10,20,30)
This SD has 4 sequence numbers: 0,10,20,30 associated the 4 numbers of arrangement 1,2,3,4.
".REFD":
Vector of 6 K24 (they all are not used at the same time):
(1): name of the MATR_ASSE of stiffness
(2): name of the MATR_ASSE of mass
(3): name of the MATR_ASSE of damping
(4): name of the NUME_DDL
(5): name of SD INTERF_DYNA_CLAS

7
Regulate construction of the names of the fields

The name of the structures of data of the fields contained in the .TACH is composed to leave:

.
of the first 8 characters of the name of the concept "made up" RESULT: resu
.
characters associated with the number in the pointer with the reference symbol with the result (K3): nusymb
.
characters (K6) associated the sequence number: nuordr (limitation with 106 steps of time) it who gives:

K8
K1 K3 K1 K6

= K19
resu//". " //nusymb// ". " //nuordr

Example:

That is to say "moderesu" a name of concept "made up" RESULT:

.
the clean vector of the mode of sequence number 1 is a cham_no of name

"moderesu.001.00001"
.
the deformation with the nodes by element of the mode of sequence number 9 is a cham_elem of name "moderesu.003.00009"
Data-processing handbook of Description
D4.06 booklet: - HT-66/05/003/A

8
Access rule to the values of the variables of access and to parameters

The value of a variable of access or a parameter of name nosymb and sequence number nuordr of a concept RESULT "made up" of name resu is in the OJB of name:

K19
K5 = K24
resu//nosuff

with the index nmax (irang-1) +ivar*

where:
· irang is the number of arrangement of the sequence number nuordr.
· the name of the suffix nosuff, the nmax number of variables, and the index ivar are respectively in the first, second and third elements of the vector of the nosymb object collection of name:

K19
K5 = K24

resu//".TAVA"

Example:

That is to say "moderesu" is a name of concept "composed" RESULT, generalized stiffness RIGI_GENE
mode of sequence number 11 (arranged with index 7 of object .ORDR) will be in the OJB vector:

"moderesu .PARA"

with the index 27 * (7 - 1) + 5

".PARA", "27" and "5" were found in the OJB of name "moderesu .TAVA" opposite name RIGI_GENE.

9
Reference symbols, variables of access and parameters of RESULT

The complete contents of the various types of SD_RESULTAT are in the instruction manual [U5]. One gives here some examples of the fields and the stored parameters.

9.1
SD_RESULTAT of the type EVOL_ELAS

.
Reference symbols of the fields:
"DEPL"
"SIEF_ELGA_DEPL"
"SIEF_ELGA_DPGE"
"EPSI_ELNO_DEPL"
"EPSI_ELNO_DPGE"
"DEGE_ELNO_DEPL"
...
.
Variables of access:
"INST"
.
Parameters:
"EFFORT_N" "MOMENT_MFY"
"MOMENT_MFZ"

9.2
SD_RESULTAT of the type MODE_MECA

•
Reference symbols of the fields:
"DEPL"
"SIEF_ELGA_DEPL"
"EPSI_ELNO_DEPL"
...
"SIGM_ELNO_CART"
"FORC_NODA"
"REAC_NODA"
•
Variables of access:
"NUME_MODE"
"FREQ"
•
Parameters:
"STANDARD" "OMEGA2"
"AMOR_REDUIT"
"ERROR"
"MASS_GENE"
"RIGI_GENE"
"AMOR_GENE"

"MASS_EFFE_DX"
"MASS_EFFE_DY"
...

10 Example
of one
SD_RESULTAT of the type MODE_MECA

.DESC
.TACH

Eigenvalue
"DEPL"

Constraints with the nodes by

"SIGM_ELNO_DEPL"
elements calculated to leave

displacements

Deformations with the nodes by

"EPSI_ELNO_DEPL"
elements calculated to leave

displacements

...
...
...
...
...

.ORDR

name
2

1
2
3
...
N

Eigen frequencies
“.freq”
2
variable of access

numbers of the clean modes
“.numo”
2
variable of access

2
modal parameters
“will.para”
parameters of calculation
.NOVA
.TAVA
NUME_MODE
“.NUMO” “1” “1” “ACCESS”
FREQ
“.FREQ” “1” “1” “ACCESS”
NORMALIZES
“.PARA” “1” “1” “PARA”
OMEGA2
“.PARA '' 1” “27 '' PARA”
...
...
MASSE_GENE
“.PARA” “4” “27” “PARA”
RIGI_GENE
“.PARA” “5” “27” “PARA”
...
...
“27” “PARA”
MASS_GENE_DZ
“.PARA '' 27 '' 27” “PARA”
Data-processing handbook of Description
D4.06 booklet: - HT-66/05/003/A

Intentionally white left page.

Version
4
Titrate:
Structure of Data char_cine
Date:
28/01/1999
Author (S):
J. PELLET
Key:
D4.06.09
Page: 1/4
Mechanical department and Digital Models
Index:
With
Diffusion:
Users - Developers
Data-processing handbook of Description
D4.06 booklet:
D4.06.09 document
Structure of Data char_cine_meca,
char_cine_ther and char_cine_acou
1
General
The structures of data char_cine_meca, char_cine_ther and char_cine_acou contain
the information introduced by the user into the order affe_char_cine. I.e. them
information concerning blockings of DDLS which one wants to eliminate (and not dualiser).
EDF
Direction of the Studies and Research
Electricity of France
Project Codes of Mechanics

Code_Aster ®

Version
4
Titrate:
Structure of Data char_cine
Date:
28/01/1999
Author (S):
J. PELLET
Key:

D4.06.09
Page: 2/4
2
Tree structure
CHAR_CINE_MECA (K8):: =record
“.CIME.MODEL.NOMO”
: OJB
S V K8
“.TYPE”
: OJB
S V K8
“(11) .DEFI”
: OJB
S V I
/“(11) .VALE”
: OJB
S V R
/“(11) .VALF”
: OJB
S V K8
CHAR_CINE_THER (K8):: =record
“.CITH.MODEL.NOMO”
: OJB
S V K8
“.TYPE”
: OJB
S V K8
“(11) .DEFI”
: OJB
S V I
/“(11) .VALE”
: OJB
S V R
/“(11) .VALF”
: OJB
S V K8
CHAR_CINE_ACOU (K8):: =record
“.CIAC.MODEL.NOMO”
: OJB
S V K8
“.TYPE”
: OJB
S V K8

“(11) .DEFI”
: OJB
S V I
“(11) .VALE”
: OJB
S V R
3
Contents of the OJB
The 3 SD char_cine_xxxx are completely similar (and could be amalgamated!).
The only difference between them is that object “.NOMO” does not have the same name exactly: ” . CIME "
for mechanics, “.CITH” for thermics and” .CIAC " for accoustics.
We will describe here the SD char_cine_meca which will be used as model for the 2 others.
3.1
Object “.CIME.MODEL.NOMO”
This object contains the name of the model associated with the kinematic load.
3.2
Object “.TYPE”
This object contains a chain of caratères “typifying” the load.
/“CIME_RE”
charge kinematic “real” mechanics
(AFFE_CHAR_CINE/MECA_IMPO)
/“CIME_FO”
charge kinematic mechanics “function”
(AFFE_CHAR_CINE/MECA_IMPO)
/“CITH_RE”
charge kinematic thermics “real”
(AFFE_CHAR_CINE/THER_IMPO)
/“CITH_FO”
charge kinematic thermics “function”
(AFFE_CHAR_CINE/THER_IMPO)
/“CIAC_CX”
charge kinematic accoustics “complexes”
(AFFE_CHAR_CINE/ACOU_IMPO)
Data-processing handbook of Description
D4.06 booklet:
Index A

3.3 Object "(11) .DEFI"

One calls a blocking, a kinematic condition being written in the form: CMP_i (NOEUD_j) = alpha_ij.
A kinematic load in is made a list of such blockings.
That is to say nbloc the number of blockings of the load,
object ".DEFI" is then length 3*nbloc + 1

.DEFI (1)
nbloc
.DEFI (2)
number of the NODE concerned with the 1st blocking
.DEFI (3)
number of the CMP concerned with the 1st blocking
.DEFI (4)
1 (useless information)
.DEFI (5)
number of the NODE concerned with the 2nd blocking
.DEFI (6)
number of the CMP concerned with the 2nd blocking
.DEFI (7)
1 (useless information)
...
...

3.4 Object "(11) .VALE"

Object ".VALE" is length nbloc.
It is used if the load is real" or "complex" kinematic load a "
(Order AFFE_CHAR_CINE).

.VALE (1)
specified value for the 1st blocking
.VALE (2)
specified value for the 2nd blocking
.VALE (3)
specified value for the 3rd blocking
...

3.5 Object "(11) .VALF"

Object ".VALF" is length nbloc.
It is used if the load is a kinematic load "function"
(Order AFFE_CHAR_CINE_F).

.VALF (1)
specified value for the 1st blocking

.VALF (2)
specified value for the 2nd blocking
.VALF (3)
specified value for the 3rd blocking
...

4
Example

```
chcine= AFFE_CHAR_CINE (model: Mo
MECA_IMPO: (GROUP_NO: (gn1) DX: 1. )
MECA_IMPO: (GROUP_NO: (gn2) DX: 2. DY: 3. )
);
IMPR_CO (CO: chcine);
SEGMENT IMPRESSION OF VALUES >CHCINE .DEFI <
1 - 5 1 1 1 2
6 - 1 1 3 1 1
11 - 5 1 1 5 2
16 - 1
-------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >CHCINE .VALE <
1 - 1.00000E+00 1.00000E+00 1.00000E+00 2.00000E+00 3.00000E+00
-------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >CHCINE .CIME.MODEL.NOMO <
1 - >MO <
-------------------------------------------------------------------------------
SEGMENT IMPRESSION OF VALUES >CHCINE .TYPE <
1 - >CIME_RE <
```

Organization (S): EDF-R & D /AMA, SINETICS

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
D4.06.10 document

Structure of Data MATR_ASSE

Summary:

This document describes the structures of data MATR_ASSE: assembled matrices (hollow).

: 2/22

1 General information

The objects of the type MATR_ASSE represent the square matrices assembled (within the meaning of the elements stop). It is in general of large objects. These matrices are hollow, which explains why their structure is not simply a square table.

A MATR_ASSE can come from an assembly of MATR_ELEM or a linear combination other MATR_ASSE.

The tables describing the storage of the MATR_ASSE are in the structure STORAGE of one NUME_DDL [D4.06.07].

With each MATR_ASSE a method of resolution is attached (one will say SOLVEUR [D4.06.11]), because it solvor determines the mode of storage.

There is symmetrical MATR_ASSE and not-symmetrical MATR_ASSE. But it is supposed that it filling of the matrix is always symmetrical.

NUME_DDL
MATR_ELEM
MATR_ELEM
MATR_ELEM
matr_asse

Appear 1-a: Bonds of the MATR_ASSE with the other structures of data
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data MATR_ASSE

Date:
13/10/05
Author (S):
***J.** Key **PELLET**, O.BOITEAU*
:
D4.06.10-C Page
: 3/22

2 Tree structure

MATR_ASSE (K19):: =record

“.REFA”:

OJB

S V K24

/% if solvor not FETI: MULT_FRONT, LDLT, GCPC or MUMPS (REFA (5) “FETI”)

“.VALE”:

OJB

XD V /R/C NUM () dim=CSTE

% if there are ddls of Lagrange:

“.CONL”:

OJB

S V R

% if the MATR_ASSE is packaged by the diagonal.

“.COND”:

OJB

S V R

% if the MATR_ASSE comes from an assembly:

“.LIME”:

OJB

S V K8

% if the MATR_ASSE were factorized with the solvor multifrontal (REFA (5) = ' MULT_FRONT'):

'.VALF
:

OJB

XD V R

NUM ()

%

if there are loads kinematics:

".ABLI":

OJB

S V I

".ALIG":

OJB

S V I

".CONI":

OJB

S V I

".LLIG":

OJB

S V I

".VALI":

OJB

XD V R
NUM () dim=CSTE

/% if solvor FETI (REFA (5) = ' FETI'):

".LIME":

OJB

S V K8

“.FETM”:

OJB

S V Indirect K24 (*) dim = nbsd
(a number of under-fields)
(*): MATR_ASSE not FETI
(i.e. FETM (K) .REFA (5) “FETI” and for the moment imposed on “MULT_FRONT”)

%

so at least one of the under-fields is floating:

“.FETF”:

OJB

S V I
dim = nbsd

“.FETP”:

OJB

XD V I
LENGTH = nbsdf (a number of
floating under-fields)

“.FETR”:

OJB

XD V R
LENGTH = nbsdf
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data MATR_ASSE

3
Contents of the OJB

.REFA:
S V K24 dim=6

.REFA (1) = name of the subjacent GRID.

.REFA (2) = name of the NUME_DDL.

.REFA (3) = name of the structure of storage of the initial matrix
STOC_LCIEL = NUME_DDL.SLCS (if LDLT),
STOC_MORSE = NUME_DDL.SMOS (if GCPC, MULT_FRONT, MUMPS or FETI).

.REFA (4) = name of the OPTION of calculation or "&&MELANGE".

.REFA (5) = standard of resolution of the linear solvor: "LDLT", "GCPC", "MULT_FRONT",
"MUMPS" or "FETI" (information coming from NUME_DDL.NUME.REFN (3)).

.REFA (6) = name of the structure of data of the type SD_FETI (information
coming from NUME_DDL.NUME.REFN (4)).

DOCU (".REFA") =/"ADZE"

initial matrix,

/
"DECT"
stamp completely factorized,

/"DECP"

stamp partially factorized.

.VALE:
XD V R NUM () LONG=CSTE
There is nbloc elements in this collection.

.VALE (I): values of the terms of the matrix contained in the ième block.
The arrangement of the terms of the matrix in the blocks is explained in

documentation of the SD STORAGE [D4.06.07]

DOCU (".VALE") =/"MS"

symmetrical matrix,

/
"MR."
nonsymmetrical matrix.

.CONL:
S V R dim=neq
neq is the number of equations of the system

This optional object is present only if there is at least a ddl of the type "LAGR":

V (ieq) = C
if ieq corresponds to a DDL named "LAGR",

1.
if not.
C is the coefficient of conditioning of the ddls of Lagrange.

.COND:
S V R dim=neq

This optional object is present only by one matrix factorized by GCPC with prepacking by the diagonal. It contains the diagonal matrix of change variables.
Attention it is a functionality currently reabsorbed.

.LIME:
List names of the MATR_ELEM having been assembled to give the MATR_ASSE.

.VALF:
This collection contains the terms of the factorized matrix when the solvor is "MULT_FRONT".

.VALI:
XD V R NUM () LONG=CSTE
There is nbloci elements in this collection.

Collection .VALI contains the lines of the initial matrix corresponding to the ddls to eliminate (those which are imposed by a "kinematic" load).
It is partitionnée in blocks constant length (like object .VALE). A line eliminated can belong only to one block of .VALI.

.ABLI:
S V I dim=nbloci+1

.ABLI (1)
:
0,

.ABLI (k+1):
number of the last ddl eliminated from the block K of .VALI.

.LLIG:
S V I dim=3*nddli + 1
nddli is the number of ddls "eliminated".

.LLIG (1)

:
nddli,

.LLIG (1+ (i-1) *3+1): ieq,

.LLIG (1+ (i-1) *3+2): jdeb,

.LLIG (1+ (i-1) *3+3): jfin.

ème
ieq is the number of the equation corresponding to I
ddl eliminated,
ème
jdeb is the number of column of the 1st term not no one of the ieq
equation,
ème
jfin is the number of column of the last term not no one of the ieq
equation.

.ALIG:
S V I dim=nddli

ème
.ALIG (I): address in .VALI (K) 1st term of I
eliminated line.
K is the number of the block containing this line.

.CONI:
S V I dim=neq

.CONI (ieq) =

1
if the ddl ieq eliminated,
0
if not.

.FETM:
S V Indirect K24 (*) dim=nbsd (a number of under-fields)

(*): MATR_ASSE not FETI (i.e. FETM (K) .REFA (5) "FETI" and for
the moment imposed on "MULT_FRONT").

Object JEVEUX listing specific SD MATR_ASSE to each under-field.
FETM (I) = name of SD MATR_ASSE of the ième under-field.

.FETF:
S V I dim=nbsd

Optional object JEVEUX indicating the character floating or not of a under-field.

.FETF (I) =
J (0<j<7)
under-field I comprises J modes of

rigid bodies.

0

nonfloating under-field.

.FETP:
XD V I LONG=nbsdf
List indices of pivots "
quasi-null
" of the floating under-fields. That is to say
V=FETP (I),

ème

ème
V (J): J
index of pivots of I
floating under-field.

The LONMAX of V (I) is equal to FETF (I).
The number of floating under-fields is accessible via attribute NUTIOC.
floating under-fields are listed in the same order as the under-fields
(floating or not) of SD SD_FETI.

.FETR:
XD V R LONG=nbsdf
Components of the rigid modes of body. That is to say V=FETR (I),

ème
ème
*V ((j-1) *nbddli+k): K*
component of I
floating under-field.

nbddli, the number of DDL of under-field I, is stored in SD_FETI.FETH (I).
The number of floating under-fields is accessible via attribute NUTIOC.
floating under-fields are listed in the same order as the under-fields
(floating or not) of SD SD_FETI.

4
Complements for FETI

4.1
Structure of recursive data MATR_ASSE

In the case of method FETI, the structure of data MATR_ASSE is recursive on two levels.
A “main” SD MATR_ASSE, concerning the total field (REFA (5) = ' FETI'), does not comprise
none usual objects JEVEUX, except the .REFA and the .LIME, with, on the other hand, in addition to the objects
specific of the decomposition of fields: .FETR, FETF, .FETP and .FETM.

This last is a pointer indicating SD MATR_ASSE "slaves" associated with each local under-fields. These local SD MATR_ASSE are consisted of same objects JEVEUX that a MATR_ASSE (.VALE, .CONL…) mono-field usual.
For the moment, the implementation of FETI in Code_Aster presupposes that these under-fields use all the same linear solvor mono-field (REFA (5) = "MULT_FRONT" imposed by defect, with thus a .VALF). This homogeneity facilitates handling of the local matrices.

under-field 1

SD MATR_ASSE

"main"

(total field)
...

.FETM

under-field I
SD MATR_ASSE

"slaves"

(under-fields)

...
Appear 4.1-a: Structure of recursive data MATR_ASSE if solvor FETI

4.2 Regulate naming

In the case of a solvor FETI, one arbitrarily chose (in ASSMAM) the rule of naming following for SD MATR_ASSE slaves dependent each on a under-field:
nom_de_la_SD_MATR_ASSE_maître (1: 11)/"F" //chaîne_de_caractères_libre (2: 8)
The free character string is generated by a call to routine GCNCON.

4.3 Particular case of parallelism MPI

During an execution in parallel mode MPI, a processor sees itself allotting a certain number of under-fields (cf objects annex "&FETI.LISTE…" structure of data SD_FETI [D4.06.21]). "Main" SD MATR_ASSE is always built, but its pointer .FETM will indicate only the under-fields concerned with the processor running: .FETM (jk) will be K24 validate that if the jk under-field is in the perimeter of the processor J.
The same applies to objects .FETF, .FETP and .FETR which are filled only if necessary.

For the processor J

vacuum

SD MATR_ASSE
j1 under-field

"main" (field

total)

.FETM

under-field J
SD MATR_ASSE "slaves"

2
(under-fields concerned with

processor J)

...

Appear 4.3-a: Structure of recursive data MATR_ASSE if solvor FETI and parallelism MPI

5 Examples

5.1
Context of examples LDLT, GCPC and MULT_FRONT

5.1.1 Grid

%
% coordinated of the nodes:
%

COOR_3D
N2 2. 0. 0.
N3 3. 0. 0.
N1 1. 0. 0.
FINSF
%
% connectivity of the meshs:
%
SEG2
M1 N1 N2
M2 N2 N3
FINSF
END

5.1.2 Orders

BEGINNING ();
email = LIRE_MAILLAGE ();
model = AFFE_MODELE (GRID: email
AFFE: (MESH: (m1, m2)
PHENOMENON: "MECHANICAL" MODELING: "DIS_T"));
carel = AFFE_CARA_ELEM (MODEL: model
DISCRETE: (CARA: "K_T_D_L" NETS: m1 VALE: (1.E5 2.E5 3.E5))
DISCRETE: (CARA: "K_T_D_L" NETS: m2 VALE: (1.E6 2.E6 3.E6)));
chbloq= AFFE_CHAR_MECA (
MODEL: model DDL_IMPO: (NODE: n1 DX: 0. DZ: 0.));
chcine= AFFE_CHAR_CINE (MODEL: model
MECA_IMPO: (NODE: n1 Dy: 7.));
rigiel=CALC_MATR_ELEM (MODEL: model CARA_ELEM: carel
CHARGE: chbloq OPTION: "rigi_meca");

5.2 Solvor
LDLT

5.2.1 Orders

nu1 = NUME_DDL (MATR_RIGI: rigiel STORAGE: "LIGN_CIEL" RENUM: "rcmk");
IMPR_CO (Co: nu1);
matas1 = ASSE_MATRICE (MATR_ELEM: rigiel NUME_DDL: nu1
CHAR_CINE: chcine);
IMPR_CO (Co: matas1);
vcine1 = CALC_CHAR_CINE (NUME_DDL: nu1 CHAR_CINE: chcine);
&MATAS1= FACT_LDLT (MATR_ASSE: matas1);
IMPR_CO (Co: matas1);

5.2.2 NUME_DDL: NU1

====> IMPR_CO OF THE STRUCTURE OF DATA: NU1????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 14
===
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >NU1 .NUME.DEEQ <
1 - 3 -1 3 -3 3
6 - 1 3 2 3 3
11 - 3 -1 3 -3 1
16 - 1 1 2 1 3
21 - 2 1 2 2 2
26 - 3

SEGMENT IMPRESSION OF VALUES >NU1 .NUME.DELG <
1 - -1 -1 0 0 0
6 - -2 -2 0 0 0
11 - 0 0 0

SEGMENT IMPRESSION OF VALUES >NU1 .NUME.LILI <
1 - >&MAILLA <>MODEL .MODELE <
3 - >CHBLOQ .CHME.LIGRE <

SEGMENT IMPRESSION OF VALUES >NU1 .NUME.LPRN <
1 - 9 0 12

SEGMENT IMPRESSION OF VALUES >NU1 .NUME.NEQU <
1 - 13 54

SEGMENT IMPRESSION OF VALUES >NU1 .NUME.NUEQ <
1 - 1 2 3 4 5
6 - 6 7 8 9 10
11 - 11 12 13

IMPRESSION OF THE COLLECTION: NU1 .NUME.PRNO
SEGMENT IMPRESSION OF VALUES >NU1 .NUME.LILI <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: NU1
1 - >&MAILLA <>MODEL .MODELE <
3 - >CHBLOQ .CHME.LIGRE <
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NU1 .NUME.PRNO< OC: 1
1 - 8 3 14 11 3
6 - 14 3 3 14
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NU1 .NUME.PRNO< OC: 3
1 - 1 1 4096 6 1

6 - 4096 2 1 4096 7
11 - 1 4096

SEGMENT IMPRESSION OF VALUES >NU1 .NUME.REFE <
1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >NU1 .SLCS.ABLO <
1 - 0 13

SEGMENT IMPRESSION OF VALUES >NU1 .SLCS.ADIA <
1 - 1 2 5 9 14
6 - 20 26 32 39 47
11 - 51 56 62

SEGMENT IMPRESSION OF VALUES >NU1 .SLCS.DESC <
1 - 13 62 1 8 0
6 - 0

SEGMENT IMPRESSION OF VALUES >NU1 .SLCS.HCOL <
1 - 1 1 3 4 5
6 - 6 6 6 7 8
11 - 4 5 6

SEGMENT IMPRESSION OF VALUES >NU1 .SLCS.IABL <
1 - 1 1 1 1 1
6 - 1 1 1 1 1
11 - 1 1 1

SEGMENT IMPRESSION OF VALUES >NU1 .SLCS.REFE <
1 - >NU1 <
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

5.2.3 MATR_ASSE after assembly: MATAS1

====> IMPR_CO OF THE STRUCTURE OF DATA: MATAS1????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 9
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
SEGMENT IMPRESSION OF VALUES >MATAS1 .ABLI <
1 - 0 1
--
SEGMENT IMPRESSION OF VALUES >MATAS1 .ALIG <
1 - 1
--
SEGMENT IMPRESSION OF VALUES >MATAS1 .CONI <
1 - 0 0 0 1 0
6 - 0 0 0 0 0
11 - 0 0 0
--
SEGMENT IMPRESSION OF VALUES >MATAS1 .CONL <
1 - 1.70000E+06 1.70000E+06 1.00000E+00 1.00000E+00 1.00000E+00
6 - 1.70000E+06 1.70000E+06 1.00000E+00 1.00000E+00 1.00000E+00
11 - 1.00000E+00 1.00000E+00 1.00000E+00
--
SEGMENT IMPRESSION OF VALUES >MATAS1 .LIME <
1 - >RIGIEL <
--
SEGMENT IMPRESSION OF VALUES >MATAS1 .LLIG <
1 - 1 4 1 10
--
SEGMENT IMPRESSION OF VALUES >MATAS1 .REFE <
1 - >MAIL <>NU1 .NUME <
3 - >NU1 .SLCS <
--
IMPRESSION OF THE COLLECTION: MATAS1 .VALE
OBJECT IMPRESSION OF COLLECTION >MATAS1 .VALE< OC: 1
1 - -1.70000E+06 -1.70000E+06 1.70000E+06 0.00000E+00 1.00000E+05
6 - 0.00000E+00 0.00000E+00 0.00000E+00 1.00000E+00 0.00000E+00
11 - 1.70000E+06 0.00000E+00 0.00000E+00 3.00000E+05 1.70000E+06
16 - 0.00000E+00 1.70000E+06 0.00000E+00 0.00000E+00 -1.70000E+06

21 - 1.70000E+06 0.00000E+00 0.00000E+00 1.70000E+06 0.00000E+00
26 - -1.70000E+06 -1.00000E+05 0.00000E+00 0.00000E+00 0.00000E+00
31 - 0.00000E+00 1.10000E+06 0.00000E+00 0.00000E+00 0.00000E+00
36 - 0.00000E+00 0.00000E+00 0.00000E+00 2.20000E+06 0.00000E+00
41 - 0.00000E+00 -3.00000E+05 0.00000E+00 0.00000E+00 0.00000E+00
46 - 0.00000E+00 3.30000E+06 -1.00000E+06 0.00000E+00 0.00000E+00
51 - 1.00000E+06 0.00000E+00 -2.00000E+06 0.00000E+00 0.00000E+00
56 - 2.00000E+06 0.00000E+00 0.00000E+00 -3.00000E+06 0.00000E+00
61 - 0.00000E+00 3.00000E+06

IMPRESSION OF THE COLLECTION: MATAS1 .VALI
OBJECT IMPRESSION OF COLLECTION >MATAS1 .VALI< OC: 1
1 - 0.00000E+00 0.00000E+00 0.00000E+00 2.00000E+05 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 -2.00000E+05 0.00000E+00
11 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
16 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
21 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
26 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
31 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
41 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
46 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
51 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
56 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
61 - 0.00000E+00 0.00000E+00

5.2.4 MATR_ASSE after factorization: MATAS1

====> IMPR_CO OF THE STRUCTURE OF DATA: MATAS1????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 10

==

IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >MATAS1 .ABLI <
1 - 0 1

SEGMENT IMPRESSION OF VALUES >MATAS1 .ALIG <
1 - 1

SEGMENT IMPRESSION OF VALUES >MATAS1 .CONI <
1 - 0 0 0 1 0
6 - 0 0 0 0 0
11 - 0 0 0

SEGMENT IMPRESSION OF VALUES >MATAS1 .CONL <
1 - 1.70000E+06 1.70000E+06 1.00000E+00 1.00000E+00 1.00000E+00
6 - 1.70000E+06 1.70000E+06 1.00000E+00 1.00000E+00 1.00000E+00
11 - 1.00000E+00 1.00000E+00 1.00000E+00

SEGMENT IMPRESSION OF VALUES >MATAS1 .LIME <
1 - >RIGIEL <

SEGMENT IMPRESSION OF VALUES >MATAS1 .LLIG <
1 - 1 4 1 10

Data-processing handbook of Description

SEGMENT IMPRESSION OF VALUES >MATAS1 .REFE <
1 - >MAIL <>NU1 .NUME <
3 - >NU1 .SLCS <

SEGMENT IMPRESSION OF VALUES >MATAS1 .TITR <
1 - >ASTER 4.00.06 CONCEPT MATAS1 CALCULATES THE 03/09/96 A 13:15: 18 OF TYPE <
*2 - >MATR_ASSE_DEPL_*

IMPRESSION OF THE COLLECTION: MATAS1 .VALE
OBJECT IMPRESSION OF COLLECTION >MATAS1 .VALE< OC: 1
1 - -1.70000E+06 -1.70000E+06 -1.00000E+00 0.00000E+00 1.80000E+06
6 - 0.00000E+00 0.00000E+00 0.00000E+00 1.00000E+00 0.00000E+00
11 - -1.00000E+00 0.00000E+00 0.00000E+00 2.00000E+06 -1.00000E+00
16 - 0.00000E+00 1.88889E+00 0.00000E+00 0.00000E+00 -6.42222E+06
21 - -1.00000E+00 0.00000E+00 0.00000E+00 1.70000E+00 0.00000E+00
26 - -5.78000E+06 -5.55556E-02 0.00000E+00 0.00000E+00 -2.94118E-02
31 - 0.00000E+00 1.10000E+06 0.00000E+00 0.00000E+00 0.00000E+00
36 - 0.00000E+00 0.00000E+00 0.00000E+00 2.20000E+06 0.00000E+00
41 - 0.00000E+00 -1.50000E-01 0.00000E+00 -8.82353E-02 0.00000E+00
46 - 0.00000E+00 3.30000E+06 -9.09091E-01 0.00000E+00 0.00000E+00
51 - 9.09091E+04 0.00000E+00 -9.09091E-01 0.00000E+00 0.00000E+00
56 - 1.81818E+05 0.00000E+00 0.00000E+00 -9.09091E-01 0.00000E+00
61 - 0.00000E+00 2.72727E+05

IMPRESSION OF THE COLLECTION: MATAS1 .VALI
OBJECT IMPRESSION OF COLLECTION >MATAS1 .VALI< OC: 1
1 - 0.00000E+00 0.00000E+00 0.00000E+00 2.00000E+05 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 -2.00000E+05 0.00000E+00
11 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
16 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
21 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
26 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
31 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
41 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00

46 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
51 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
56 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
61 - 0.00000E+00 0.00000E+00

5.3 Solvor
MULT_FRONT

5.3.1 Orders

nu2 = NUME_DDL (MATR_RIGI: rigiel STORAGE: "MORSE" RENUM: "Mandelevium");
impr_co (Co: nu2);
MATAS2 = ASSE_MATRICE (MATR_ELEM: rigiel NUME_DDL: nu2
CHAR_CINE: chcine);
impr_co (CO: matas2);
vcine2 = CALC_CHAR_CINE (NUME_DDL: nu2 CHAR_CINE: chcine);
&MATAS2= FACT_LDLT (MATR_ASSE: matas2);
impr_co (CO: matas2);

5.3.2 NUME_DDL: NU2

====> IMPR_CO OF THE STRUCTURE OF DATA: NU2????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 18
===
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >NU2 .MLTF.ADNT <
1 - 68 69 70 66 67
6 - 65 10 11 9 7
11 - 14 15 13 8 12
16 - 5 6 4 2 3
21 - 1 26 16 48 49
26 - 47 28 0 43 54
31 - 55 53 0 0 44
36 - 50 41 42 40 0
41 - 18 36 37 35 31
46 - 46 52 39 61 21
51 - 45 51 38 56

SEGMENT IMPRESSION OF VALUES >NU2 .MLTF.GLOB <
1 - 1 2 3 4 5
6 - 6 7 11 12 13
11 - 8 9 10 11 12
16 - 13 11 12 13 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0

SEGMENT IMPRESSION OF VALUES >NU2 .MLTF.LOCL <
1 - 0 0 0 0 0
6 - 0 0 1 2 3
11 - 0 0 0 1 2
16 - 3 0 0 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0

31 - 0 0 0 0 0
36 - 0 0

--
SEGMENT IMPRESSION OF VALUES >NU2 .MLTF.PNTI <
1 - 13 3 3 12 54
6 - 0 1 3 6 10
11 - 15 21 22 23 29
16 - 36 44 49 54 1
21 - 8 11 14 0 0
26 - 0 0 0 0 0
31 - 0 0 0 8 7
36 - 11 9 10 13 12
41 - 6 4 5 3 1
46 - 2 12 13 11 9
51 - 10 8 2 1 4
56 - 5 3 7 6 2
61 - 1 3 0 0 0
66 - 0 0 0 0 0
71 - 0 0 3 3 0
76 - 0 0 0 0 0
81 - 0 0 0 0 0
86 - 0 0 2 0 0
91 - 0 0 0 0 0
96 - 0 0 0 0 1
101 - 0 0 0 0 0
106 - 0 0 0 0 0
111 - 0 7 3 3 0
116 - 0 0 0 0 0
121 - 0 0 0 0 3
126 - 3 0 0 0 0
131 - 0 0 0 0 0
136 - 0 0 3 3 0
141 - 0 0 0 0 0
146 - 0 0 0 0 0

151 - 1 11 20 28 35
156 - 41 46 50 56 61
161 - 65 68 70 71 1
166 - 50 65 71 0 0
171 - 0 0 0 0 0
176 - 0 0 0 7 1
181 - 1 0 0 0 0
186 - 0 0 0 0 0
191 - 0 1 11 17 20
196 - 0 0 0 0 0
201 - 0 0 0 0 0
206 - 10 6 3 0 0
211 - 0 0 0 0 0
216 - 0 0 0 15 49
221 - 6 0 0 0 0
226 - 0 0 0 0 0
231 - 0 1 1 1 0
236 - 0 0 0 0 0
241 - 0 0 0 0 1
246 - 1 1 0 0 0
251 - 0 0 0 0 0
256 - 0 0

SEGMENT IMPRESSION OF VALUES >NU2 .NUME.DEEQ <
1 - 1 1 1 2 1
6 - 3 2 1 2 2
11 - 2 3 3 -1 3
16 - -3 3 1 3 2
21 - 3 3 3 -1 3
26 - -3

SEGMENT IMPRESSION OF VALUES >NU2 .NUME.DELG <
1 - 0 0 0 0 0
6 - 0 -1 -1 0 0
11 - 0 -2 -2

SEGMENT IMPRESSION OF VALUES >NU2 .NUME.LILI <
1 - >&MAILLA <>MODEL .MODELE <
3 - >CHBLOQ .CHME.LIGRE <

SEGMENT IMPRESSION OF VALUES >NU2 .NUME.LPRN <
1 - 9 0 12

SEGMENT IMPRESSION OF VALUES >NU2 .NUME.NEQU <
1 - 13 54

SEGMENT IMPRESSION OF VALUES >NU2 .NUME.NUEQ <
1 - 1 2 3 4 5
6 - 6 7 8 9 10
11 - 11 12 13

IMPRESSION OF THE COLLECTION: NU2 .NUME.PRNO
SEGMENT IMPRESSION OF VALUES >NU2 .NUME.LILI <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: NU2
1 - >&MAILLA <>MODEL .MODELE <

3 - >CHBLOQ .CHME.LIGRE <
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NU2 .NUME.PRNO< OC: 1
1 - 1 3 14 4 3
6 - 14 9 3 14
OBJECT IMPRESSION OF COLLECTION CONTIGUE>NU2 .NUME.PRNO< OC: 3
1 - 7 1 4096 12 1
6 - 4096 8 1 4096 13
11 - 1 4096

SEGMENT IMPRESSION OF VALUES >NU2 .NUME.REFE <
1 - >MAIL <
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

SEGMENT IMPRESSION OF VALUES >NU2 .SMOS.ABLO <
1 - 0 13

SEGMENT IMPRESSION OF VALUES >NU2 .SMOS.ADIA <
1 - 1 3 6 10 15
6 - 21 22 23 29 36
11 - 44 49 54

SEGMENT IMPRESSION OF VALUES >NU2 .SMOS.DESC <
1 - 13 54 1 8 0
6 - 0

SEGMENT IMPRESSION OF VALUES >NU2 .SMOS.HCOL <
1 - 1 1 2 1 2
6 - 3 1 2 3 4
11 - 1 2 3 4 5
16 - 1 2 3 4 5
21 - 6 7 8 1 2
26 - 3 7 8 9 1
31 - 2 3 7 8 9
36 - 10 1 2 3 7
41 - 8 9 10 11 7

46 - 9 10 11 12 8
51 - 9 10 11 13

SEGMENT IMPRESSION OF VALUES >NU2 .SMOS.IABL <
1 - 1 1 1 1 1
6 - 1 1 1 1 1
11 - 1 1 1

SEGMENT IMPRESSION OF VALUES >NU2 .SMOS.REFE <
1 - >NU2 <

5.3.3 MATR_ASSE after assembly: MATAS2

====> IMPR_CO OF THE STRUCTURE OF DATA: MATAS2????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 9

==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >MATAS2 .ABLI <
1 - 0 1

SEGMENT IMPRESSION OF VALUES >MATAS2 .ALIG <
1 - 1

SEGMENT IMPRESSION OF VALUES >MATAS2 .CONI <
1 - 0 0 0 0 0
6 - 0 0 0 0 1
11 - 0 0 0

SEGMENT IMPRESSION OF VALUES >MATAS2 .CONL <
1 - 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00
6 - 1.00000E+00 1.70000E+06 1.70000E+06 1.00000E+00 1.00000E+00
11 - 1.00000E+00 1.70000E+06 1.70000E+06

SEGMENT IMPRESSION OF VALUES >MATAS2 .LIME <
1 - >RIGIEL <

SEGMENT IMPRESSION OF VALUES >MATAS2 .LLIG <
1 - 1 10 1 13

SEGMENT IMPRESSION OF VALUES >MATAS2 .REFE <
1 - >MAIL <>NU2 .NUME <
3 - >NU2 .SMOS <

IMPRESSION OF THE COLLECTION: MATAS2 .VALE
OBJECT IMPRESSION OF COLLECTION >MATAS2 .VALE< OC: 1
1 - 1.10000E+06 0.00000E+00 2.20000E+06 0.00000E+00 0.00000E+00
6 - 3.30000E+06 -1.00000E+06 0.00000E+00 0.00000E+00 1.00000E+06
11 - 0.00000E+00 -2.00000E+06 0.00000E+00 0.00000E+00 2.00000E+06
16 - 0.00000E+00 0.00000E+00 -3.00000E+06 0.00000E+00 0.00000E+00
21 - 3.00000E+06 -1.70000E+06 -1.70000E+06 -1.00000E+05 0.00000E+00
26 - 0.00000E+00 1.70000E+06 0.00000E+00 1.00000E+05 0.00000E+00
31 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - 1.00000E+00 0.00000E+00 0.00000E+00 -3.00000E+05 0.00000E+00
41 - 1.70000E+06 0.00000E+00 0.00000E+00 3.00000E+05 1.70000E+06
46 - 1.70000E+06 0.00000E+00 0.00000E+00 -1.70000E+06 1.70000E+06

51 - 0.00000E+00 0.00000E+00 1.70000E+06 -1.70000E+06
--
IMPRESSION OF THE COLLECTION: MATAS2 .VALI
OBJECT IMPRESSION OF COLLECTION >MATAS2 .VALI< OC: 1
1 - 0.00000E+00 -2.00000E+05 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 2.00000E+05
11 - 0.00000E+00 0.00000E+00 0.00000E+00

5.3.4 MATR_ASSE after factorization: MATAS2

====> IMPR_CO OF THE STRUCTURE OF DATA: MATAS2?????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 11
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
SEGMENT IMPRESSION OF VALUES >MATAS2 .ABLI <

---

1 - 0 1
--
SEGMENT IMPRESSION OF VALUES >MATAS2 .ALIG <
1 - 1
--
SEGMENT IMPRESSION OF VALUES >MATAS2 .CONI <
1 - 0 0 0 0 0
6 - 0 0 0 0 1
11 - 0 0 0
--
SEGMENT IMPRESSION OF VALUES >MATAS2 .CONL <
1 - 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00
6 - 1.00000E+00 1.70000E+06 1.70000E+06 1.00000E+00 1.00000E+00
11 - 1.00000E+00 1.70000E+06 1.70000E+06
--
SEGMENT IMPRESSION OF VALUES >MATAS2 .LIME <
1 - >RIGIEL <
--

SEGMENT IMPRESSION OF VALUES >MATAS2 .LLIG <
1 - 1 10 1 13
--
SEGMENT IMPRESSION OF VALUES >MATAS2 .REFE <
1 - >MAIL <>NU2 .NUME <
3 - >NU2 .SMOS <
--
SEGMENT IMPRESSION OF VALUES >MATAS2 .TITR <
1 - >ASTER 4.00.06 CONCEPT MATAS2 CALCULATES THE 03/09/96 A 13:15: 18 OF TYPE <
2 - >MATR_ASSE_DEPL_R <
--
IMPRESSION OF THE COLLECTION: MATAS2 .VALE
OBJECT IMPRESSION OF COLLECTION >MATAS2 .VALE< OC: 1
1 - 1.10000E+06 0.00000E+00 2.20000E+06 0.00000E+00 0.00000E+00
6 - 3.30000E+06 -1.00000E+06 0.00000E+00 0.00000E+00 1.00000E+06
11 - 0.00000E+00 -2.00000E+06 0.00000E+00 0.00000E+00 2.00000E+06
16 - 0.00000E+00 0.00000E+00 -3.00000E+06 0.00000E+00 0.00000E+00
21 - 3.00000E+06 -1.70000E+06 -1.70000E+06 -1.00000E+05 0.00000E+00
26 - 0.00000E+00 1.70000E+06 0.00000E+00 1.00000E+05 0.00000E+00
31 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - 1.00000E+00 0.00000E+00 0.00000E+00 -3.00000E+05 0.00000E+00
41 - 1.70000E+06 0.00000E+00 0.00000E+00 3.00000E+05 1.70000E+06
46 - 1.70000E+06 0.00000E+00 0.00000E+00 -1.70000E+06 1.70000E+06
51 - 0.00000E+00 0.00000E+00 1.70000E+06 -1.70000E+06
--
IMPRESSION OF THE COLLECTION: MATAS2 .VALF
OBJECT IMPRESSION OF COLLECTION >MATAS2 .VALF< OC: 1
1 - 3.00000E+06 0.00000E+00 0.00000E+00 -1.00000E+00 0.00000E+00
6 - 0.00000E+00 1.00000E+06 0.00000E+00 0.00000E+00 -1.00000E+00
11 - 0.00000E+00 2.00000E+06 0.00000E+00 0.00000E+00 -1.00000E+00
OBJECT IMPRESSION OF COLLECTION >MATAS2 .VALF< OC: 2
1 - -1.70000E+06 0.00000E+00 -1.00000E+00 0.00000E+00 0.00000E+00
6 - -1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - -1.70000E+06 0.00000E+00 -1.00000E+00 0.00000E+00 0.00000E+00
16 - -1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 2.00000E+06
21 - 0.00000E+00 0.00000E+00 1.70000E+00 0.00000E+00 -1.50000E-01
26 - 0.00000E+00 0.00000E+00 1.80000E+06 0.00000E+00 0.00000E+00
31 - 1.88889E+00 0.00000E+00 -5.55556E-02 0.00000E+00 1.00000E+00
36 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
41 - -5.78000E+06 0.00000E+00 -8.82353E-02 0.00000E+00 0.00000E+00
46 - -6.42222E+06 0.00000E+00 -2.94118E-02 0.00000E+00
OBJECT IMPRESSION OF COLLECTION >MATAS2 .VALF< OC: 3
1 - 3.00000E+05 0.00000E+00 0.00000E+00 1.00000E+05 0.00000E+00
6 - 2.00000E+05
--
IMPRESSION OF THE COLLECTION: MATAS2 .VALI
OBJECT IMPRESSION OF COLLECTION >MATAS2 .VALI< OC: 1
1 - 0.00000E+00 -2.00000E+05 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 2.00000E+05
11 - 0.00000E+00 0.00000E+00 0.00000E+00
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

5.4 Solvor GCPC

5.4.1 Orders

nu3 = NUME_DDL (MATR_RIGI: rigiel STORAGE: "Morse" RENUM: "without");
impr_co (Co: nu3);
matas3 = ASSE_MATRICE (MATR_ELEM: rigiel NUME_DDL: nu3
CHAR_CINE: chcine);
IMPR_CO (CO: matas3);
vcine3 = CALC_CHAR_CINE (NUME_DDL: nu3 CHAR_CINE: chcine);
fatas3 = FACT_GRAD (MATR_ASSE: matas3);
IMPR_CO (CO: fatas3);

5.4.2 NUME_DDL: NU3

====> IMPR_CO OF THE STRUCTURE OF DATA: NU3????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 14
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
SEGMENT IMPRESSION OF VALUES >NU3 .NUME.DEEQ <
1 - 1 1 1 2 1
6 - 3 2 1 2 2
11 - 2 3 3 -1 3
16 - -3 3 1 3 2
21 - 3 3 3 -1 3
26 - -3
--
SEGMENT IMPRESSION OF VALUES >NU3 .NUME.DELG <
1 - 0 0 0 0 0
6 - 0 -1 -1 0 0
11 - 0 -2 -2
--
SEGMENT IMPRESSION OF VALUES >NU3 .NUME.LILI <
1 - >&MAILLA <>MODEL .MODELE <
3 - >CHBLOQ .CHME.LIGRE <
--
SEGMENT IMPRESSION OF VALUES >NU3 .NUME.LPRN <
1 - 9 0 12

SEGMENT IMPRESSION OF VALUES >NU3 .NUME.NEQU <

1 - 13 54

SEGMENT IMPRESSION OF VALUES >NU3 .NUME.NUEQ <

1 - 1 2 3 4 5

6 - 6 7 8 9 10

11 - 11 12 13

IMPRESSION OF THE COLLECTION: NU3 .NUME.PRNO

SEGMENT IMPRESSION OF VALUES >NU3 .NUME.LILI <

>>>>> REPERTORY OF NAMES OF THE COLLECTION: NU3

1 - >&MAILLA <>MODEL .MODELE <

3 - >CHBLOQ .CHME.LIGRE <

OBJECT IMPRESSION OF COLLECTION CONTIGUE>NU3 .NUME.PRNO< OC: 1

1 - 1 3 14 4 3

6 - 14 9 3 14

OBJECT IMPRESSION OF COLLECTION CONTIGUE>NU3 .NUME.PRNO< OC: 3

1 - 7 1 4096 12 1

6 - 4096 8 1 4096 13

11 - 1 4096

SEGMENT IMPRESSION OF VALUES >NU3 .NUME.REFE <

1 - >MAIL <

SEGMENT IMPRESSION OF VALUES >NU3 .SMOS.ABLO <

1 - 0 13

SEGMENT IMPRESSION OF VALUES >NU3 .SMOS.ADIA <

1 - 1 3 6 10 15

6 - 21 22 23 29 36

11 - 44 49 54

SEGMENT IMPRESSION OF VALUES >NU3 .SMOS.DESC <

1 - 13 54 1 8 0

6 - 0

SEGMENT IMPRESSION OF VALUES >NU3 .SMOS.HCOL <

1 - 1 1 2 1 2

6 - 3 1 2 3 4

11 - 1 2 3 4 5

16 - 1 2 3 4 5

21 - 6 7 8 1 2

26 - 3 7 8 9 1

31 - 2 3 7 8 9

36 - 10 1 2 3 7

41 - 8 9 10 11 7

46 - 9 10 11 12 8

51 - 9 10 11 13

SEGMENT IMPRESSION OF VALUES >NU3 .SMOS.IABL <

---

1 - 1 1 1 1 1
6 - 1 1 1 1 1
11 - 1 1 1
--
SEGMENT IMPRESSION OF VALUES >NU3 .SMOS.REFE <
1 - >NU3 <

5.4.3 MATR_ASSE after assembly: MATAS3

====> IMPR_CO OF THE STRUCTURE OF DATA: MATAS3?????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 9
===
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
SEGMENT IMPRESSION OF VALUES >MATAS3 .ABLI <
1 - 0 1
--
SEGMENT IMPRESSION OF VALUES >MATAS3 .ALIG <
1 - 1
--
SEGMENT IMPRESSION OF VALUES >MATAS3 .CONI <
1 - 0 0 0 0 0
6 - 0 0 0 0 1
11 - 0 0 0
--
SEGMENT IMPRESSION OF VALUES >MATAS3 .CONL <
1 - 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00
6 - 1.00000E+00 1.70000E+06 1.70000E+06 1.00000E+00 1.00000E+00
11 - 1.00000E+00 1.70000E+06 1.70000E+06
--
SEGMENT IMPRESSION OF VALUES >MATAS3 .LIME <
1 - >RIGIEL <
--
SEGMENT IMPRESSION OF VALUES >MATAS3 .LLIG <
1 - 1 10 1 13
--
SEGMENT IMPRESSION OF VALUES >MATAS3 .REFE <
1 - >MAIL <>NU3 .NUME <

3 - >NU3 .SMOS <

IMPRESSION OF THE COLLECTION: MATAS3 .VALE
OBJECT IMPRESSION OF COLLECTION >MATAS3 .VALE< OC: 1
1 - 1.10000E+06 0.00000E+00 2.20000E+06 0.00000E+00 0.00000E+00
6 - 3.30000E+06 -1.00000E+06 0.00000E+00 0.00000E+00 1.00000E+06
11 - 0.00000E+00 -2.00000E+06 0.00000E+00 0.00000E+00 2.00000E+06
16 - 0.00000E+00 0.00000E+00 -3.00000E+06 0.00000E+00 0.00000E+00
21 - 3.00000E+06 -1.70000E+06 -1.70000E+06 -1.00000E+05 0.00000E+00
26 - 0.00000E+00 1.70000E+06 0.00000E+00 1.00000E+05 0.00000E+00
31 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - 1.00000E+00 0.00000E+00 0.00000E+00 -3.00000E+05 0.00000E+00
41 - 1.70000E+06 0.00000E+00 0.00000E+00 3.00000E+05 1.70000E+06
46 - 1.70000E+06 0.00000E+00 0.00000E+00 -1.70000E+06 1.70000E+06
51 - 0.00000E+00 0.00000E+00 1.70000E+06 -1.70000E+06

IMPRESSION OF THE COLLECTION: MATAS3 .VALI
OBJECT IMPRESSION OF COLLECTION >MATAS3 .VALI< OC: 1
1 - 0.00000E+00 -2.00000E+05 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 2.00000E+05
11 - 0.00000E+00 0.00000E+00 0.00000E+00

5.4.4 MATR_ASSE of prepacking: FATAS3

====> IMPR_CO OF THE STRUCTURE OF DATA: FATAS3????????????????
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 2
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >FATAS3 .REFE <
1 - >MAIL <>NU3 .NUME <
3 - >NU3 .SMOS <

IMPRESSION OF THE COLLECTION: FATAS3 .VALE
OBJECT IMPRESSION OF COLLECTION >FATAS3 .VALE< OC: 1
1 - 9.09091E-07 0.00000E+00 4.54545E-07 0.00000E+00 0.00000E+00
6 - 3.03030E-07 -9.09091E-01 0.00000E+00 0.00000E+00 1.10000E-05
11 - 0.00000E+00 -9.09091E-01 0.00000E+00 0.00000E+00 5.50000E-06
16 - 0.00000E+00 0.00000E+00 -9.09091E-01 0.00000E+00 0.00000E+00
21 - 3.66667E-06 -5.88235E-07 -5.88235E-07 -9.09091E-02 0.00000E+00
26 - 0.00000E+00 -1.00000E+00 0.00000E+00 5.58376E-07 0.00000E+00
31 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - 1.00000E+00 0.00000E+00 0.00000E+00 -9.09091E-02 0.00000E+00
41 - -1.00000E+00 0.00000E+00 0.00000E+00 5.06912E-07 -1.00000E+00
46 - 1.89848E+00 0.00000E+00 0.00000E+00 -1.54923E-07 -1.00000E+00
51 - 0.00000E+00 0.00000E+00 1.72350E+00 -1.70651E-07

5.5 Solvor
FETI

The series of following orders (resulting from case-test FETI002A)

BEGINNING (CODE=_F (NAME = ' FETI002A', NIV_PUB_WEB=' INTRANET'))
MATER=DEFI_MATERIAU (
ELAS=_F (E = 180000. , NAKED = 0.30, ALPHA = 15.E-6, RHO = 7700. ,))
MAIL=LIRE_MAILLAGE ()
MODM=AFFE_MODELE (MAILLAGE=MAIL,
AFFE= (_F (GROUP_MA = "STRU", PHENOMENON = "MECHANICAL",
MODELING = "D_PLAN",),
_F (GROUP_MA = "POISCR",
MODELING = "2D_DIS_T", PHENOMENE=' MECANIQUE'),
_F (GROUP_MA = "POIACR",
MODELING = "2D_DIS_T", PHENOMENE=' MECANIQUE'),))
CHCAR=AFFE_CARA_ELEM (MODELE=MODM,
DISCRET= (
_F (GROUP_MA=' POIACR', CARA = "K_T_N", VALE = (0. , 0. , 0. ,),),
_F (GROUP_MA=' POISCR', CARA = "K_T_N", VALE = (180000. , 0. , 180000. ,),),))
CHMAT=AFFE_MATERIAU (MAILLAGE=MAIL,
AFFE= (_F (TOUT=' OUI', MATER=MATER, TEMP_REF=20.,),))
CH1=AFFE_CHAR_MECA (MODELE=MODM,
PRES_REP= (_F (GROUP_MA=' DDLI', CLOSE = 1000. ,),
_F (GROUP_MA=' DDLI1', CLOSE = 2000. ,),),)
SDFETI=DEFI_PART_OPS (NOM=' SD',
MODELE=MODM,
INFO=1,
DEFI= (_F (GROUP_MA = "FETI1", GROUP_MA_BORD = "B1",),
_F (GROUP_MA = "FETI2", GROUP_MA_BORD = "B2",),
_F (GROUP_MA = "FETI3", GROUP_MA_BORD = "B3",),
_F (GROUP_MA = "FETI4", GROUP_MA_BORD = "B4",),),);
RESU=MECA_STATIQUE (MODELE=MODM,
CARA_ELEM=CHCAR,
CHAM_MATER=CHMAT,
SOLVEUR=_F (METHODE=' FETI',
PARTITION=SDFETI,),
EXCIT= (_F (CHARGE=CH1,),),)

Built a "main" SD MATR_ASSE (after factorization) "&&MESTAT_MATR_ASSEM"…

====> IMPR_CO OF THE STRUCTURE OF DATA: &&MESTAT_MATR_ASSEM?????
ATTRIBUTE: F CONTENTS: T BASE: > <
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 7

==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MATR_ASSEM.FETF <
1 - -1 3 -1 3

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MATR_ASSEM.FETM <
1 - >&&MESTAT_MAF0000085 <>&&MESTAT_MAF0000088 <
3 - >&&MESTAT_MAF0000092 <>&&MESTAT_MAF0000096 <

IMPRESSION OF THE COLLECTION: &&MESTAT_MATR_ASSEM.FETP
SEGMENT IMPRESSION OF VALUES >&&MESTAT_MATR_ASSEM.FETP$$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: &&MESTAT_MATR
1 - >SD2 <>SD4 <
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MATR_ASSEM.FETP< OC: 1
1 - 9 10 12
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MATR_ASSEM.FETP< OC: 2
1 - 2 9 10

IMPRESSION OF THE COLLECTION: &&MESTAT_MATR_ASSEM.FETR
SEGMENT IMPRESSION OF VALUES >&&MESTAT_MATR_ASSEM.FETR$$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: &&MESTAT_MATR
1 - >SD2 <>SD4 <
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MATR_ASSEM.FETR< OC: 1
1 - -1.00000D+00 -2.22045D-16 -1.00000D+00 9.71445D-17 -1.00000D+00
6 - 3.60822D-16 -1.00000D+00 -1.38778D-17 -1.00000D+00 0.00000D+00
11 - -1.00000D+00 0.00000D+00 -1.00000D+00 3.33067D-16 2.50000D-01
16 - 7.50000D-01 1.75000D+00 -1.25000D+00 -2.50000D-01 -2.75000D+00
21 - -1.75000D+00 -7.50000D-01 0.00000D+00 -1.00000D+00 -7.50000D-01
26 - 0.00000D+00 7.50000D-01 -2.00000D+00 -2.50000D-01 -1.75000D+00
31 - -1.75000D+00 2.50000D-01 2.50000D-01 1.75000D+00 1.75000D+00
36 - -2.50000D-01 0.00000D+00 0.00000D+00 7.50000D-01 -1.00000D+00
41 - -7.50000D-01 1.00000D+00
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MATR_ASSEM.FETR< OC: 2
1 - 7.00000D+00 -1.00000D+00 -1.00000D+00 -7.00000D+00 1.00000D+00
6 - 7.00000D+00 -7.00000D+00 1.00000D+00 0.00000D+00 0.00000D+00
11 - 3.00000D+00 -4.00000D+00 -3.00000D+00 4.00000D+00 -1.00000D+00

---

16 - 0.00000D+00 -1.00000D+00 -3.10862D-15 -1.00000D+00 3.38618D-15
21 - -1.00000D+00 5.27356D-16 -1.00000D+00 0.00000D+00 -1.00000D+00
26 - -1.77636D-15 -1.00000D+00 1.88738D-15 -7.00000D+00 0.00000D+00
31 - 1.00000D+00 6.00000D+00 -1.00000D+00 -8.00000D+00 7.00000D+00
36 - -2.00000D+00 0.00000D+00 -1.00000D+00 -3.00000D+00 3.00000D+00
41 - 3.00000D+00 -5.00000D+00

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MATR_ASSEM.LILI <
1 - >&MAILLA <>MODM .MODELE <

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MATR_ASSEM.LIME <
1 - >&&MATELE<

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MATR_ASSEM.REFA <
1 - >MAIL <>RESU .00000.NUME <
3 - >RESU .00000.SMOS <> <
5 - >FETI <>SDFETI <

… and of SD MATR_ASSE “slaves” (after factorization) “&&MESTAT_MAF0000085”…

====> IMPR_CO OF THE STRUCTURE OF DATA: &&MESTAT_MAF0000085?????
ATTRIBUTE: F CONTENTS: T BASE: > <
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 5
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MAF0000085.&IN2 <
1 - >&&MESTAT_MAF0000085 <

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MAF0000085.&INT <
1 - 19 2072258 14 1 1
6 - 0 2 0 0 0
11 - ************ ************ ************ 1 69
16 - ************ ************ ************ 0

SEGMENT IMPRESSION OF VALUES >&&MESTAT_MAF0000085.REFA <
1 - >MAIL <>RESU F0000050.NUME <
3 - >RESU F0000050.SMOS <>RIGI_MECA <
5 - >MULT_FRO <

IMPRESSION OF THE COLLECTION: >&&MESTAT_MAF0000085.VALE
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MAF0000085.VALE< OC: 1
1 - 1.02115D+05 -2.94231D+04 2.09423D+05 3.65192D+05 -5.36538D+04
6 - 3.06346D+05 -1.21154D+04 -5.01923D+04 1.85192D+05 -3.28846D+04
11 - 1.21154D+04 -5.36538D+04 1.26346D+05 -1.21154D+04 -3.28846D+04
16 - 1.02115D+05 -5.01923D+04 1.21154D+04 -2.94231D+04 2.09423D+05
21 - 1.73077D+04 1.73077D+04 -1.48846D+05 6.57692D+04 -1.48846D+05
26 - 6.57692D+04 1.73077D+04 1.73077D+04 6.23077D+05 1.73077D+04
31 - -1.73077D+05 6.57692D+04 -6.92308D+03 6.57692D+04 -6.92308D+03
36 - 1.73077D+04 -1.73077D+05 0.00000D+00 6.23077D+05 -1.07308D+05
41 - 6.23077D+04 -2.42308D+04 2.07692D+04 -1.80000D+05 -8.30769D+04
46 - 4.91538D+05 4.50000D+04 -4.84615D+04 3.80769D+04 -1.31538D+05
51 - -8.30769D+04 -1.31538D+05 0.00000D+00 4.91538D+05 -2.42308D+04

56 - 2.07692D+04 -1.07308D+05 6.23077D+04 -1.80000D+05 -8.30769D+04
61 - 3.11538D+05 3.80769D+04 -1.31538D+05 4.50000D+04 -4.84615D+04
66 - -8.30769D+04 -1.31538D+05 0.00000D+00 3.11538D+05

IMPRESSION OF THE COLLECTION: >&&MESTAT_MAF0000085.VALF
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MAF0000085.VALF< OC: 1
1 - 4.91538D+05 0.00000D+00 -4.92958D-02 4.22535D-02 -2.18310D-01
6 - 1.26761D-01 -3.66197D-01 -1.69014D-01 0.00000D+00 4.91538D+05
11 - 7.74648D-02 -2.67606D-01 9.15493D-02 -9.85915D-02 -1.69014D-01
16 - -2.67606D-01 3.61048D+05 -1.17548D-01 -3.35561D-02 -1.39018D-01
21 - -2.43063D-02 1.89049D-02 -4.19013D-01 1.99041D-01 0.00000D+00
26 - 2.65279D+05 -1.29331D-01 2.34296D-02 5.85979D-02 -5.57864D-02
31 - 1.25754D-01 -1.13713D-01 3.11538D+05 0.00000D+00 -7.77778D-02
36 - 6.66667D-02 -3.44444D-01 2.00000D-01 -5.77778D-01 -2.66667D-01
41 - 0.00000D+00 3.11538D+05 1.22222D-01 -4.22222D-01 1.44444D-01
46 - -1.55556D-01 -2.66667D-01 -4.22222D-01
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MAF0000085.VALF< OC: 2
1 - 1.78654D+05 -2.01292D-01 -7.75027D-02 6.02799D-02 -6.78149D-02
6 - -2.80947D-01 -8.54682D-01 4.21959D-01 0.00000D+00 6.21843D+04
11 - 3.75766D-01 -3.60985D-01 -5.68043D-01 3.23569D-02 1.92277D-01
16 - -6.71372D-01 0.00000D+00 0.00000D+00 4.39566D+04 1.68979D-01
21 - 3.19639D-01 -1.44029D-01 -1.13251D+00 6.30910D-01 0.00000D+00
26 - 0.00000D+00 0.00000D+00 1.72292D+05 -9.27453D-02 3.55056D-02
31 - 2.35090D-01 -1.10584D+00 0.00000D+00 0.00000D+00 0.00000D+00
36 - 0.00000D+00 4.65854D+04 -2.15106D-01 -8.39044D-02 -6.52780D-01
41 - 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00 0.00000D+00
46 - 1.78341D+05 -9.01603D-02 -8.51224D-01
OBJECT IMPRESSION OF COLLECTION >&&MESTAT_MAF0000085.VALF< OC: 3
1 - 1.48901D+05 2.05445D-01 0.00000D+00 3.50234D+04
...

They are pressed on “main” SD NUME_DDL (“RESU”) and “slaves” (`RESU F00000… '') following [D4.06.07]

====> IMPR_CO OF THE STRUCTURE OF DATA: RESU .00000??????????

ATTRIBUTE: F CONTENTS: T BASE: > <
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 11
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.FETN <
1 - >RESU F0000050 <>RESU F0000052 <
3 - >RESU F0000054 <>RESU F0000056 <
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NEWN <
1 - 1 2 3 4 5
6 - 6 7 8 9 10
11 - 11 12 13 14 15
16 - 16 17 18 19
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NUME.DEEQ <
1 - 1 1 1 2 2
6 - 1 2 2 3 1
11 - 3 2 4 1 4
16 - 2 5 1 5 2
21 - 6 1 6 2 7
26 - 1 7 2 8 1
31 - 8 2 9 1 9
36 - 2 10 1 10 2
41 - 11 1 11 2 12
46 - 1 12 2 13 1
51 - 13 2 14 1 14
56 - 2 15 1 15 2
61 - 16 1 16 2 17
66 - 1 17 2 18 1
71 - 18 2 19 1 19
76 - 2
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NUME.DELG <
1 - 0 0 0 0 0
6 - 0 0 0 0 0
11 - 0 0 0 0 0
16 - 0 0 0 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NUME.LILI <
1 - >&MAILLA <>MODM .MODELE <
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NUME.LPRN <
1 - 95 0
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NUME.NEQU <
1 - 38 361
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NUME.NUEQ <
1 - 1 2 3 4 5
6 - 6 7 8 9 10

11 - 11 12 13 14 15
16 - 16 17 18 19 20
21 - 21 22 23 24 25
26 - 26 27 28 29 30
31 - 31 32 33 34 35
36 - 36 37 38
--
IMPRESSION OF THE COLLECTION: RESU .00000.NUME.PRNO
OBJECT IMPRESSION OF COLLECTION CONTIGUE>RESU .00000.NUME.PRNO< OC: 1
1 - 1 2 6 0 0
6 - 3 2 6 0 0
11 - 5 2 6 0 0
16 - 7 2 6 0 0
21 - 9 2 6 0 0
26 - 11 2 6 0 0
31 - 13 2 6 0 0
36 - 15 2 6 0 0
41 - 17 2 6 0 0
46 - 19 2 6 0 0
51 - 21 2 6 0 0
56 - 23 2 6 0 0
61 - 25 2 6 0 0
66 - 27 2 6 0 0
71 - 29 2 6 0 0
76 - 31 2 6 0 0
81 - 33 2 6 0 0
86 - 35 2 6 0 0
91 - 37 2 6 0 0
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.NUME.REFN <
1 - >MAIL <>DEPL_R <
3 - >FETI <>SDFETI <
--
SEGMENT IMPRESSION OF VALUES >RESU .00000.OLDN <
1 - 1 2 3 4 5
6 - 6 7 8 9 10
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

---

11 - 11 12 13 14 15
16 - 16 17 18 19

====> IMPR_CO OF THE STRUCTURE OF DATA: RESU F0000050??????????
ATTRIBUTE: F CONTENTS: T BASE: > <
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 36
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.ADNT <
1 - 85 86 94 17 18
6 - 26 -53 -54 49 -61
11 - -62 50 58 19 27
16 - 67 20 28 68 76
21 - 87 95 23 31 55
26 - 63 71 79 97 88
31 - 96 24 32 56 64
36 - 72 80 98 100 -5
41 - -6 -3 -4 -7 -8
46 - 1 -13 -14 -11 -12
51 - -15 -16 2 10 -35
56 - -36 -37 -38 -39 -40
61 - 33 -43 -44 -45 -46
66 - -47 -48 34 42
--
SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.ADPI <
1 - 1 1 22 1 1
--
SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.ADRE <
1 - 1 9 17 25 33
6 - 35 0 0 0 0
11 - 0 0 0 0 0
--
SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.ANCI <
1 - 11 12 3 4 13
6 - 14 5 6 7 8
11 - 1 2 9 10
--
SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.DECA <
1 - 1 17 33 1 1

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.DESC <
1 - 14 5 3 45 69

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.FILS <
1 - 0 1 0 2 4

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.FRER <
1 - 0 3 0 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.GLOB <
1 - 1 2 3 4 11
6 - 12 13 14 3 4
11 - 9 10 11 12 13
16 - 14 5 6 7 8
21 - 9 10 13 14 7
26 - 8 9 10 11 12
31 - 13 14 13 14

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.LFRN <
1 - 6 6 6 2 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.LGBL <
1 - 48 48 4 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.LGSN <
1 - 2 2 2 6 2

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.LOCL <
1 - 0 0 1 2 5
6 - 6 7 8 0 0
11 - 3 4 5 6 7
16 - 8 0 0 1 2
21 - 3 4 7 8 0
26 - 0 0 0 0 0
31 - 1 2 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.NBAS <
1 - 2 4 4 2 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.NBLI <
1 - 8 8 8 8 2

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.NCBL <
1 - 3 1 1 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.NOUV <
1 - 11 12 3 4 7
6 - 8 9 10 13 14
11 - 1 2 5 6

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.RENU <
1 - >METIS <

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.SEQU <
1 - 1 2 3 4 5
6 - 0 0 0 0 0
11 - 0 0 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.MLTF.SUPN <
1 - 1 3 5 7 13
6 - 15 0 0 0 0
11 - 0 0 0 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NEWN <
1 - 1 2 3 4 0
6 - 0 0 0 0 5
11 - 0 0 0 6 0
16 - 0 7 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NUME.DEEQ <
1 - 1 1 1 2 2
6 - 1 2 2 3 1
11 - 3 2 4 1 4
16 - 2 10 1 10 2
21 - 14 1 14 2 17
26 - 1 17 2

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NUME.DELG <
1 - 0 0 0 0 0
6 - 0 0 0 0 0
11 - 0 0 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NUME.LILI <
1 - >&MAILLA <>&F000012.MODELE <

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NUME.LPRN <
1 - 95 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NUME.NEQU <
1 - 14 93

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NUME.NUEQ <
1 - 1 2 3 4 5
6 - 6 7 8 9 10
11 - 11 12 13 14

IMPRESSION OF THE COLLECTION: RESU F0000050.NUME.PRNO
OBJECT IMPRESSION OF COLLECTION CONTIGUE>RESU F0000050.NUME.PRNO< OC: 1
1 - 1 2 6 0 0
6 - 3 2 6 0 0
11 - 5 2 6 0 0
16 - 7 2 6 0 0
21 - 0 0 0 0 0
26 - 0 0 0 0 0
31 - 0 0 0 0 0
36 - 0 0 0 0 0
41 - 0 0 0 0 0
46 - 9 2 6 0 0
51 - 0 0 0 0 0
56 - 0 0 0 0 0
61 - 0 0 0 0 0
66 - 11 2 6 0 0
71 - 0 0 0 0 0
76 - 0 0 0 0 0
81 - 13 2 6 0 0
86 - 0 0 0 0 0
91 - 0 0 0 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.NUME.REFN <
1 - >MAIL <>DEPL_R <
3 - >MULT_FRO <

SEGMENT IMPRESSION OF VALUES >RESU F0000050.OLDN <
1 - 1 2 3 4 10
6 - 14 17 0 0 0
11 - 0 0 0 0 0
16 - 0 0 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.SMOS.ABLO <
1 - 0 14

SEGMENT IMPRESSION OF VALUES >RESU F0000050.SMOS.ADIA <
1 - 1 3 4 6 9
6 - 13 16 20 29 39
11 - 46 54 61 69

SEGMENT IMPRESSION OF VALUES >RESU F0000050.SMOS.DESC <
1 - 14 69 1 10 0
6 - 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.SMOS.HCOL <
1 - 1 1 2 3 3
Data-processing handbook of Description

---

6 - 4 1 2 5 1
11 - 2 5 6 3 4
16 - 7 3 4 7 8
21 - 1 2 3 4 5
26 - 6 7 8 9 1
31 - 2 3 4 5 6
36 - 7 8 9 10 1
41 - 2 3 4 9 10
46 - 11 1 2 3 4
51 - 9 10 11 12 5
56 - 6 7 8 9 10
61 - 13 5 6 7 8
66 - 9 10 13 14 0
71 - 0 0 0 0

SEGMENT IMPRESSION OF VALUES >RESU F0000050.SMOS.IABL <
1 - 1 1 1 1 1
6 - 1 1 1 1 1
11 - 1 1 1 1

SEGMENT IMPRESSION OF VALUES >RESU F0000050.SMOS.REFE <
1 - >RESU F0000050 <
...

Code_Aster ®
Version
8.1

Titrate:
Structure of Data MATR_ASSE

Intentionally white left page.

Titrate:
Structure of Data SOLVEUR

Date:
29/09/05
Author (S):
***J.** Key **PELLET**, O.BOITEAU*
:
D4.06.11-D Page
: 1/10

Organization (S): EDF-R & D /AMA, SINETICS

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
D4.06.11 document

Structure of Data SOLVEUR

Summary:

This document describes the structure of data SOLVEUR which defines the method of resolution of the systems linear mono-field (LDLT, MULT_FRONT, MUMPS or GCPC) or multi-fields (FETI).

1 General information

An object of the type SOLVEUR is created (via routines CRESOL or CRESO2 (for DYNA_LINE_TRAN)) on the basis volatile and used in the following total orders:

- *CALC_FORC_AJOU (OP0152)*
- *CALC_MATR_AJOU (OP0199)*
- *DYNA_LINE_TRAN (OP0048)*

· *DYNA_NON_LINE (OP0070 via NMLECT)*
· *DYNA_TRAN_EXPLI (OP0069 via NMLECT)*
· *MECA_STATIQUE (OP0046 via NMLECT)*
· *THER_LINEAIRE (OP0025 via NXLECT)*
· *THER_NON_LINE (OP0186 via NXLECT)*
· *THER_NON_LINE_MO (OP0171)*
· *STAT_NON_LINE (OP0070 via NMLECT)*

This object defines the method of resolution of the linear systems mono-field (LDLT, MULT_FRONT or GCPC) or multi-fields (FETI) and a certain number of parameters necessary to its setting in work.

It is used as driving belt between the various stages of the algorithms which make resolutions of linear system.

2 Relations with the other SD

This SD is internal with the above mentioned total orders and does not have any interaction with the SD of the user.

---

3 Tree structure

SOLVEUR (K19)
:: =record

".SLVK":
OJB
S V K24

".SLVR":
OJB
S V R

“.SLVI”:
OJB
S V I

% if solvor FETI (SLVK (1) = ' FETI'):
“.FETS”: OJB
S V K24
indirect (*)
dim=nbsd

(a number of under-fields)
(*): SOLVEUR not FETI (i.e. FETS (K) .SLVK (1) “FETI” and for the moment imposed on “MULT_FRONT”)
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

4
Contents of the basic objects

4.1 Case
FETI

One speaks here about the contents of the “total” SOLVEUR.

.SLVK*:*
S V K24 dim=9
SLVK (1) *= method of resolution: “FETI”*
SLVK (2) *= prepacking of the matrix of work. The possible values are:*
. “LUMPE”
. “WITHOUT”
SLVK (3) *= Currently used to transmit the order to check the coherence of structure of data SD_FETI with respect to calculation in progress.*
The possible values are: “YES” or “NOT”
SLVK (4) *= method of renumerotation. The possible values are:*
. “MANDELEVIUM” or “MDA” or “MONGREL”
The paramètrage is identical for the main SD and the SD slaves.
SLVK (5) *= symmetrized approximation of the operator of work (carried out in ASMATR) started by key word SOLVEUR/SYME=' OUI'/“NOT”*

The possible values are: "YES" or "NOT"
The paramètrage is identical for the main SD and the SD slaves.
***SLVK (6)** = name of the structure of data of the type SD_FETI*
***SLVK (7)** = method of reorthogonalisation of the directions of descent of the GCPPC of FETI.*
The possible values are: "GS" or "GSM" or "IGSM"
***SLVK (8)** = option of scaling of the preconditionnor of the GCPPC of FETI.*
The only possible value is: "MULT"
***SLVK (9)** = parameter of storage of the matrix of projection **GI**.*
The possible values are:
. "YES" (storage in the shape of a vector)
. "NOT" (one each time rebuilds it starting from objects MATR_ASSE.FETR)
*. "CAL" ("NOT" if size of **GI** > average size of the .VALE of the MATR_ASSE slaves, "YES" if not)*

.SLVR:
S V R dim=4

***.SLVR (1)**: unutilised*
***.SLVR (2)** = resi_rela*
it is the allowed relative precision on the residue
***.SLVR (3)**: unutilised*
***.SLVR (4)** = test_co*
it is the precision of the test of continuity to the interfaces during the rebuilding of local fields

.SLVI:
S V I dim=5
***SLVI (1)** = nprec*
it is the number of figures of the mantissa for the test of relative precision on the pivot.
***SLVI (2)** = nmax_iter*
it is the maximum number of authorized iterations
***SLVI (3)** = istop*
test of singularity during factorization (takes value 1 if STOP_SINGULIER = "NOT", 0 if not).
***SLVI (4)**: unutilised*
***SLVI (5)** = nbreor (if FETI)*
it is the maximum number of directions of descent used for the phase of reorthogonalisation of the GCPPC of FETI

:
D4.06.11-D Page
: 5/10

.FETS:
S V Indirect K24 (*) dim=nbsd (a number of under-fields)

(*): SOLVEUR not FETI (i.e. FETS (K) .SLVK (1) "FETI" and for

the moment imposed on "MULT_FRONT").

Object JEVEUX listing specific SD SOLVEUR to each under-field.
***FETS (I)** = name of SD SOLVEUR of the ième under-field.*

4.2 Case LDLT or MULT_FRONT

***.SLVK**:*
S V K24 dim=5
***SLVK (1)** = method of resolution: "LDLT" or "MULT_FRONT"*
***SLVK (2)**: unutilised*
***SLVK (3)**: unutilised*
***SLVK (4)** = method of renumerotation. The possible values are:*
. "RCMK" or "WITHOUT" (if LDLT)
. "MANDELEVIUM" or "MDA" or "MONGREL" (if MULT_FRONT)
***SLVK (5)** = symmetrized approximation of the operator of work (carried out in routine ASMATR) started by key word SOLVEUR/SYME=' OUI'/"NOT"*
The possible values are: "YES" or "NOT"

.SLVR:
S V R dim=3
***.SLVR (1)**: unutilised*
***.SLVR (2)**: unutilised*
***.SLVR (3)** = size block (if LDLT)*
it is the size of a block in the case of a storage in line of sky [D4.06.07].

.SLVI:
S V I dim=4
***SLVI (1)** = nprec*
it is the number of figures of the mantissa for the test of relative precision on the pivot.
***SLVI (2)**: unutilised*
***SLVI (3)** = istop*
test of singularity during factorization (takes value 1 if SINGULAR STOP = "NOT", 0 if not).
***SLVI (4)**: unutilised*

4.3 Case
GCPC

***.SLVK**:*
S V K24 dim=5

***SLVK (1)** = method of resolution: “GCPC”*
***SLVK (2)** = prepacking of the matrix of work. The possible values are:*
. “LDLT_INC”
. “WITHOUT”
***SLVK (3)**: unutilised*
***SLVK (4)** = method of renumerotation. The possible values are:*
. “RCMK” or “WITHOUT”
***SLVK (5)** = symmetrized approximation of the operator of work (carried out in*
routine ASMATR) started by key word SOLVEUR/SYME=' OUI'/“NOT”
The possible values are: “YES” or “NOT”

.SLVR:
S V R dim=3
***.SLVR (1)**: unutilised*
***.SLVR (2)** = resi_rela*
it is the allowed relative precision on the residue
***.SLVR (3)**: unutilised*

.SLVI:
S V I dim=4
***SLVI (1)**: unutilised*
***SLVI (2)** = nmax_iter*
it is the maximum number of authorized iterations
***SLVI (3)**: unutilised*
***SLVI (4)** = niremp (if SLVK (2) = ' LDLT_INC')*
it is the level of filling of the incomplete prepacking of the GCPC.

4.4 Case
MUMPS

***.SLVK**:*
S V K24 dim=5
***SLVK (1)** = method of resolution: “MUMPS”*
***SLVK (2)**: unutilised*
***SLVK (3)**: algorithm of resolution wished:*

"NONSYM": not-symmetrical matrix (factorization LU)
"SYMGEN": symmetrical matrix "general"
"SYMDEF": symmetrical matrix "definite positive"
***SLVK (4)**: renumerotation: "WITHOUT"*
***SLVK (5)** = symmetrized approximation of the operator of work (carried out in routine ASMATR) started by key word SOLVEUR/SYME=' OUI'/"NOT"*
The possible values are: "YES" or "NOT"

.SLVR:
S V R dim=3
***.SLVR (1)**: unutilised*
***.SLVR (2)** = resi_rela*
It is the relative precision required for the solution (defect: 1.e-6)
If resi_rela <= 0. , one does not control the quality of the solution obtained, but less expensive in time CPU.
***.SLVR (3)**: unutilised*

.SLVI:
S V I dim=4
***SLVI (1)**: unutilised*
***SLVI (2)**: PCPIV percentage of additional memory necessary to the swivellings late (by defect: 20). PCPIV can be > 100.*
***SLVI (3)**: unutilised*
***SLVI (4)**: unutilised*
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

5
Complements in the case of solvor FETI

5.1
Structure of recursive data SOLVEUR

In the case of method FETI, the structure of data SOLVEUR is recursive on two levels. One "Main" SD SOLVEUR, concerning the total field (SLVK (1) = ' FETI'), gathers the objects Usual JEVEUX with in more one object .FETS. This last is a pointer indicating the SD SOLVEUR "slaves" associated with each under-fields.
These local SD SOLVEUR are consisted of same objects JEVEUX as a mono linear solvor usual field (such MULT_FRONT or LDLT) and are concerned with the same parameter setting. On the level of a under-field, one has to be worried only inversions of local matrices and not, for example, name of SD SD_FETI or type of reorthogonalisation installation by the solvor of interface total (of which the parameter setting is contained in "main" SD SOLVEUR).
For the moment, the implementation of FETI in Code_Aster presupposes that these under-fields use all the same linear solvor mono-field (SLVK (1) = "MULT_FRONT" imposed by defect) and with same parameter setting (RENUM and STOP_SINGULIER/NPREC). This homogeneity facilitate handling of the matrices and second local members.

under-field 1

SD SOLVEUR

"main"

(total field)

...
.FETS

under-field I
SD SOLVEUR

"slaves"

...
(under-fields)

Appear 5.1-a: Structure of recursive data SOLVEUR if solvor FETI

5.2
Regulate naming

In the case of a solvor FETI, one chose the following rule of naming for SD SOLVEUR slaves dependent each on a under-field J:
nom_de_la_SD_SOLVEUR_maître (1: 11)/"F" //chaîne_de_caractères_libre (2: 8)
The free character string is generated by a call to routine GCNCON.

5.3
Particular case of parallelism MPI

During an execution in parallel mode MPI, a processor sees itself allotting a certain number of under-fields (cf objects annex "&FETI.LISTE…" structure of data SD_FETI [D4.06.21]). "Main" SD SOLVEUR is always built, but its pointer .FETS will indicate that under-fields concerned with the processor running: .FETS (jk) will be K24 validate that if the jk under-field is in the perimeter of the processor J.

For the processor J

vacuum

SD SOLVEUR
j1 under-field

"main"

(total field)

.FETS

under-field J
SD SOLVEUR "slaves"
2

(under-fields concerned with

...
the processor J)

Appear 5.3-a: Structure of recursive data SOLVEUR if solvor FETI and parallelism MPI
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

5.4 Example

In the case test FETI002A, partitioning in four under-fields leads to SD SOLVEUR following:

Built a main SD SOLVEUR "&&OP0046.SOLVEUR"

====> IMPR_CO OF THE STRUCTURE OF DATA: &&OP0046.SOLVEUR?????
ATTRIBUTE: F CONTENTS: T BASE: >V<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 4
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
*SEGMENT IMPRESSION OF VALUES >**&&OP0046.SOLVEUR .FETS***

1 - >&&OP0046.S.SOF0000000 <>&&OP0046.SOF0000001 <
3 - >&&OP0046.S.SOF0000002 <>&&OP0046.SOF0000003 <

SEGMENT IMPRESSION OF VALUES >&&OP0046.SOLVEUR .SLVI <
1 - 8 100 0 0 100

SEGMENT IMPRESSION OF VALUES >&&OP0046.SOLVEUR .SLVK <
1 - >FETI <>LUMPE <
3 - >OUI <>METIS <
5 - >NON <>SDFETI <
7 - >GSM <>MULT <
9 - >CAL

SEGMENT IMPRESSION OF VALUES >&&OP0046.SOLVEUR .SLVR <
1 - 0.00000D+00 1.00000D-08 8.00000D+02 1.00000D-08

and of SD SOLVEUR slaves "&&OP0046.SOF0000000"

====> IMPR_CO OF THE STRUCTURE OF DATA: &&OP0046.SOF0000000?????
ATTRIBUTE: F CONTENTS: T BASE: >V<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 3
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >&&OP0046.SOF0000000.SLVI
1 - 8 100 0 0 100

SEGMENT IMPRESSION OF VALUES >&&OP0046.SOF0000000.SLVK <
1 - >MULT_FRO <>LUMPE <
3 - >OUI <>METIS <
5 - >NON <
7 - >GSM <>MULT <
9 - >CAL

SEGMENT IMPRESSION OF VALUES >&&OP0046.SOF0000000.SLVR <
1 - 0.00000D+00 1.00000D-08 8.00000D+02 1.00000D-08
====> FINE IMPR_CO OF STRUCTURE OF DATA: &&OP0046.SOF0000000?????
...
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data SOLVEUR

Date:
29/09/05
Author (S):
***J.** Key **PELLET**, O.BOITEAU*
:
D4.06.11-D Page

: 10/10

Intentionally white left page.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
4
Titrate:
Structure of Data INFO_CHARGE
Date:
28/01/1999
Author (S):
J. PELLET
Key:
D4.06.12
Page: 1/4
Mechanical department and Digital Models
Index:
With
Diffusion:
Developers

Data-processing handbook of Description
D4.06 booklet:
D4.06.12 document

Structure of Data INFO_CHARGE

1
General

An object of the type INFO_CHARGE is created on the volatile basis and is used in the total orders.
It gives information on the loadings used in the total order.

2
Relations with the other SD

This SD is internal with the above mentioned total orders and does not have any interaction with the SD of *the user.*

---

3 Tree structure

INFO_CHARGE
(K19)
:: =record
".INFC"
:
OJB S
V
I
".LCHA"
:
OJB S
V
K24

4 Contents of the basic objects

4.1 .INFC

.INFC:
S V I
That is to say nchar the number of loads used in the total order.

4.1.1 In mechanics

- *dimension = 4 X nchar + 3 in mechanics*
- *dimension = 2 X nchar + 1 in thermics*

.INFC (1) = nchar
The values infc (2) with infc (1+nchar) are reserved for the loads of the Dirichlet type.
For 1 ² ichar ² nchar
INFC (1+ichar)
= 0
if not load
= 1
if the load results from AFFE_CHAR_MECA

= 2
if the load results from AFFE_CHAR_MECA_F and if it is
independent of time
= 3
if the load results from AFFE_CHAR_MECA_F and if it is
dependent on time
= 4
following force
= 1
if the load results from AFFE_CHAR_CINE
= 2
if the load results from AFFE_CHAR_CINE_F and if it is
independent of time
= 3
if the load results from AFFE_CHAR_CINE_F and if it is
dependent on time
*The values infc (2+nchar) with infc (1+2*nchar) are reserved for the mechanical loads of*
Neuman type.
For 1 ² ichar ² nchar
INFC (1+nchar+ichar)
= 0
if not load
= 1
if the load results from AFFE_CHAR_MECA
= 2
if the load results from AFFE_CHAR_MECA_F and if it is
independent of time
= 3
if the load results from AFFE_CHAR_MECA_F and if it is
dependent on time
*.INFC (1+2*nchar+1) = number of the load-bearing capacity the temperature in the case of a load*
thermics.
*.INFC (1+2*nchar+2) = a number of loads giving of the forces of Laplace.*
*The values infc (3+2*nchar+1) with infc (3+3*nchar) are reserved for the loads of the type*
contact.

For 1 ² ichar ² nchar
*INFC (3+2*nchar+ichar)*
= 1
if the load is of contact type

= 0
if not
*The values infc (3+3*nchar+1) with infc (3+4*nchar) are reserved for the differential heads.*
For 1 ² ichar ² nchar
*INFC (3+3*nchar+ichar)*
= 1
if the load is differential
= 0
if not

4.1.1
In thermics

.INFC (1) = nchar
The values infc (2) with infc (1+nchar) are reserved for the loads of the Dirichlet type.
For 1 ² ichar ² nchar
INFC (1+ichar)
= 0
if not load
= 1
if the load results from AFFE_CHAR_THER
= 2
if the load results from AFFE_CHAR_THER_F and if it is
independent of time
= 3
if the load results from AFFE_CHAR_THER_F and if it is
dependent on time
= 4
following force
= 1
if the load results from AFFE_CHAR_CINE
= 2
if the load results from AFFE_CHAR_CINE_F and if it is
independent of time
= 3
if the load results from AFFE_CHAR_CINE_F and if it is
dependent on time
*The values infc (1+nchar+1) with infc (1+2*nchar) are reserved for the loads of the type*
Neuman.
For 1 ² ichar ² nchar
INFC (1+nchar+ichar)
= 0
if not load
= 1
if the load results from AFFE_CHAR_THER
= 2
if the load results from AFFE_CHAR_THER_F and if it is
independent of time
= 3
if the load results from AFFE_CHAR_THER_F and if it is
dependent on time

4.2
.LCHA

.LCHA:
S V K24
dimension = nchar
LCHA contains the name of all the loads implied in the total order.

Intentionally white left page.

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D4.06 booklet: -
D4.06.13 document

Structure of Data LISTE_RELA

1 General information

The structure of data LISTE_RELA is a “volatile” structure of data which is useful in operators of structure intermediate enters the data of the user (them linear relations with dualiser) and their translation in finite elements of Lagrange (in the LIGREL of CHARGE) and in CHARTS containing the coefficients of the relations.

Approximately, the idea is as follows: one progressively stores the linear relations (one by one) in a LISTE_RELA (routine AFRELA) then one “pours” the LISTE_RELA in the LOAD in end of order (routine AFLRCH).

---

2 Tree structure

LISTE_RELA (K19):: =record

“.RLCO”
: OJB
/S V R

/S V C

“.RLBE”
: OJB
/S V R

/S V C

/S V K24

“.RLDD”
: OJB
/S V K8

“.RLNO”
: OJB
/S V K8

“.RLNT”
: OJB
/S V I

“.RLPO”
: OJB
/S V I

“.RLSU”
: OJB
/S V I

“.RLTC”
: OJB
/S.E.K8

“.RLTV”
: OJB
/S.E.K8

“.RLNR”
: OJB
/S.E.I

3
Contents of the OJB

3.1 Notations

nb_rela
a number of relations stored in the LISTE_RELA

nb_coef_1rel
coeficients of a linear relation numbers (“left” part of the equation)
The coefficients can be:
· real
· or
complexes

nb_coef_tot
numbers total coefficients of the whole of the linear relations stored in LISTE_RELA.
nb_coef_tot = nb_coef_1rel (1) +nb_coef_1rel (2) +…

coef_impo
specified value for the linear relation (“right” part of the equation or “second member ").
The specified value can be:
· real
· complex
· or
function

3.2 Object
“.RLNR”

This object is length 1.
RLNR (1): a number of linear relations nb_rela.

3.3 Object
“.RLTC”

This object is length 1.
RLTC (1): type of the coefficients of the relations “REALITY” or “COMP”.

3.4 Object
".RLTV"

This object is length 1.
RLTV (1): type of the second members of the relations "REALITY", "COMP" or "FONC".
Handbook of Data-processing Description
D4.06 booklet: -
HT-66/05/003/A

3.5 Object “.RLNT”

This object is length nb_rela.
It contains the numbers of coefficients for each relation.

RLNT (irela): coefficents of the relation irela numbers
= nb_coef_1rel (irela).

3.6 Object “.RLPO”

This object is length nb_rela.
It pemet “to point” (for a given relation) in objects” .RLCO ", “.RLNO” and” .RLDD ". These 3 objects have same dimension and one points there in the same way.

RLPO (irela): address in .RLCO (for example) last coefficient of the relation irela.

RLCO (RLPO (irela) - RLNT (irela) + 1) is the 1st term of the relation irela.

3.7 Object “.RLCO”

This object is length nb_coef_tot.
It contains the coefficients (R or C) of the equations.

3.8 Object “.RLNO”

This object is length nb_coef_tot.
It contains the names of the nodes implied in the relations.

3.9 Object
".RLDD"

This object is length nb_coef_tot.
It contains the names of the CMPS implied in the relations.

3.10 Object
".RLBE"

This object is length nb_rela.
It contains the second members of the relations.

RLBE (irela): second member of the relation irela = coef_impo (irela).

3.11 Object
".RLSU"

This object is length nb_rela.
It contains an indicator (0 or 1) for saying if each linear relation must be taken into account (or not).

Indeed, before "pouring" the linear relations in the LOAD, one examines whether certain relations are not given in several specimens. If it is the case, one eliminates all the doubled blooms while putting their indicator with 1.

RLSU (irela):
0: to take into account

1: to be unaware of.

Intentionally white left page.

Titrate:
Structures of data related to contact-friction

Date:
06/12/04
Author (S):
***Mr. ABBAS**, NR. TARDIEU Key*
:
D4.06.14-D Page

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
Document: D4.06.14

Structures of data related to contact-friction

Summary:

This document describes the structures of data necessary to the definition (SD "DEFI_CONT") and for the resolution
(SD "RESO_CONT") of the problems of contact-friction defined by the key word CONTACT of the operator
AFFE_CHAR_MECA. One endeavours to give the detailed instructions of the majority of the tables used in
corresponding routines. Description is purely data-processing, and it is advised to read them before
reference materials [R5.03.50], [R5.03.51], of use [U4.25.01], implementation practical
[U2.04.04] and of maintenance of the contact [D9.05.02].

Version
7.4

Titrate:
Structures of data related to contact-friction

Date:
06/12/04
Author (S):
Mr. ABBAS, NR. TARDIEU Key
:
D4.06.14-D Page
: 2/16

1 Philosophy general

Potential zones of contact are defined in operator AFFE_CHAR_MECA for each occurrence of the key word "CONTACT". Each zone is defined by an occurrence of the key word. One zone of contact comprises several surfaces (two in general) which one seeks to prevent the interpenetration two to two. In the case of methods "NODAL", "MAIT_ESCL" and "MAIT_ESCL_SYME", there are two surfaces whose composition is given under the key words GROUP_MA_MAIT (or MAILLE_MAIT) and GROUP_MA_ESCL (or MAILLE_ESCL). For the methods "TERRITORY" and "HIERARCHICAL" (not implemented to date), one will use key word GROUP_MA (or NETS): in this case, each group of meshs (each mesh) of the list will define a potential surface of contact (there could thus be more than two surfaces per zone). data relating to the various zones and surfaces of contact are stored in a structure of given of type "DEFI_CONT" whose name is TANK (1: 8)/".CONTACT".

In operators STAT_NON_LINE and DYNA_NON_LINE, one supposes that only one load contains contact (one checks it in the routine nmdome). During step of time and iterations of Newton, one fills of the tables of size fixes (dimensioned using the maximum of nodes slaves) who contain the data necessary to the treatment of the contact (structure of data of the type "RESO_CONT"). They are sometimes under-tables resulting from the tables created in AFFE_CHAR_MECA: they relate to only the nodes or meshs in the course of treatment with the algorithm (zones of contact effective current). In these tables, information is followed sequentially without concept of zone or of surface of contact: very coarsely, one stores the couples node slave - mesh (or node) main and characteristics associated (ddls concerned, coefficients, components of *normal, play running).*

Note:

· The system of contact is composed of several zones, themselves consistent in surfaces, made up of meshs, container of the nodes,
· Surfaces of contact are located by their absolute number I in the list of all surfaces of contact, all zones confused,
· Seuls tables CONTMA, CONTNO and SANSNO index the nodes and the meshs by their absolute number in the code; all the other tables use the index in CONTMA and CONTNO to indicate a mesh or a node,
· One always takes three components for the normal, in 2D as in 3D. On the other hand, the table of the degrees of freedom contains of them two per node in 2D, and three in 3D.

---

2 Structure of data DEFI_CONT

The structure of data DEFI_CONT contains the tables defining the potential zones of contact (created in AFFE_CHAR_MECA, except DDLCO and PDDLCO created in STAT_NON_LINE).

2.1 List variables

It is supposed that only one load contains the key word factor "CONTACT".
All the tables start with TANK (1: 8)/".CONTACT" (variable DEFICO (1: 16)), with the suffix:

Variable suffix
Type
Dimension Subscripted
by
Contents
.NDIMCO NDIMCO I 9+NZOCO
list useful entireties, and numbers effective
nodes slaves for each zone
(the max at the beginning)
*.METHCO METHCO I 1+8*NZOCO*
number of
a number of zones and characteristic of
zone of contact
method of pairing for each one
*.TOLECO TOLECO R 2*NZOCO number of*
parameters of geometrical tolerance
zone of contact
for pairing
*.CONVCO CONVCO I 3*NZOCO number of*
parameters of convergence
zone of contact
.SYMECO SYMECO I NZOCO+1
information on the symmetrical zones
of pairing (MAIT_ESCL_SYME)
.PZONECO PZONE
I
NZOCO+1 number of
number of the last surface of
zone of contact
each zone
.MAILCO CONTMA I
Pointer NMACO PSURMA lists numbers of the meshs of
contact of various surfaces
potential
.PSUMACO PSURMA
I
1+NSUCO number of
index of the last mesh of each
surface
surface in vector CONTMA

.NOEUCO CONTNO I
Pointer NNOCO PSURNO lists numbers of the nodes of
contact of various surfaces
potential
.PSUNOCO PSURNO
I
1+NSUCO number of
index of the last node of each
surface
surface in vector CONTNO
*.NOEUQU CONOQU I 3*NNOCQ/2*
pointer PNOQUA lists numbers of the nodes of
contact "quadratic" of different
potential surfaces
.PNOEUQU PNOQUA
I
1+NSUCO number of
index of the last node of each
surface
surface in vector CONOQU
.MANOCO MANOCO I
Pointer NMANO PMANO indices of the meshs of CONTMA
containing a given node of CONTNO
.PMANOCO PMANO
I
1+NNOCO index of node
index in MANOCO of the last
in CONTNO
net containing a given node
.NOMACO NOMACO I
Pointer NNOMA PNOMA indices of the nodes of CONTNO
belonging to a mesh given of
CONTMA
.PNOMACO PNOMA
I
1+NMACO index of mesh
index in NOMACO of the last node
in CONTMA
belonging to a given mesh
.MAMACO MAMACO I
Pointer NMAMA PMAMA indices of the meshs of CONTMA
adjacent with a given mesh
.PMAMACO PMAMA

I
1+NMACO index of mesh
index in MAMACO of the last
in CONTMA
adjacent mesh with a given mesh
.SSNOCO SANSNO I
Pointer NNOCO PSANS
absolute numbers of the nodes to be excluded
nodes slaves
.PSSNOCO PSANS
I
1+NZOCO number of
index of the last node to be excluded in
zone of contact
SANSNO

.JSUPCO JEUSUP R8
NZOCO number of
value of the fictitious play
zone of contact
.NOZOCO NOZOCO I
NNOCO index of node
number of the zone of contact to which
in CONTNO
the node belongs

.CHAMCO CHAMCO I
NZOCO number of
code field to which apply
zone of contact
unilateral conditions
.COEFCO COEFCO R8
NZOCO number of
coefficient of the unilateral relation for
zone of contact
pressure or the temperature
.DDLCO DDLCO I
Pointer NDDL PDDL numbers of the degrees of freedom
potentially implied in the writing
unilateral relations
.PDDLCO PDDL
I
NNOCO index of node
index in DDLCO of the last ddl of one
in CONTNO
node of CONTNO given
.JEUFO1 JJFO1 K8
NZOCO Number of
Fictitious play when it is given by one
surface
function of space in
contact
AFFE_CHAR_MECA_F
.JEUFO2 JJFO2 K8
NZOCO Number of
Fictitious play when it is given by one
surface
function of space in
contact
AFFE_CHAR_MECA_F
DIRCO JDIR R8
3*NZOCO
Number of
Direction fixes nodal pairing
surface
data by VECT_Y
contact
RUB IFRO R8 NESMAX
Number of

Coefficient of friction of Coulomb
node slave
PENAL IPENA
R8
*2*NESMAX*
Number of
Coefficient of regularization of the contact
node slave
and of friction
COMAFO ICOMA R8 NESMAX
Number of
Value of COEF_MATR_FROT
node slave
TANDEF JTGDEF
*R8 3*NZOCO*
Number of
Value of VECT_Y
surface
contact
NORLIS JNORLI I NZOCO+1
Number of
Indicate the presence of smoothing of
surface
normals
contact

This part gathers the objects suitable for the CONTINUOUS method:

.CARACF JCMCF
*R 6*NZOCO+1*
number of
integration and coefficients of
zone of contact
regularization
.ECPDON JECPD
*I 5*NZOCO+1*
number of
parameters of the loops of
zone of contact
CONTINUOUS method
*.MAESCL JMAESC I 3*NTMA+1*
number of
for each mesh one gives the number

net slave
of its zone numbers points of
contact
*.NOESCL JNOESC R 10*NNOCO+*
number of node vectors tangent and normal of
of contact
each point.
.TABFIN JTABF
*R 16*NTPC+1*
number of point characteristics of pairing.
of contact

2.2 Table
NDIMCO (addresses JDIM)

ZI (JDIM) =
NDIM
dimension of space (two or three)
ZI (JDIM+1) =
NZOCO
a number of zones of contact
ZI (JDIM+2) =
NSUCO

a number of surfaces of contact
ZI (JDIM+3) =
NMACO
a number of meshs of contact
ZI (JDIM+4) =
NNOCO
a number of nodes of contact
ZI (JDIM+5) =
NMANO
dimension of MANOCO
ZI (JDIM+6) =
NNOMA
dimension of NOMACO
ZI (JDIM+7) =
NMAMA
dimension of MAMACO
ZI (JDIM+8)
= NESMAX
a maximum number of nodes slaves
ZI (JDIM+8+IOC) = an effective number of nodes slaves in zone IOC (a number maximum at the time of initialization), IOC=1, NZOCO

2.3 Table
METHCO (addresses JMETH)

ZI (JMETH) = NZOCO: a number of zones of contact

For zone N:

ZI (JMETH+8* (n-1) +1) =
-1 if APPARIEMENT= "NOT"
0 if "NODAL" APPARIEMENT=

1 if APPARIEMENT= "MAIT_ESCL" or "MAIT_ESCL_SYME"

2 if APPARIEMENT= "TERRITORY"

3 if "HIERARCHICAL" APPARIEMENT=

4 if VECT_NORM_2 is defined
ZI (JMETH+8* (n-1) +2) =
1 VECT_Y is indicated and 0 if not
ZI (JMETH+8* (n-1) +3) =
not used

ZI (JMETH+8 (n-1) +4) =*
1 if linear projection (rectilinear segment or plane triangle)

2 if quadratic projection
ZI (JMETH+8 (n-1) +5) =*
+1 if RECHERCHE= "NOEUD_BOUCLE"

+-2 if RECHERCHE= "NOEUD_VOISIN"/"MAILLE_VOISIN"
+-3 if RECHERCHE= "NOEUD_BOITE"/"MAILLE_BOITE"
ZI (JMETH+8 (n-1) +6) =*
-1 if the method of CONTACT used east "PENALIZED"

0 if the method of CONTACT used is "CONTRAIN"

1 if the method of CONTACT used is "LAGRANGI"

2 if the method of FRICTION 2D used is "LAGRANGI"

3 if the method of FRICTION 2D or 3D used east "PENALIZED"

4 if the method of FRICTION 3D used is "LAGRANGI"

5 if the method of CONTACT and FRICTION used is "PENALIZED"

6 if the method used is "CONTINUOUS"
ZI (JMETH+8 (n-1) +7) =*
0 if REAC_GEOM= "WITHOUT"

-1 if "AUTOMATIC" REAC_GEOM=

if not NB_REAC_GEOM (geometrical frequency of reactualization)
ZI (JMETH+8 (n-1) +8) =*
0 if NORMALE= "MAIT"

1 if NORMALE= "MAIT_ESC"
ZI (JMETH+9 (n-1) +9) =*
0 if STOP_SINGULIER= "YES"
1 if STOP_SINGULIER= "NOT"

2.4 Table
CONVCO (addresses JCONV)

For zone N:

ZI (JCONV+3 (n-1)) =*
0 if STOP_SINGULIER= "YES"
1 if STOP_SINGULIER= "NOT"
ZI (JCONV+3 (n-1) +1) =*
NB_RESOL
ZI (JCONV+3 (n-1) +2) =*
ITER_MULT_MAXI

2.5 Table
TOLECO (addresses JTOLE)

For zone N:

ZI (JTOLE+2 (n-1)) =*
TOLE_PROJ_EXT
ZI (JTOLE+2 (n-1) +1) =*
TOLE_PROJ_INT

2.6 Table
SYMECO (addresses JSYME)

ZI (JSYME): A number of symmetrical zones of contact

For symmetrical zone N:

ZI (JSYME+n) =
Number of the zone principal partner at symmetrical zone N

2.7 Table
CARACF (addresses JCMCF)

ZI (JCMCF) = NZOCO: numbers total zones of contact.
In this table some parameters for the methods are stored CONTINUES, LAGRANGIAN and PENALIZATION. For the method CONTINUES, one specifies, inter alia, the diagram of integration with
to use for the terms of contact and friction and the coefficients of increase. Let us recall that integration with the nodes is taken by defect INTEGRATION=' NOEUD' and that diagrams SIMPSON,
SIMPSON1 and SIMPSON2 are available only in 2D.

For zone N:

CARACF (1+6 (n-1) +1) =*
1 if INTEGRATION=' NOEUD'

2 if INTEGRATION=' GAUSS'

3 if INTEGRATION=' SIMPSON'

4 if INTEGRATION=' SIMPSON1'

5 if INTEGRATION=' SIMPSON2'
CARACF (1+6 (n-1) +2) =*
Coefficient of increase for the contact
COEF_REGU_CONT
CARACF (1+6 (n-1) +3) =*
Coefficient of increase for friction
COEF_REGU_FROT
CARACF (1+6 (n-1) +4) =*
Coefficient of Coulomb for friction.
CARACF (1+6 (n-1) +5) =*
1 if FROTTEMENT=' SANS'

3 if FROTTEMENT=' COULOMB'
CARACF (1+6 (n-1) +6) =*
the value of COEF_MATR_FROT

2.8

System of contact: pointer PZONE (addresses JZONE)

absolute number (I) of the first surface of zone N:
ZI (JZONE+n-1) +1
absolute number (I) of the last surface of zone N:
ZI (JZONE+n)
a number of surfaces of zone N:
ZI (JZONE+n) - ZI (JZONE+n-1)

2.9 Zone
N: pointers PSURMA, PSURNO and PNOQUA (addresses JSUMA,
JSUNO and JNOQUA)

number of the first
: i1 = ZI (JZONE+n-1) +1
surface
number of the last surface: i2 = ZI (JZONE+n)
a number of meshs of the zone: ZI (JSUMA+i2) - ZI (JSUMA+i1-1)

=
ZI (JSUMA+ZI (JZONE+n)) - ZI (JSUMA+ZI (JZONE+n-1))
a number of nodes of the zone: ZI (JSUNO+i2) - ZI (JSUNO+i1-1)

=
ZI (JSUNO+ZI (JZONE+n)) - ZI (JSUNO+ZI (JZONE+n-1))

a number of nodes of the zone: ZI (JNOQUA+i2) - ZI (JNOQUA +i1-1)
being node mediums of one
quadratic element

=
ZI (JNOQUA +ZI (JZONE+n)) - ZI (JNOQUA +ZI (JZONE+n-1))

2.10 Surface I: tables CONTMA, CONTNO and CONOQU (addresses JMACO, JNOCO and JNOQU), pointer PSURMA, PSURNO and PNOQUA (addresses JSUMA, JSUNO and JNOQUA)

The number of meshs of surface I east: nbma = ZI (JSUMA+i) - ZI (JSUMA+i-1)
The index in CONTMA of the first mesh of surface I east: ZI (JSUMA+i-1) +1
The index in CONTMA of the last mesh of surface I east: ZI (JSUMA+i)
The list of the numbers of the meshs of surface I is ZI (JMACO+jdecma+ima-1) for ima=1, nbma, with jdecma = ZI (JSUMA+i-1).
The imaième mesh of surface I has as an absolute number: ZI (JMACO+jdecma+ima-1); its index in CONTMA is: posma = jdecma+ima.
Net index posma in CONTMA: its absolute number is ZI (JMACO+posma-1).

The number of nodes of surface I east: nbno = ZI (JSUNO+i) - ZI (JSUNO+i-1)
The index in CONTNO of the first node of surface I east: ZI (JSUNO+i-1) +1
The index in CONTNO of the last node of surface I east: ZI (JSUNO+i)
The list of the numbers of the nodes of surface I is ZI (JNOCO+jdecno+ino-1) for ino=1, nbno, with jdecno = ZI (JSUNO+i-1).
The inoième node of surface I has as an absolute number: ZI (JNOCO+jdecno+ino-1); its index in CONTNO is: posno = jdecno+ino.
Node of index posno in CONTNO: its absolute number is ZI (JNOCO+posno-1).

The number of nodes of surface I being node medium of a quadratic mesh is: nnoq = ZI (JNOQUA+i) - ZI (JNOQUA +i-1)
The index in CONOQU of the first "quadratic" node medium of surface I east: 3*ZI (JNOQUA +i-1) +1
The index in CONOQU of the last node "quadratic" medium of surface I east: 3*ZI (JNOQUA +i) - 2
The list of the numbers of the nodes medium "
quadratic
" of surface I is ZI (JNOQU
+jdecqu+3* (inq-1)) for inq=1, nnoq, with jdecqu = 3*ZI (JNOQUA +i-1) +1.
The list of the numbers of the associated nodes top for surface I is ZI (JNOQU +jdecqu+3* (inq-1) +1) and ZI (JNOQU +jdecqu+3* (inq-1) +2) for inq=1, nnoq, with jdecqu = 3*ZI (JNOQUA +i-1) +1.
The inqième "quadratic" node medium of surface I has as an absolute number: ZI (JNOQU

+jdecqu+3 (inq-1) - 1); its index in CONOQU is: posqu = jdecqu+3* (inq-1).*
"Quadratic" node of index posqu in CONOQU: its absolute number is ZI (JNOQU+posqu-1).

2.11 Tables MANOCO, NOMACO and MAMACO (addresses JMANO, JNOMA and JMAMA) pointer PMANO, PNOMA and PMAMA (addresses JPOMA, JPONO and JPOIN)

For the node of index posno in CONTNO, the index in CONTMA of the meshs of contact of same surface containing the node is ZI (JMANO+jdec+ima-1) with jdec = ZI (JPOMA+posno-1) and ima varying of 1 with nbma, for nbma = ZI (JPOMA+posno) - ZI (JPOMA+posno-1), or even ZI (JMANO+k-1) for K varying of ZI (JPOMA+posno-1) +1 with ZI (JPOMA+posno).

For the mesh of index posma in CONTMA, the index in CONTNO of the nodes of contact of this mesh is ZI (JNOMA+jdec+ino-1) with jdec = ZI (JPONO+posma-1) and ino varying of 1 with nbno, for nbno = ZI (JPONO+posma) - ZI (JPONO+posma-1), or even ZI (JNOMA+k-1) for K varying of ZI (JPONO+posma-1) +1 with ZI (JPONO+posma).

The list of the indices in CONTMA of the meshs close to the mesh of index posma is: ZI (JMAMA+jdec+ima-1), with jdec = ZI (JPOIN+posma-1) and ima varying of 1 with nbma, for nbma = ZI (JPOIN+posma) - ZI (JPOIN+posma-1).

2.12 Table
SANSNO and pointer PSANS (addresses JSANS and JPSANS)

For zone N:

a number of nodes to be excluded from the nodes slaves: nsans = ZI (JPSANS+n) - ZI (JPSANS+n-1)
absolute numbers of the nodes to be excluded: ZI (JSANS+jdec+ino-1), for ino = 1, nsans, with
jdec = ZI (JPSANS+n-1).

2.13 Table
JEUSUP (addresses JJSUP)

For zone N: ZR (JJSUP+n-1) = value for the zone of the fictitious play (DIST_1+DIST_2, or COEF_IMPO) given by the user.

2.14 Tables NOZOCO (addresses JZOCO), CHAMCO (addresses JCHAM) and
COEFCO (addresses JCOEF)

For the node of index posno in CONTNO, the number of the zone of contact to which this one belongs is:
N = ZI (JZOCO+posno-1).

For zone N:

Code field to which applies the unilateral relation: icode = ZI (JCHAM+n-1)
icode = +1: relation on displacements (with pairing: “deformable” contact)
icode = -1: relation on displacements (without pairing: “rigid” contact)
icode = -2: relation on the pressure (without pairing: only for one modeling “THM”)
icode = -3: relation on the temperature (without pairing”: only for one modeling “THM”)
icode = -4: relation on pressure 1 (without pairing”: only for one modeling “THM”)
icode = -5: relation on pressure 2 (without pairing”: only for one modeling “THM”)

Multiplying coefficient of the unilateral relation: ZR (JCOEF+n-1)
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/04/003/A

2.15 Table DDLCO and pointer PDDLCO (addresses JDDL and JPDDL)

These two tables are in the structure of data DEFI_CONT but are defined in the routine crsdco called by STAT_NON_LINE.

For the node of index posno in CONTNO, the degrees of freedom are ZI (JDDL+jdecdl+iddl-1) for iddl varying of 1 with nbddl with nbddl = ZI (JPDDL+posno) - ZI (JPDDL+posno-1) and jdecdl = ZI (JPDDL+posno-1).

One gathered in the continuation the structures of data suitable for the CONTINUOUS method.

2.16 Table ECPDON (addresses JECPD)

This table gives some parameters total necessary for the algorithmic one used by CONTINUOUS method. Let us note that parameters ITER_CONT_MAX, ITER_FROT_MAX and ITER_GEOM_MAX fix the maximum number of the loops of contact, threshold of friction and of geometry. These numbers can not be reached if the test of stop of each ball is satisfied. It test, to see routine mmmcri.f, relates to the value of the relative increment of displacement:

U
U
with 10
= -2.

NR being the number of the zone of contact.

ECPDON (1+5 (N-1) +1) =*
1 if MODL_AXIS=' OUI'

0 if MODL_AXIS=' NON'
ECPDON (1+5 (N-1) +2) =*
The value of ITER_CONT_MAX
ECPDON (1+5 (N-1) +3) =*
The value of ITER_FROT_MAX
ECPDON (1+5 (N-1) +4) =*
The value of ITER_GEOM_MAX
ECPDON (1+5 (N-1) +5) =*
The initial threshold value SEUIL_INIT

MAESCL (1): numbers total zones of contact.

2.17 Table
MAESCL (addresses JMAESC)

NR being the number of the mesh slave.

MAESCL (1+3 (N-1) +1) =*
index of the mesh NR in table CONTMA
MAESCL (1+3 (N-1) +2) =*
number of the zone of contact of NR
MAESCL (1+3 (N-1) +3) =*
numbers points of contact in NR

MAESCL (1): numbers total meshs slave.

2.18 Table
NOESCL (addresses JNOESC)

NR is the number of node of contact. The whole parameter I varies from 1 to 3.

NOESCL (1+3 (N-1) +1) =*
0 if NR is slave 1 if not.
NOESCL (1+3 (N-1) +1+I) =*
components of the first tangent vector
NOESCL (1+3 (N-1) +4+I) =*
components of the first tangent vector
NOESCL (1+3 (N-1) +7+I) =*
components of the first tangent vector
Data-processing handbook of Description

NOESCL (1): numbers total nodes of contact. Let us recall that in 2D the 3rd component of vectors tangent or normal is taken equalizes to zero.

This table is used for the smoothing (routine "lissag") which makes it possible to smooth the normals on the surfaces of contact intervening in the calculation of the matrix of contact. Let us recall that smoothing is made in two stage. The first consists in carrying out an average of the normals to the meshs which contain the node of contact. The second consists in interpolating, with the functions of form associated with the element, a field of normals in any point of the element.

2.19 Table
TABFIN (addresses JTABF)

In this table is classified all information necessary for the resolution concretes problem of the rubbing contact. This information is described in the routine mappar. Let us recall that pairing is made in an exact way using a method of Newton for the resolution of one problem of optimization with constraints (cf routine mprojp) and which allows us, at the same time to recover the values of the tangent vectors.

NR being the number of the point of contact. TABFIN (1): numbers total points of contact. whole parameter I varies from 1 to 3.

TABFIN (1+16 (N-1) +1) =*
absolute number of the mesh Master

TABFIN (1+16 (N-1) +2) =*
absolute number of the mesh slave
TABFIN (1+16 (N-1) +3) =*
first barycentric parameter of the point NR
TABFIN (1+16 (N-1) +4) =*
first barycentric parameter of the point in screw-with live
TABFIN (1+16 (N-1) +5) =*
second barycentric parameter of the point in opposite
TABFIN (1+16 (N-1) +5+I) =*
3 components of the 1st tangent vector
TABFIN (1+16 (N-1) +8+I) =*
3 components of the 2nd tangent vector
TABFIN (1+16 (N-1) +12) =*
second barycentric parameter of the point NR
TABFIN (1+16 (N-1) +13) =*
statute of contact
TABFIN (1+16 (N-1) +14) =*
initial value of contact pressure
TABFIN (1+16 (N-1) +15) =*
number of the zone of contact of the point NR
TABFIN (1+16 (N-1) +16) =*
weight of the point of contact

3
Structure of data RESO_CONT

The structure of data RESO_CONT contains the tables defining the effective couples of contact (created in STAT_NON_LINE or DYNA_NON_LINE) and the variables used in the method of resolution by FORCED methods, LAGRANGIAN and PENALIZATION.

3.1
List variables

All the tables start with RESOCO (1: 14) ("&&RESOCO"), followed suffix:

Variable suffix Standard Dimension
Subscripted
by
Contents
.APREAC APREAC
I
*4*NZOCO number of*
indicator of reactualization for
zone of contact pairing, fixed meter of pairing,
type of projection and reactualization of
normals, number of surface slave
.APPARI APPARI
*I 1+3*NESMAX*
number of
a number of nodes slaves, and for each one

node slave
: index of the node slave, index of the mesh
Master paired, indicating of reactualization
.APMEMO APMEMO
I
*4*NNOCO index of the node*
data relating to pairing of the last
in CONTNO
blow where this node was slave
.APPOIN APPOIN
I
1+NESMAX number of
pointer of navigation in APCOEF,
node slave
APCOFR and APDDL

.APCOEF APCOEF
*R8 30*NESMAX pointer APPOIN multiplying coefficients of the ddls (1 by ddl) for imposition of nonthe penetration*
(+1 for the node slave, and the opposite of
value of the function of form for each
main node)
.APCOFR APCOFR
*R8 60*NESMAX pointer APPOIN multiplying coefficients of the ddls (1 by ddl) for imposition friction (+1 for*
node slave, and opposite of the value of
function of form for each main node)
.APDDL APDDL
*I 30*NESMAX*
pointer APPOIN numbers of the degrees of freedom of the node
slave and of the nodes of the mesh Master
paired
.NORINI NORINI
R8
*3*NNOCO index of the node*
direction of evaluation of the normal play on
in CONTNO
the whole of the potential nodes of contact
.NORMCO NORMCO
R8
*3*NESMAX number of*
direction of evaluation of the normal play on
node slave
connection of contact
.TANGCO TANGCO
R8
*6*NESMAX number of*
direction of evaluation of the tangent play on
node slave
connection of contact
.APJEU APJEU
R8
NESMAX number of
value of the normal play running between the node
node slave
slave and the mesh Master paired
.APJEFX APJEFX
R8
NESMAX

number of
value of the tangent play in direction 1
node slave
between the node slave and the mesh Master
paired
.APJEFY APJEFY
R8
NESMAX
number of
value of the tangent play in direction 2
node slave
between the node slave and the mesh Master
paired
.JEUINI JEUINI
R8
NESMAX
number of
value of the initial play when
node slave
REAC_GEOM=' SANS'
.COCO COCO
I
8

to remember the state of preceding contact
.LIAC LIAC
I
*3*NESMAX+1*
number of
absolute numbers of the active connections of
node slave
contact-friction
.CONVEC CONVEC
*K8 3*NESMAX+1*
number of
Type of active connections
: contact or
node slave
adherent friction
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/04/003/A

.LIOT LIOT
I
*4*NESMAX+4*
number of
removed connections of contact-friction
node slave
DRIVEN .MU
R8
*6*NESMAX*
number of
multipliers of Lagrange related to
node slave
contact-friction
.COEFMU COEFMU
R8
NESMAX
number of
coefficient by which it is necessary to multiply it
node slave
multiplier of Lagrange of the front contact
to test its sign
.ATMU ATMU
R8
NEQ number of ddl
nodal forces of contact
.AFMU AFMU
R8

NEQ number of ddl
nodal forces of friction
.DEL0 DELT0 R8
NEQ number of ddl
vector used in the algorithm of resolution
.DELT R8 DELTA
NEQ number of ddl
vector used in the algorithm of resolution
.CM1A CM1A

second members used in the algorithm
of resolution of the contact
.CM2A CM2A

second members used in the algorithm
of resolution of friction
.CM3A CM3A

second members used in the algorithm
of resolution of friction
.MATR MATR

SD of the type MATR_ASSE [D4.06.10]: stamp
used in the algorithm of resolution
.SLCS STOC

SD of the type STOC_LCIEL [D4.06.07]:
description of the storage of matrix MATR

For the bond of the variables described with the resolution of the problem of contact by the method of the constraints
active, one will refer to the document [R5.03.50] and for the resolution of the problem of contact-friction to
document [R5.03.51].

3.2 Table

APREAC (addresses JREAC)

For zone N:
ZI (JREAC+4 (n-1)) : reactualization of pairing*

0 :
not
-1 :
no pairing but initial passage in rechno
1 :
by double loop on the nodes ("NOEUD_BOUCLE")
+/2: by vicinity of "last" ("NOEUD_VOISIN"/"MAILLE_VOISIN")
+/3: by boxes of position ("NOEUD_BOITE"/"MAILLE_BOITE")
ZI (JREAC+4 (n-1) +1): a number of times where pairing was kept fixed*
ZI (JREAC+4 (n-1) +2): type of projection and normal reactualization geometry/*
+/1: linear projection
+/2: quadratic projection
> 0
so normal and recomputed co-ordinates
< 0
if not

ZI (JREAC+4 (n-1) +3): not used*

3.3 Table
APPARI (addresses JAPPAR)

ZI (JAPPAR) = NESCL: numbers effective nodes slaves

For the iesclième node slave
ZI (JAPPAR+3 (iescl-1) +1): index in CONTNO of the node slave*
ZI (JAPPAR+3 (iescl-1) +2): index in main CONTMA of the paired mesh*
(negative if nodal pairing: opposed index in CONTNO of the paired main node)
(0 if not of pairing)
ZI (JAPPAR+3 (iescl-1) +3): indicator of reactualization*
0
: no the reactualization of projection
+1
: reactualized linear projection + normal
+2
: reactualized quadratic projection + normal
-1
: reactualized linear projection, not normals (not used)
-2

3.4 Table
APMEMO (addresses JAPMEM)

For the posnoième node of CONTNO (slave or not)

ZI (JAPMEM+4 (posno-1)) : 1 if the node is slave*
0 if the node is a Master
-1 if the node is excluded (belongs to SNAS_GROUP_NO)
-2 if the node is excluded (null pivot during symmetrical pairing)
ZI (JAPMEM+4 (posno-1) +1): index in CONTNO of the main node nearest the last time that*
this node was slave
ZI (JAPMEM+4 (posno-1) +2): index in main CONTMA of the paired mesh the last time that it*
node was slave
ZI (JAPMEM+4 (posno-1) +3): number of the box of current position*

3.5 Pointer
APPOIN and tables APCOEF, APCOFR and APDDL (addresses
JAPPTR, JAPCOE, JAPCOF and JAPDDL)

Tables APCOEF and APDDL have same dimension (a coefficient by ddl implied). They are subscripted by
even pointer APPOIN.

For the iesclième node slave:

ZI (JAPPTR+iescl-1) + 1: beginning of the arrangement in APCOEF and APDDL
ZI (JAPPTR+iescl)
: end of the arrangement in APCOEF and APDDL

ZI (JAPPTR+iescl) - ZI (JAPPTR+iescl-1) = nbddl1 + nap of the nbddl2 of the main nodes
nbddl1 = a number of ddls of the node slave of index posno1 in CONTNO:
nbddl1 = ZI (JPDDL+posno1) - ZI (JPDDL+posno1-1),
with posno1 = ZI (JAPPAR+3 (iescl-1) +1)*
nbddl2 = a number of ddls of each main node of index posno2 in CONTNO:
nbddl2 = ZI (JPDDL+posno2) - ZI (JPDDL+posno2-1)

jdec1 = ZI (JAPPTR+iescl-1)
ZI (JAPDDL+jdec1+k-1), k=1, nbddl1:
number of the kth ddl of the node slave
ZR (JAPCOE+jdec1+k-1), k=1, nbddl1:
coefficient associated with the kth ddl with the node slave

jdec2 = ZI (JAPPTR+iescl-1) + nbddl1
for m = 1, nmaitr (nmaitr main nodes with each one nbddl2 ddls) and K = 1, nbddl2
*ZI (JAPDDL+jdec2+ (M-1) *nbddl2+k-1): number of the kth ddl of the main mièmenoeud*
*ZR (JAPCOE+jdec2+ (M-1) *nbddl2+k-1): coefficient associated with the kth ddl with the main mièmenoeud*

Tables APCOFR and APDDL, subscripted by same pointer APPOIN, are used in the case of presence of
friction. The arrangement is exactly the same one as in what precedes with the details close following:

· APCOFR connects the ddl nodes Master and slave concerning displacements in
tangent plan on the surface of contact
· APCOFR contains 60*NESMAX terms is twice as much as APCOEF. Indeed, of 1 with 30*NESMAX
the relations in a direction of the tangent plan are stored, of 30*NESMAX+1 with 60*NESMAX are stored relations in the orthogonal direction with the preceding one in the tangent plan (useful only in 3D).

NB:

In the case of the contact between two surfaces, the coefficients of the ddls are then multiplied by components of the entering normal of the mesh Master paired. In the case of the rigid contact, them coefficients are then multiplied by the components of the normal outgoing slave.

3.6 Table
NORINI (addresses JNRINI)
Components of the current normal to the posnoième node: ZI (JNRINI+3* (posno-1) +k-1), k=1,3

3.7 Table
NORMCO (addresses JNORMO)
For the iesclième node slave

ZI (JNORMO+3* (iescl-1) +k-1), k=1,3: components of the direction (normalized) of evaluation of the play normal.

3.8 Table
TANGCO (addresses JTANGO)
For the iesclième node slave

ZI (JTANGO+6* (iescl-1) +k-1), k=1,3: components of the first direction (normalized) of evaluation of the tangent play.
ZI (JTANGO+6* (iescl-1) +k-1), k=4,6: components of the second direction (normalized) of evaluation of the tangent play.

3.9 Table
APJEU (addresses JAPJEU)

For the iesclième node slave

ZI (JAPJEU+iescl-1):
play enters the node slave and the mesh (or the node) main

or: specified value of the second member (case without pairing)

3.10 Table
COCO (addresses JCOCO)

It contains the memories of the state of preceding contact.

ZI (JCOCO)
= NDIM: dimension of space (2 or 3)
ZI (JCOCO+1) = INDIC: 0 if initialization
+1 if one added a connection
1 if a connection were removed
ZI (JCOCO+2) = NBLIAC: a number of active connections in the preceding state
ZI (JCOCO+3) = AJLIAI: index in the list of the active connections of the last connection having been
calculated for vector CM1A
ZI (JCOCO+4) = SPLIAI: index in the list of the active connections of the last correct line of calculation of the matrix
1
T
A.C-. With
ZI (JCOCO+5) = LLF: a number of connections of adherent friction in the preceding state
ZI (JCOCO+6) = LLF1: a number of connections of adherent friction following the first direction in the preceding state
ZI (JCOCO+7) = LLF: a number of connections of adherent friction following the second direction in the preceding state

3.11 Table
LIAC (addresses JLIAC)

It contains the absolute numbers of the active connections of contact and adherent friction.
The list is not ordered

3.12 Table
CONVEC (addresses JVECC)
It is fixed on table LIAC. It contains the type of the connection:
C0 if it is about a connection in contact
F0 if it is about a connection in adherent friction following the two directions of slips
F1 if it is about a connection in adherent friction following the first direction of slips
F2 if it is about a connection in adherent friction following the second direction of slips
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/04/003/A

3.13 Table
LIOT (addresses JLIOT)

It contains the absolute numbers of the connections of contact-friction causing the appearance of a null pivot
at the time of the resolution. This connection will thus be removed system.

ZI (JLIOT) =
A number of connections of contact to null pivot
ZI (JLIOT+1) with ZI (JLIOT+NBLIAC)
=
Connections of contact to null pivot
ZI (JLIOT+NBLIAI+1) =
A number of connections of friction to null pivot in
directions 1 and 2
ZI (JLIOT+NBLIAI+2) with

=
Connections of friction to null pivot in
*ZI (JLIOT+2*NBLIAC+1)*
directions 1 and 2
*ZI (JLIOT+2*NBLIAI+2) =*
A number of connections of friction to null pivot in
direction 1
*ZI (JLIOT+2*NBLIAI+3) with*
=
Connections of friction to null pivot in the direction
*ZI (JLIOT+3*NBLIAC+2)*
1
*ZI (JLIOT+3*NBLIAI+3) =*
A number of connections of friction to null pivot in
direction 2
*ZI (JLIOT+3*NBLIAI+4) with*
=
Connections of friction to null pivot in the direction
*ZI (JLIOT+4*NBLIAC+3)*
2

Caution:

Each under-vector of LIOT is length NBLIAI, this is why one stores at the beginning of these the last their working length.

3.14 Table
DRIVEN (addresses JMU)

*It contains the multipliers of Lagrange associated with contact-friction. Its maximum length is 6*NESMAX, but its effective length with a given iteration is based on the number of connections active NBLIAC. It is organized as follows:*

ZR (JMU) with ZR (JMU+NBLIAC-1)
= Lagrange of the contact
*ZR (JMU+NBLIAC) with ZR (JMU+2*NBLIAC-1)*
= Lagrange of the adherent connections in the direction
1
*ZR (JMU+2*NBLIAC) with ZR (JMU+3*NBLIAC-1) = Lagrange of the adherent connections in the direction*
2
*ZR (JMU+3*NBLIAC) with ZR (JMU+4*NBLIAC-1) = Lagrange of the sliding joints*
*ZR (JMU+6*NBLIAC-1) = useful Size for the resolution*

3.15 Table
COEFMU (addresses JCMU)

It contains the coefficient by which it is necessary to multiply the multiplier of Lagrange DRIVEN in the routine algoco before testing its sign. This coefficient is worth +1 in the case of a unilateral relation on displacement, a -1 in the case of unilateral relation on the pressure or the temperature of the elements THM (this in order to be coherent with the fact that the hydraulic equation and the thermal equation of problem coupled THM are multiplied by -1).

3.16 Table
ATMU (addresses JATMU)

It contains the nodal reactions of contact, i.e. DRIVEN AT
.
, where A is the matrix of contact. Its dimension is the total number of degrees of freedom of the problem, that is to say NEQ.

3.17 Tables
DELT0 and DELTA (addresses JDELT0 and JDELTA)

They are auxiliary vectors, dimensioned with the total number of degrees of freedom NEQ, used in the algorithm of active constraints.

3.18 Variables
CM1A, MATR and STOC

CM1A is a collection of NBLIAI objects length NEQ: each one of these objects contains one column of the matrix
1
T
C. With, where C is the matrix of tangent rigidity (including/understanding the terms of Lagrange) and With the matrix of contact. These vectors are used in the calculation of the matrix
1
T
A.C-
-
. With, stored in matrix MATR, with a line storage of sky describes by the variable STOC. In these vectors and matrices, matrix A is reduced to the only active connections.

3.19 Variables
CM2A and CM3A

CM2A and CM3A are collections of NBLIAI objects length NEQ: each one of these objects contains a column of the tangent matrices of friction. For more precise details, to refer to the document [R5.03.51].

Document: D4.06.15
STRUCTURES OF DATA CRITNL and CRITTH
1 General information
One wants to store the criteria of resolution, the residues and the iteration counts of a calculation (requiring iterations) in a Structure of Data RESULT.
These criteria are checked in routines NMCRAR and NTCRAR (and are printed on the file MESSAGE).
To make forward these criteria of routines NMCRAR and NTCRAR with the routines of storage NMSTOC and
NTSTOC, one created the Structures of Data CRITNL for mechanical nonlinear calculation and CRITTH for thermal nonlinear calculation.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HI-75/98/003 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Structurse de Données CRITNL and CRITTH
Date:
19/03/98
Author (S):
J. PELLET, L. VIVAN
Key:
D4.06.15-A
Page:
2/4
2 Tree structure
CRITNL
(K19)
:: = record
.CRTI
:
OJB
S V I
LENGTH (2)
1
=
Nb iter_glob
2
=
Nb iter_line

.CRTR
:
OJB
S V R
LENGTH (5)
1
=
resi_glob_rela
2
=
resi_glob
3
=
inco_glob_rela
4
=
eta_pilotage
5
=
inco_pilo_rela
.CRDE
:
OJB
S V K24
LENGTH (7)
1
=
“ITER_GLOB”
2
=
“ITER_LINE”
3
=
“RESI_GLOB_RELA”
4
=
“RESI_GLOB”
5
=
“INCO_GLOB_RELA”
6
=
“ETA_PILOTAGE”

7
=
“INCO_PILO_RELA”
CRITTH
(K19)
:: = record
.CRTI
:
OJB
S V I
LENGTH (2)
1
=
Nb iter_glob
2
=
Nb iter_inte
.CRTR
:
OJB
S V R
LENGTH (3)
1
=
resi_glob_rela
2
=
crit_lagr_rela
3
=
crit_inte_rela
.CRDE
:
OJB
S V K24
LENGTH (5)
1
=
“ITER_GLOB”
2
=
“ITER_INTE”
3

=
"RESI_GLOB_RELA"
4
=
"CRIT_LAGR_RELA"
5
=
"CRIT_INTE_RELA"
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HI-75/98/003 - Ind A

---

3 Contents of the OJB

3.1 Object .CRTI

Vector of entireties storing the iteration counts.

3.2 Objet.CRTR

Vector of realities storing the criteria of resolutions and the residues.

3.3 Objet.CRDE

Vector of K24 describing stored values daN sles objects .CRTI and .CRTR.
The first parameters describe the values of the whole type.
The following describes the values of the real type.

Intentionally white left page.

Code_Aster ®
Version
6.0

Titrate:
SD cabl_precont

Date:
10/12/02
Author (S):
C. CHAVANT*, Key Mr. LAINET*
:
D4.06.16-A Page
: 1/4

Organization (S): EDF-R & D /AMA, CS IF

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
Document: D4.06.16

Structure of Data cabl_precont

Summary

Description of the SD cabl_precont.

This Structure of Data is created by operator DEFI_CABLE_BP and is used in the operator AFFE_CHAR_MECA.

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/02/008/A

---

1 Presentation

The structure of data cabl_precont is produced by operator DEFI_CABLE_BP [U4.42.04]. This operator calculates the initial profiles of tension along the cables of prestressed of a structure concrete, knowing the tension applied at the ends and other parameters characteristic of anchorings and of materials. Each cable is defined by an occurrence of the key word factor DEFI_CABLE.

The structure of data cabl_precont is then used by operator AFFE_CHAR_MECA [U4.44.01], in order to define a mechanical loading of type RELA_CINE_BP, with an aim of calculating the state of balance of the unit structure concrete/cables of prestressing. The resolution is carried out by operator STAT_NON_LINE [U4.51.03], option COMP_INCR.

The structure of data cabl_precont gathers a table, an elementary chart (chart of

initial constraints) and a list of relations (relations kinematics between the DDL of the nodes of cables and DDL of their nodes "close" to the structure concrete).

2
Structure of data

2.1 Tree structure

cabl_precont (K8)

:: = record

".CHME.SIGIN"
:
SD
carte_SIEF_R

`(11)'
:
SD
count

".LIRELA"
:
SD
liste_rela

The table associated with the SD cabl_precont contains the 10 following parameters:

"NUMC_CABLE"
I
"NOEUD_CABLE"

K8
"ABSC_CURV"
R
"ALPHA" R

"TENSION"
R
"MAILLE_BETON_VOISINE"
K8
"NOEUD_BETON_VOISIN" K8
"INDICE_IMMERSION"
I
"INDICE_PROJECTION"
I
"ECCENTRICITY"
R
"NUM_CABLE"
K8

The SD table is described in [D4.02.05].
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/02/008/A

2.2 Object
".CHME.SIGIN"

The chart (constant field by mesh) associated the SD cabl_precont has as a denomination

CABL_PR (K8)/".CHME.SIGIN"

and size SIEF_R represents.

The SD chart is described in [D4.06.05].

2.3 Object
".LIRELA"

The list of relations (SD liste_rela) with the SD cabl_precont has as a denomination

CABL_PR (K8)/".LIRELA"

The SD liste_rela is described in [D4.06.13].
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/02/008/A

Code_Aster ®
Version
6.0

Titrate:
SD cabl_precont

Date:
10/12/02
Author (S):
C. CHAVANT, Key Mr. LAINET
:
D4.06.16-A Page
: 4/4

Intentionally white left page.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/02/008/A

Code_Aster ®
Version
6.3

Titrate:
SD disc_inst

Date:
12/12/02
Author (S):
NR. TARDIEU, E. Key LORENTZ
:
D4.06.17-A Page
: 1/4

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description

D4.06 booklet: Structures related to the finite elements
Document: D4.06.17

Structures of Data disc_inst

Summary

Description of the SD disc_inst.

A SD disc_inst gathers information related to the temporal discretization of a calculation.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/02/008/A

1
The Structure of Data in a few words

The structure of data disc_inst contains all information necessary to description and for the management of a temporal discretization: list moments of calculation, frequency of filing, possible possibility of recutting of the step of time,…

2 Tree structure

Disc_inst (K19):: =record
".DITR"
: OJB S V R8
".DIAL"
: OJB S V L
".DIIR"
: OJB S V R8
".DIEK"
: OJB S V K16

3
Contents of objects JEVEUX

".DITR"
: OJB S V R8 LONG=*

List moments of calculation.

".DIAL"
: OJB S V L LONG=*

List which indicates if it is necessary to file.

".DIIR"
: OJB S V R8 LONG=5

(1)
Next number of filing
(2)
Frequency of filing
(3)
Numbers of under-not by recutting
(4)
Cut of step minimal authorized
(5)
Ratio of 1st under-not

".DIEK"
: OJB S V K16 LONG=*

List which indicates the fields excluded from filing.

4
Methods the SD disc_inst

4.1
OPTILI (RESULT, MODEL, SUBDUES, LISCHA, PARTPS)

RESULT
K8
NAME USER OF THE RESULT
MODEL
MODEL K24
MECANIQUEC
SUBDUE K24
FIELD
MATERIAL
LISCHA
K19 SD
L_CHARGE
PARTPS
K19 SD
DICSCRETISATION

Recutting of the list of mechanical step of time according to the thermal loading.

4.2
DIINIT (INSTAN, PARTPS, DERNIE, RESULT)

INSTAN
R8
INITIAL MOMENT WHEN ETAT_INIT is indicated (OR R8VIDE)
PARTPS
K19 SD
DISCRETIZATION
DERNIE
I
LAST NUMBER FILES (OR 0 SO NOT REENTRANT)
RESULT
K8
NAME USER OF THE RESULT

Initialize the structure of data.

4.3
DIDECO (PARTPS, NUMINS, RETURN)

NUMINS
I: NUMBER
MOMENTS
RETURN
I: CODE
RETURN
0
=
OK
1
=
CUTTING
NOT
ASK

2 = MAXIMUM SMOOTHNESS ATTACK: PROHIBITED RECUTTING

Carry out the cutting of the list of moments.

4.4
DIINST (PARTPS, NUMINS)

PARTPS
K19
: SD DISCRETIZATION
NUMINS
I
:
NUMBER
MOMENTS
DIINST
R8
: VALUE OF THE MOMENT

Turn over the value of the moment.

4.5
DIARCH (PARTPS, NUMINS, FORCE)

PARTPS
K19
: SD DISCRETIZATION
NUMINS
I
:
NUMBER
MOMENTS
FORCE
L
: TRUTH IF ONE WISHES TO FORCE FILING
DIARCH
I
: NUMBER RUNNING Of FILING

= -1 If IT THERE NOT PLACE TO FILE EC NOT TIME

Turn over the number of filing and 1 if there is not.
If FORCE=.TRUE., one files obligatorily
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/02/008/A

Code_Aster ®
Version
6.3

Titrate:
SD disc_inst

Date:
12/12/02
Author (S):
NR. TARDIEU, E. Key LORENTZ
:
D4.06.17-A Page
: 4/4

4.6
DINUAR (PARTPS)

PARTPS
K19
: SD DISCRETIZATION

Return the next number of filing.

4.7
DIINCL (PARTPS, NUMINS, NOMCHZ)

PARTPS
K19
: SD DISCRETIZATION
NUMINS
I
:

NUMBER
From MOMENT
NOMCHA
K16
: NAME OF THE FIELD
DIINCL
L
: TRUTH IF THE FIELD MUST BE SAFE

Return true if the field is to be filed.

4.8
DIDERN (PARTPS, NUMINS)

PARTPS
K19
: SD DISCRETIZATION
NUMINS
I
:
NUMBER
MOMENTS
DIINST
L
: TRUTH IF ONE LEAVES the LIST Of MOMENT

Turn over true if it is about the last moment of calculation.

4.9
DIDECO (PARTPS, NUMINS, RETURN)

PARTPS
K19
: SD DISCRETIZATION
NUMINS
I
:
NUMBER
MOMENTS
RETURN
I
:

CODE
RETURN
0
=
OK
1
=
CUTTING
NOT
ASK

2 = MAXIMUM SMOOTHNESS ATTACK: PROHIBITED RECUTTING

Carry out the cutting of the list of moments.

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/02/008/A

Code_Aster ®
Version
7.4

Titrate:
Structures of data related to materials

Date:
29/09/05
Author (S):
Key ***J.P. LEFEBVRE***
:
D4.06.18-B Page
: 1/18

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
D4.06.18 document

Structures of data related to materials

Summary:

One describes here the structures of data built starting from the description and the value of different parameters associated with the behaviors with a material (compor, MATER, cham_mater, cham_mater_code).

The two types of structures of data are presented:

- ***the SD in access by name: to subdue,***
- ***the SD in access by address: mater_code (which replaces the preceding one in the TExxxx routines).***

Count

matters

---

1 General information

The material appears in many orders of Code_Aster and intervenes in the majority
routines of calculation of the elementary terms (TExxxx).

Indeed, and contrary to other codes, in Code_Aster, one distinguishes the concept of "material",

defined by order DEFI_MATERIAU, and being able to be filed in the form of catalogue, (see command INCLUDE_MATERIAU [U4.43.02]) of the concept of behavior used during a calculation given. The material is composed of values of parameters associated with a certain number of models behaviors. In practice these parameters are defined via simple key words, under key words ratios control DEFI_MATERIAU. These key words factors (for example ELAS, ECRO_LINE, CIN1_CHAB, LEMAITRE) make it possible to define whole or part of the parameters necessary to a model of behavior. For example, the behavior (chosen by the user in STAT_NON_LINE or DYNA_NON_LINE under the key word factor COMP_INCR) VMIS_CINE_LINE *will use the parameters material defined under the key words factor ELAS and ECRO_LINE of DEFI_MATERIAU. Behavior VISC_CIN1_CHAB will use those definite by ELAS, CIN1_CHAB and possibly LEMAITRE.*

By abuse language, in this document, one will call "behavior" a key word factor of DEFI_MATERIAU.

Moreover, certain total orders (MECA_STATIQUE, THER_LINEAIRE,…) do not allow the user to choose the behavior used for calculation. It is necessary thus that the calculation (for example of the matrix of rigidity in the preceding examples) can be done without ambiguity. This is why it is necessary that certain behaviors of comparable nature (ELAS_xxx, THER_xx) are single in material. By convention of language in this document, one will say that the whole of behaviors of comparable nature (even prefix : ELAS, THER) belong to same "phenomenon". Certain "behaviors" can thus be classified by "phenomenon" in which they are excluded mutually.

The structure of data associated with material contains the name and the values associated with parameters describing each behavior. The parameters can be of type real, complex or function. Within this structure of data one reaches the values of the parameters by name.

For reasons of performances of the code (in particular in the case of behaviors non-linear), the coded material was introduced. The structure of data associated with coded material is temporary, it contains the addresses of the various objects constituting the structure of data MATER. The access to the values of the parameters does not require any more in this case a setting in memory of the objects JEVEUX and a research by name with each time one uses material. The access by name to parameters remain nevertheless.

These two structures of data are based on structures of the type function (constant function, function of a parameter or tablecloth) and objects simple JEVEUX of the vector type.

The creation of a structure of data function (. &&RDEP) prefixed by the name of material allows to then have a permanent space of memory necessary to the interpolation of tablecloths defining the traction diagrams depending on the temperature. This structure of data is created on the TOTAL basis in order to be exchanged between the various orders of the code.

29/09/05
Author (S):
*Key **J.P. LEFEBVRE***
:
D4.06.18-B Page
: 4/18

2
The structure of MATER data and cham_mater

The name of the "phenomenon" is the root of the name of the "behavior". In the order where the user does not choose the model of behavior, it is necessary to know to find without ambiguity them material characteristics necessary, for example, the calculation of the matrices of rigidity through order MECA_STATIQUE cannot make the distinction between ELAS and ELAS_ORTHO, this is why these two behaviors are excluded. On the other hand, a material can contain phenomena ELAS and THER. For the mechanical non-linear behaviors, unicity is not necessary, because the user chooses a relation of behavior on the level of the total order (STAT_NON_LINE).

For example, behaviors ELAS, ELAS_FLUI, ELAS_ISTR, ELAS_ORTH, ELAS_THM, ELAS_COQUE constitute phenomenon ELAS.

The structure of MATER data is made up:

.
several phenomena (K10: name of the key word factor "behaviors" truncated with the first 10 characters),
.
possibly of a function &&RDEP for the behaviors TRACTION and META_TRACTION,
.
possibly of a function &&MZP for parameter RELA_MZ of the behavior DIS_CONTACT.

to subdue (K8):: = record

".MATERIAU.NOMRC": OJB S V K16

% Behavior General Rubber bands

|
/".ELAS"
:
COMPOR
/
'.ELAS_FLUI
'
:
COMPOR
/

'.ELAS_ISTR
'
:
COMPOR
/
'.ELAS_ORTH
'
:
COMPOR
/
'.ELAS_THM

'
:
COMPOR
/
".ELAS_COQUE"
:
COMPOR
-
-

% General Nonlinear Mechanical Behaviors

|

".TRACTION"
:
COMPOR

". &&RDEP"
:
FUNCTION

| *'.ECRO_LINE*
'
:
COMPOR

|

".PRAGER"
:
COMPOR

| *'.CIN1_CHAB*

'
:
COMPOR

| '.CIN2_CHAB
'
:
COMPOR

|

“.TAHERI”
:
COMPOR

|

“.POLY_CFC”
:
COMPOR

|
/“.LEMAITRE”
:
COMPOR
/
“.ZIRC_CYRA2”
:
COMPOR
/
'.ZIRC_EPRI
'
:
COMPOR

|

“.LMARC”
:
COMPOR

| “.NORTON_HOF”
:
COMPOR

% Behaviors related to the damage and the rupture

...

Note:

The structure of MATER data does not contain information on the form of the laws of behavior: elasticity, Lemaitre, etc… These last do not exist in the form of structures of data, but only in “hard” in FORTRAN.

2.1
Contents of the s.d. to subdue

.MATERIAU.NOMRC:
S V K16

*Vector of the type CHARACTER*16 dimensioned with the number of behaviors present at the time of definition of material.*

The name of the behaviors affected in order DEFI_MATERIAU contains or DEFI_COQU_MULT.

Order DEFI_COQU_MULT stores there the name of behavior ELAS_COQMU or THER_COQMU, a white chain and the name of each material for each layer. Parameter LONMAX (attribute length of associated object JEVEUX) of this object is recovered in various routines to obtain the number of layers.

. &&RDEP
:
FUNCTION

Is present only if behaviors TRACTION and META_TRACTION appear in material. It is related to a variable dimensioned with the maximum number of points of the functions composing the traction diagrams depending on the temperature.

. &&MZP
:
FUNCTION

Is present only if parameter RELA_MZ of behavior DIS_CONTACT appears in material.
It is a function representing the curve (moment) according to DR. (degree of rotation).
The material such as it was defined above is not usable for elementary calculations, it is necessary still to assign this last to groups of meshs. The structure of data cham_mater allows to define this relation.

Cham_mater (K8):: = record

".CHAMP_MAT"
:
carte_neut_f

".TEMP_REF"
:
carte_temp_r
".SECH_REF"
:
carte_temp_r

Note:

It can be necessary to repeat several times the affection of the same material with one temperature of different reference.

In the case of the mono and polycrystalline materials, it can be necessary to define several materials by assignment (order DEFI_COMPOR allowing for choice of them characteristics material to be affected). This is why size associated with the CHAMP_MAT is NEUT_F, which has 30 CMP. In practice one will not be able definite more than 30 materials in a given place (mesh or group_ma).

3
The structure of data compor

One defines first of all a structure of data related to each behavior which can define it material. A behavior is a whole of named parameters (K8) associated a value. If value is K8 the parameter is associated a function.

compor (K19):: = record

".VALK":
OBJ S V K8

".VALR":
OBJ S V R

".VALC":
OBJ S V C

The complete name of the structure of data compor (K19) is consisted the name user of material (K8) follow-up of one "." follow-up of the first ten characters of the key word factor appearing in catalogue order DEFI_MATERIAU.

It is thus imperative to differentiate all the key words control ratios DEFI_MATERIAU on the first ten characters.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

3.1
Contents of the S.D. compor

VALK:

S V K8

*Vector of the type CHARACTER *8 dimensioned to 2 times the maximum number of parameters (it is necessary to be able to store all the names of parameters and all the names of functions, if all parameters are associated functions (2*NBPAR)) for the law of behavior considered. For example E and NAKED are parameters of elastic behavior ELAS.*

Contains in the order:

.
names of the parameters associated with actual values,
.
names of the parameters associated with complex values,
.
names of the parameters associated with functions,
.

names of the functions.

Order DEFI_COQU_MULT fills this object with names with parameters associated with coefficients homogenized like for each layer. An actual value is stored for each one.

.VALR:
S V R

*Vector of the type REAL*8 dimensioned with the maximum number of parameters (NBPAR) for the law of behavior considered.*

Contains the values associated with the real parameters.

.VALC:
S V C

*Vector of the type COMPLEX*16 dimensioned with the maximum number of parameters (NBPAR) for the law of behavior considered.*

Contains the values associated with the complex parameters.

4
The structure of data mater_code and cham_mater_code

It is a temporary structure of data (created on the VOLATILE basis) containing the addresses memory of objects JEVEUX constituting a material (SD MATER).

Mater_code (K19):: = record

".CODI"

: OBJ S V I

The name of this structure of data is indexed on the occurrence of material in the cham_mater.

It is the analogue of the structure of data cham_mater for a coded material. The chart created on base TOTAL refers to objects of the VOLATILE base, it must thus be reactualized in continuation.

Cham_mater_code (K8):: = record

'. .MATE_CODE
=
carte_adrsjeve
(I)
Data-processing handbook of Description

4.1
mater_code

.CODI:
S V I

Vector of the type INTEGER whose dimension depends on the number of behaviors described in structures of data MATER Object JEVEUX temporary created on the VOLATILE basis.

Parameters are associated coded material

LMAT:
a number of parameters associated with the behavior
LFCT:
a number of parameters associated with the concepts (functions, tables)
LSUP:
a number of additional parameters (functions &&RDEP, &&MZP)

*This vector is length 2 + NBCM*LMAT + NBCO*LFCT + NBT*LSUP where*

NBCM:
a number of behaviors present in material
NBCO:
a number of concepts (functions, tables) present in material
NBT:
a number of traction diagrams present in material

This vector contains the addresses memory of the objects composing the structure of data MATER.

CODI (1):
A number of different materials (N)
CODI (2):
Index in CODI of first material.
CODI (3):
Index in CODI of second material (if necessary)
.....
....

CODI (n+1):
Index in CODI of nth material (if necessary)

CODI (n+2):
address .MATERIAU.NOMRC. (for first material)
CODI (n+3):
NBCM numbers behaviors present in
material.
CODI (n+2+1: n+2+NBCM):
pointer of the Kth behavior in CODI, for K=1 with
NBCM

for each Kth behavior of material 1

that is to say ipi = CODI (n+2+K)

CODI (n+2+k):
ipi, pointer of the Kth behavior
CODI (ipi):
a number of parameters associated with realities.
CODI (ipi+1):
a number of parameters associated with complexes.
CODI (ipi+2):
a number of parameters associated with concepts (functions, tables)
CODI (ipi+3):
address memory of object .VALK.
CODI (ipi+4):
address memory of object .VALR.
CODI (ipi+5):
address memory of object .VALC.

for Lième concept of the type counts associated with a parameter with the Kth behavior,
that is to say ipif = ipi+LMAT-1

CODI (ipif+LFCT (L-1)) :*
the table is transformed into a list of realities
(LIST_R8)
address memory of object .VALE of the LIST_R8
CODI (ipif+LFCT (L-1) +1):*
0
CODI (ipif+LFCT (L-1) +2):*
0
CODI (ipif+LFCT (L-1) +3): ISNNEM ()*
CODI (ipif+LFCT (L-1) +4): ISNNEM ()*
CODI (ipif+LFCT (L-1) +5): ISNNEM ()*
CODI (ipif+LFCT (L-1) +6): ISNNEM ()*
CODI (ipif+LFCT (L-1) +7): ISNNEM ()*
CODI (ipif+LFCT (L-1) +8): ISNNEM ()*

---

for Lième concept of the function type associated with a parameter with the Kth behavior,
that is to say ipif = ipi+LMAT-1

CODI (ipif+LFCT (L-1)) :*
a number of points of the associated function.
CODI (ipif+LFCT (L-1) +1):*
address memory of object .PROL.
CODI (ipif+LFCT (L-1) +2):*
address memory of object .VALE.
CODI (ipif+LFCT (L-1) +3): address memory of the pointer length for one tablecloth.*
CODI (ipif+LFCT (L-1) +4): address memory of object .PARA for a tablecloth*
CODI (ipif+LFCT (L-1) +5): attribute LONUTI of object .PARA for a tablecloth.*
CODI (ipif+LFCT (L-1) +6): pointer in CODI for the traction diagrams &&RDEP or for the fuel assemblies &&MZP*
CODI (ipif+LFCT (L-1) +7): safeguard index of the interval of interpolation.*
CODI (ipif+LFCT (L-1) +8): safeguard of an index of research (equation not linear in thermics).*

that is to say ipifc = CODI (ipif+LFCT (L-1) +6)*

CODI (ipifc):
address memory of the &&MZP.PROL object.
CODI (ipifc+1):
address memory of the &&MZP.VALE object.

or

CODI (ipifc):
address memory of the &&RDEP.PROL object.
CODI (ipifc+1):
address memory of the &&RDEP.VALE object.

Titrate:
Structures of data related to materials

5 Example (SD to subdue and SD cham_mater)

5.1 Command file

The orders below make it possible to define 3 laws of behavior: elasticity, plasticity with traction diagram depending on the temperature and linear thermics.

```
...
MA=LIRE_MAILLAGE ()
#
# given of modeling
#
F_E = DEFI_FONCTION (NOM_PARA = “TEMP”,
PROL_DROITE = “LINEAR”,
PROL_GAUCHE = “LINEAR”,
VALE = (0. , 200.E+03,
50. , 198.E+03,),
)
#
F_NU = DEFI_CONSTANTE (VALE = 0.3)
#
F_AL = DEFI_CONSTANTE (VALE = 10.E+06)
#
FCT1 = DEFI_FONCTION (NOM_PARA = “EPSI”,
PROL_DROITE = “LINEAR”,
PROL_GAUCHE = “LINEAR”,
VALE = (0.200E-02, 400. ,
0.400E-02, 500. ,),
)
#
FCT2 = DEFI_FONCTION (NOM_PARA = “EPSI”,
PROL_DROITE = “LINEAR”,
PROL_GAUCHE = “LINEAR”,
VALE = (0.100E-02, 200. ,
0.300E-02, 300. ,),
)
#
CTRACB = DEFI_NAPPE (NOM_PARA = “TEMP”,
PROL_DROITE = “LINEAR”,
PROL_GAUCHE = “LINEAR”,
PARA = (0. , 50. ,),
FUNCTION = (FCT1, FCT2,),
)
#
```

 # material isotropic
#
CHECHMATE = DEFI_MATERIAU (THER =_F (RHO_CP = 0.0E-03, LAMBDA = 1.0E-03,),
ELAS_FO =_F (E = F_E, NAKED = F_NU, ALPHA =
F_AL, TEMP_DEF_ALPHA=20.0,),
TRACTION =_F (SIGM = CTRACB,),
)
#
IMPR_CO (CO = CHECHMATE)
#
MAT2 = DEFI_MATERIAU (ELAS_FO =_F (E = F_E, NAKED = F_NU, ALPHA = F_AL,

TEMP_DEF_ALPHA=40.),

ECRO_LINE=_F (SY=200., D_SIGM_EPSI=2000.)
)
#
##IMPR_CO (CO = MAT2)
#
CHMAT = AFFE_MATERIAU (MAILLAGE=MA,
AFFE= (_F (GROUP_MA = ("GM1",),
MATER = (CHECHMATE, MAT2), TEMP_REF = 20.),
_F (GROUP_MA = ("GM2",),
MATER = CHECHMATE, TEMP_REF = 50.),),
)
#
IMPR_CO (CO = CHMAT)

#
....
SOLNL=STAT_NON_LINE (MODELE=EPD3,
CHAM_MATER=CHMAT,
EXCIT=_F (CHARGE=COND_LIM,
TYPE_CHARGE=' FIXE_CSTE',),
COMP_INCR=_F (RELATION=' ELAS',
DEFORMATION=' PETIT',

TOUT=' OUI',),
INCREMENT=_F (LIST_INST=LINST,
NUME_INST_FIN=1,

),
TITRE=' TEST',);

5.2
Impression of the structure of data

STRUCTURE OF DATA: CHMAT?????????????????

====> IMPR_CO OF THE STRUCTURE OF DATA: CHECHMATE????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 12

==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >MAT. &&RDEP .PROL <
>>>>>
1 - >FONCTION <>LIN FLAX <>EPSI <
4 - >TOUTRESU <> <> <

SEGMENT IMPRESSION OF VALUES >MAT. &&RDEP .VALE <
>>>>>
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .ELAS .VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
3 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
5 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
7 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .ELAS .VALK <
>>>>>
1 - >B_ENDOGE<>K_DESSIC<>PRECISIO<>TEMP_DEF<>E <>ALPHA <>NU <
8 - >F_E <>F_AL <>F_NU <> <> <> <> <
15 - > <> <

SEGMENT IMPRESSION OF VALUES >MAT .ELAS .VALR <
>>>>>
1 - 0.00000E+00 0.00000E+00 1.00000E+00 2.00000E+01 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .MATERIAU.NOMRC <

>>>>>
1 - >TRACTION <>THER <>ELAS <

SEGMENT IMPRESSION OF VALUES >MAT .THER .VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .THER .VALK <
>>>>>

1 - >RHO_CP <>LAMBDA <> <> <

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

SEGMENT IMPRESSION OF VALUES >MAT .THER .VALR <
>>>>>

1 - 0.00000E+00 1.00000E-03

SEGMENT IMPRESSION OF VALUES >MAT .TRACTION .VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .TRACTION .VALK <
>>>>>

1 - >SIGM <>CTRACB <> <> <

SEGMENT IMPRESSION OF VALUES >MAT .TRACTION .VALR <
>>>>>

1 - 0.00000E+00 0.00000E+00
====> FINE IMPR_CO OF STRUCTURE OF DATA: CHECHMATE?????????????????

====> IMPR_CO OF THE STRUCTURE OF DATA: CHMAT?????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 10

==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >CHMAT .CHAMP_MAT .DESC <
>>>>>
1 - 76 2 2 3 1
6 - 3 2 6 2

IMPRESSION OF THE COLLECTION: CHMAT .CHAMP_MAT .LIMA

OBJECT IMPRESSION OF COLLECTION CONTIGUE>CHMAT .CHAMP_MAT .LIMA< OC: 1
>>>>>

1 - 1

OBJECT IMPRESSION OF COLLECTION CONTIGUE>CHMAT .CHAMP_MAT .LIMA< OC: 2
>>>>>

1 - 2

SEGMENT IMPRESSION OF VALUES >CHMAT .CHAMP_MAT .NOLI <
>>>>>
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CHMAT .CHAMP_MAT .NOMA <
>>>>>

1 - >MA <

SEGMENT IMPRESSION OF VALUES >CHMAT .CHAMP_MAT .VALE <
>>>>>
1 - >MAT <>MAT2 <> <> <> <> <> <
8 - > <> <> <> <> <> <> <
15 - > <> <> <> <> <> <> <
22 - > <> <> <> <> <> <> <
29 - > <> <>MAT <> <> <> <> <
36 - > <> <> <> <> <> <> <
43 - > <> <> <> <> <> <> <
50 - > <> <> <> <> <> <> <
57 - > <> <> <> <

SEGMENT IMPRESSION OF VALUES >CHMAT .TEMPE_REF .DESC <
>>>>>
1 - 108 2 2 3 1

6 - 3 2 2 2

IMPRESSION OF THE COLLECTION: CHMAT .TEMPE_REF .LIMA

OBJECT IMPRESSION OF COLLECTION CONTIGUE>CHMAT .TEMPE_REF .LIMA< OC: 1
>>>>>

1 - 1

OBJECT IMPRESSION OF COLLECTION CONTIGUE>CHMAT .TEMPE_REF .LIMA< OC: 2
>>>>>

1 - 2

SEGMENT IMPRESSION OF VALUES >CHMAT .TEMPE_REF .NOLI <
>>>>>
1 - > <> <

SEGMENT IMPRESSION OF VALUES >CHMAT .TEMPE_REF .NOMA <
>>>>>

1 - >MA <

SEGMENT IMPRESSION OF VALUES >CHMAT .TEMPE_REF .VALE <
>>>>>
1 - 2.00000E+01 0.00000E+00 0.00000E+00 0.00000E+00 5.00000E+01
6 - 0.00000E+00 0.00000E+00 0.00000E+00
====> FINE IMPR_CO OF STRUCTURE OF DATA: CHMAT????????????????

Moreover the S.D mater_code was printed in STA_NON_LINE (routine RCMFMC):

====> IMPR_CO OF THE STRUCTURE OF DATA: CHECHMATE????????????????????
ATTRIBUTE: T CONTENTS: T BASE: > <
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 32

==
IMPRESSION OF THE ATTRIBUTES OF THE OBJECTS FIND:

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT. &&RDEP .PROL<
> <
CLAS STANDARD >G< GENR >V< >K< LTYP 16 DOCU > < DATE 0 ORIG
> < LONMAX 6 LONUTI 6 LONO 6 IADM 2166556 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT. &&RDEP .VALE<
> <
CLAS STANDARD >G< GENR >V< >R< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 8 LONUTI 8 LONO 8 IADM 2166592 IADD 0 LADD 0 USE >U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0001 .CODI<
> <
CLAS STANDARD >V< GENR >V< >I< LTYP 4 DOCU > < DATE 0 ORIG > <

LONMAX 109 LONUTI 109 LONO 109 IADM 2255598 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0002 .CODI<
> <
CLAS STANDARD >V< GENR >V< >I< LTYP 4 DOCU > < DATE 0 ORIG > <
LONMAX 64 LONUTI 64 LONO 64 IADM 2256097 IADD 0 LADD 0 USE >U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0010001001.VALC<
> <
CLAS STANDARD >V< GENR >V< >C< LTYP 16 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2255210 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0010001001.VALK<
> <
CLAS STANDARD >V< GENR >V< >K< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 4 LONUTI 2 LONO 4 IADM 2255230 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0010001001.VALR<
> <
CLAS STANDARD >V< GENR >V< >R< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2255196 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0010002001.VALC<
> <
CLAS STANDARD >V< GENR >V< >C< LTYP 16 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2255895 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0010002001.VALK<
> <
CLAS STANDARD >V< GENR >V< >K< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 4 LONUTI 2 LONO 4 IADM 2255915 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0010002001.VALR<
> <
CLAS STANDARD >V< GENR >V< >R< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2255881 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0020001001.VALC<
> <
CLAS STANDARD >V< GENR >V< >C< LTYP 16 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2255262 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0020001001.VALK<
> <
CLAS STANDARD >V< GENR >V< >K< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 4 LONUTI 2 LONO 4 IADM 2255282 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0020001001.VALR<
> <
CLAS STANDARD >V< GENR >V< >R< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 2 LONO 2 IADM 2255248 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0020002001.VALC<
> <
CLAS STANDARD >V< GENR >V< >C< LTYP 16 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2255947 IADD 0 LADD 0 USE

:
D4.06.18-B Page
: 14/18

>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0020002001.VALK<
> <
CLAS STANDARD >V< GENR >V< >K< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 4 LONUTI 2 LONO 4 IADM 2255967 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0020002001.VALR<
> <
CLAS STANDARD >V< GENR >V< >R< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 2 LONUTI 2 LONO 2 IADM 2255933 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0030001001.VALC<
> <
CLAS STANDARD >V< GENR >V< >C< LTYP 16 DOCU > < DATE 707052303 ORIG
> < LONMAX 8 LONUTI 0 LONO 8 IADM 2255326 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0030001001.VALK<
> <
CLAS STANDARD >V< GENR >V< >K< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 16 LONUTI 10 LONO 16 IADM 2255370 IADD 0 LADD THE 0 USA
GE >U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0030001001.VALR<
> <
CLAS STANDARD >V< GENR >V< >R< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 8 LONUTI 4 LONO 8 IADM 2255300 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0030002001.VALC<
> <
CLAS STANDARD >V< GENR >V< >C< LTYP 16 DOCU > < DATE 707052303 ORIG
> < LONMAX 8 LONUTI 0 LONO 8 IADM 2256011 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0030002001.VALK<
> <
CLAS STANDARD >V< GENR >V< >K< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 16 LONUTI 10 LONO 16 IADM 2256055 IADD 0 LADD THE 0 USA
GE >U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .0030002001.VALR<

> <
CLAS STANDARD >V< GENR >V< >R< LTYP 8 DOCU > < DATE 707052303 ORIG
> < LONMAX 8 LONUTI 4 LONO 8 IADM 2255985 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .ELAS .VALC<
> <
CLAS STANDARD >G< GENR >V< >C< LTYP 16 DOCU > < DATE 0 ORIG
> < LONMAX 8 LONUTI 0 LONO 8 IADM 2166696 IADD 0 LADD 0 USE
>X D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .ELAS .VALK<
> <
CLAS STANDARD >G< GENR >V< >K< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 16 LONUTI 10 LONO 16 IADM 2166740 IADD 0 LADD 0 USE >U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .ELAS .VALR<
> <
CLAS STANDARD >G< GENR >V< >R< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 8 LONUTI 4 LONO 8 IADM 2166670 IADD 0 LADD 0 USE >U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .MATERIAU.NOMRC <
> <
CLAS STANDARD >G< GENR >V< >K< LTYP 16 DOCU > < DATE 0 ORIG
> < LONMAX 3 LONUTI 3 LONO 3 IADM 2166410 IADD 0 LADD 0 USE
>U D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .THER .VALC<
> <
CLAS STANDARD >G< GENR >V< >C< LTYP 16 DOCU > < DATE 0 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2166632 IADD 0 LADD 0 USE
>X D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .THER .VALK<
> <
CLAS STANDARD >G< GENR >V< >K< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 4 LONUTI 2 LONO 4 IADM 2166652 IADD 0 LADD 0 USE >X D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .THER .VALR<
> <
CLAS STANDARD >G< GENR >V< >R< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 2 LONUTI 2 LONO 2 IADM 2166618 IADD 0 LADD 0 USE >X D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .TRACTION .VALC<
> <
CLAS STANDARD >G< GENR >V< >C< LTYP 16 DOCU > < DATE 0 ORIG
> < LONMAX 2 LONUTI 0 LONO 2 IADM 2166470 IADD 0 LADD 0 USE
>X D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .TRACTION .VALK<
> <
CLAS STANDARD >G< GENR >V< >K< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 4 LONUTI 2 LONO 4 IADM 2166490 IADD 0 LADD 0 USE >X D<

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >MAT .TRACTION .VALR<
> <
CLAS STANDARD >G< GENR >V< >R< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 2 LONUTI 0 LONO 2 IADM 2166456 IADD 0 LADD 0 USE >X D<

==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

SEGMENT IMPRESSION OF VALUES >MAT. &&RDEP .PROL <
>>>>>
1 - >FONCTION <>LIN FLAX <>EPSI <
4 - >TOUTRESU <> <> <

SEGMENT IMPRESSION OF VALUES >MAT. &&RDEP .VALE <
>>>>>
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .0001 .CODI <
>>>>>
1 - 2 -46973472 -46973471 5 66
6 - -23508908 3 -93946422 -93946405 -93946399
11 - 0 0 1 -46973407 -46973422
16 - -23486706 2147483647 -23509252 -47018426 -94036872
21 - -47018476 2 -93946407 1 1
26 - -23508872 -47017724 2 0 0
31 - -46973381 -46973396 -23486693 4 0

36 - 3 -46973337 -46973370 -23486677 2
41 - -23510835 -47021644 2147483647 2147483647 2147483647
46 - 2147483647 1 1 1 -23486580
51 - -46973142 2147483647 2147483647 2147483647 2147483647
56 - 1 1 1 -23510520 -47021022
61 - 2147483647 2147483647 2147483647 2147483647 1
66 - 1 -23508815 2 -93946362 -93946356
71 - 2 0 0 -46973288 -46973303
76 - -23486646 4 0 3 -46973244

---

81 - -46973277 -23486630 2 -23510835 -47021644
86 - 2147483647 2147483647 2147483647 2147483647 1
91 - 1 1 -23486568 -46973119 2147483647
96 - 2147483647 2147483647 2147483647 1 1
101 - 1 -23510520 -47021022 2147483647 2147483647
106 - 2147483647 2147483647 1 1
--

SEGMENT IMPRESSION OF VALUES >MAT .0002 .CODI <
>>>>>
1 - 1 -46973470 3 -23508908 3
6 - -93945925 -93945908 -93945902 0 0
11 - 1 -46973065 -46973080 -23486535 2147483647
16 - -23509252 -47018426 -94036872 -47018476 2
21 - -93945910 1 1 -23508872 -47017724
26 - 2 0 0 -46973039 -46973054
31 - -23486522 4 0 3 -46972995
36 - -46973028 -23486506 2 -23510835 -47021644
41 - 2147483647 2147483647 2147483647 2147483647 1
46 - 1 1 -23486469 -46972920 2147483647
51 - 2147483647 2147483647 2147483647 1 1
56 - 1 -23510520 -47021022 2147483647 2147483647
61 - 2147483647 2147483647 1 1
--

SEGMENT IMPRESSION OF VALUES >MAT .0010001001.VALC <
>>>>>

1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .0010001001.VALK <
>>>>>

1 - >SIGM <>CTRACB <> <> <

SEGMENT IMPRESSION OF VALUES >MAT .0010001001.VALR <
>>>>>

1 - 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .0010002001.VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .0010002001.VALK <
>>>>>

1 - >SIGM <>CTRACB <> <> <

SEGMENT IMPRESSION OF VALUES >MAT .0010002001.VALR <
>>>>>

1 - 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .0020001001.VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .0020001001.VALK <
>>>>>

1 - >RHO_CP <>LAMBDA <> <> <

SEGMENT IMPRESSION OF VALUES >MAT .0020001001.VALR <
>>>>>

1 - 0.00000E+00 1.00000E-03

SEGMENT IMPRESSION OF VALUES >MAT .0020002001.VALC <
>>>>>

1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .0020002001.VALK <
>>>>>

1 - >RHO_CP <>LAMBDA <> <> <

SEGMENT IMPRESSION OF VALUES >MAT .0020002001.VALR <
>>>>>

1 - 0.00000E+00 1.00000E-03

SEGMENT IMPRESSION OF VALUES >MAT .0030001001.VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
3 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
5 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
7 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .0030001001.VALK <
>>>>>
1 - >B_ENDOGE<>K_DESSIC<>PRECISIO<>TEMP_DEF<>E <>ALPHA <>NU <
8 - >F_E <>&&000000<>F_NU <> <> <> <> <
15 - > <> <

SEGMENT IMPRESSION OF VALUES >MAT .0030001001.VALR <
>>>>>
1 - 0.00000E+00 0.00000E+00 1.00000E+00 2.00000E+01 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .0030002001.VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
3 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
5 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
7 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .0030002001.VALK <
>>>>>
1 - >B_ENDOGE<>K_DESSIC<>PRECISIO<>TEMP_DEF<>E <>ALPHA <>NU <
8 - >F_E <>&&000002<>F_NU <> <> <> <> <
15 - > <> <

SEGMENT IMPRESSION OF VALUES >MAT .0030002001.VALR <
>>>>>
1 - 0.00000E+00 0.00000E+00 1.00000E+00 2.00000E+01 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .ELAS .VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
3 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
5 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)
7 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .ELAS .VALK <
>>>>>
1 - >B_ENDOGE<>K_DESSIC<>PRECISIO<>TEMP_DEF<>E <>ALPHA <>NU <
8 - >F_E <>F_AL <>F_NU <> <> <> <> <
15 - > <> <

SEGMENT IMPRESSION OF VALUES >MAT .ELAS .VALR <
>>>>>
1 - 0.00000E+00 0.00000E+00 1.00000E+00 2.00000E+01 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00

SEGMENT IMPRESSION OF VALUES >MAT .MATERIAU.NOMRC <
>>>>>
1 - >TRACTION <>THER <>ELAS <

SEGMENT IMPRESSION OF VALUES >MAT .THER .VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .THER .VALK <
>>>>>

1 - >RHO_CP <>LAMBDA <> <> <

SEGMENT IMPRESSION OF VALUES >MAT .THER .VALR <
>>>>>

1 - 0.00000E+00 1.00000E-03

SEGMENT IMPRESSION OF VALUES >MAT .TRACTION .VALC <
>>>>>
1 - (0.00000E+00, 0.00000E+00) (0.00000E+00, 0.00000E+00)

SEGMENT IMPRESSION OF VALUES >MAT .TRACTION .VALK <
>>>>>

1 - >SIGM <>CTRACB <> <> <

SEGMENT IMPRESSION OF VALUES >MAT .TRACTION .VALR <
>>>>>

1 - 0.00000E+00 0.00000E+00
====> FINE IMPR_CO OF STRUCTURE OF DATA: CHECHMATE????????????????????

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
6.0

Titrate:
SD matr_elem and vect_elem

Date:
09/10/01
Author (S):
J.** Key **PELLET
:
D4.06.20-A Page
: 1/4

Organization (S): EDF/MTI/MMN

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
Document: D4.06.20

Structures of Data matr_elem and vect_elem

Summary

Description of the SD matr_elem and vect_elem.

A SD matr_elem represents a whole of elementary matrices.
A SD vect_elem represents a whole of elementary vectors.

1
Structures of Data in a few words

A matr_elem (resp. vect_elem) is a whole of elementary matrices (resp. vectors elementary) that I' one can assemble to obtain one matr_asse (resp. a cham_no)

Practically, the matr_elem and the vect_elem consist of a whole of resuelem [D4.06.05]. A resuelem being the whole of the matrices (or vectors) elementary correspondents with the elements of a ligrel.

A matr_elem (or a vect_elem) can not contain any resuelem. That can arrive if the model contains only static substructures.

2 Tree structure

matr_elem (K8):: =record
".REFE_RESU"

: OJB S V LONG K24 ())
".LISTE_RESU": OJB S V indirect K24 (*)
(*)
resuelem

vect_elem (K8):: =record
".REFE_RESU"

: OJB S V LONG K24 ())
| ".LISTE_RESU": OJB S V indirect K24 (*)
(*)
resuelem
|
".LISTE_CHAR"
: OJB XC V I NO () LONG (cste)

3 Contents of objects JEVEUX

3.1 ".REFE_RESU": S V LONG K24 (5)

That is to say V = ".REFE_RESU",

V (1)
name of the subjacent model
V (2)
name of the attached on-option: "RIGI_MECA", "MASS_THER", "CHAR_MECA",…
V (3)/
"OUI_SOUS_STRUC"
/"NON_SOUS_STRUC"

V (3)
Indicate if the elementary terms (matrices or vectors) of the static substructures are to be taken into account (or not). For example, one should not take into account them substructures if rigidity were calculated by the only loads of blocking.

V (4)
Name of the cham_mater subjacent with the matr_elem (or vect_elem).
V (5)

Name of the cara_elem subjacent with the matr_elem (or vect_elem).

Object .REFE_RESU is obtained by calling routine MEMARE.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HI-75/01/005/A

3.2
".LISTE_RESU": S V Indirect K24 (*)

This object contains the list of the resuelem composing the matr_elem (or the vect_elem).

That is to say V = '.LISTE_RESU,

V (I) (1: 19): name of the ième resuelem of the matr_elem (or of the vect_elem).

The number of resuelem is obtained by "LONUTI" of object .LISTE_RESU.

3.3
".LISTE_CHAR": XC V I NO () LONG (cste)

This object exists only if the grid contains super-meshs. This collection is named by
loading case indicated by the user in order CALC_VECT_ELEM.

That is to say nomcas such a loading case,

V = ".LISTE_CHAR" (nomcas).
LENGTH (V) = nbmas = a number of super-meshs of the subjacent grid

for I = 1, nbmas
V (I):
/1 if super-mesh I is active for the loading nomcas
/0 if super-mesh I is not active for the loading nomcas

Code_Aster ®
Version
6.0

Titrate:
SD matr_elem and vect_elem

Date:

4 Examples

4.1 Context

model= affe_modele (grid: tail1e_bloc netted: 2.

affe: (all: “yes” phenomenon: “tnermic” modeling: “plane”));
chthl= affe_char_ther (model: model temp_impo: (node: (N3, N4) temp: 1.2)

flux_rep (all: “YES” flun: 5.));
chth2= affe_char_ther (model: model temp_impo: (node: N8 temp: 3.4)

source: (all: “yes” sour: 7.));

4.2
MATR_ELEM

matrel=calc_matr_elem (model: model cham_mater: chmat
charge: (chthl, chth2) option: “rigi_ther”);

SEGMENT IMPRESSION OF VALUES >MATREL .LISTE_RESU <
1 - >MATREL .MEOOl <>MATREL .MEOO2 <
3 - >MATREL .MEOO3 <

SEGMENT IMPRESSION OF VALUES >MATREL .REFE_RESU <
1 - >MODEL <>RIGI_THER <
3 - >NON_SOUS_STRUC <>CHMAT .MATE_CODE <
5 ***-*** *> <*

4.3
VECT_ELEM

vectel=calc_vect_elem (load: (chthl, chth2) option: “char_ther”);

SEGMENT IMPRESSION OF VALUES >VECTEL .LISTE_RESU <
1 - >VECTEL .VEOO1 <>VECTEL .VEOO2 <
3 - >VECTEL .VEOO3 <>VECTEL .VEOO4 <
5 ***-*** *> <> <*

SEGMENT IMPRESSION OF VALUES >VECTEL .REFE_RESU <
1 - >MODEL <>CHAR_THER <
3 - >NON_SOUS_STRUC <> <
5 ***-*** *> <*

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
D4.06.21 document

Structures of Data SD_FETI

Summary:

Description of the data-processing objects allowing to represent the decomposition in under-fields of one grid (cf operator of decomposition DEFI_PART_FETI [U4.23.05]). This partitioning is intended for to nourish a linear solvor multidomaine of the type FETI (cf solvor FETI [U4.50.01] [R6.01.03]).

---

1 General information

An object of the type SD_FETI is created by operator DEFI_PART_FETI [U4.23.05] on the total basis in order to represent the decomposition in under-fields of a grid. It must be provided to the solvor linear multi-fields FETI (key word SOLVEUR/PARTITION).
The size of this object is about nb_ma_tot +2 X nb_no_tot + 12 X nb_no_int + nb_sd (cf object .DIME for the notations).

This concept of partitioning FETI requires some explanations on the described entities. In summary:

- *The* ***meshs*** *of the ligrel of the* ***model*** *are divided into several under-****fields****. The latter thus consist of a whole of only one holding (connexity 1) of meshs listed in object .FETA. A mesh can thus belong only to one under-field: no the mesh divided of pieces or commune with several under-fields.*
- *The new borders generated by this cutting constitute the* ***interface****.* ***Nodes of interface*** *describing it are divided with at least two under-fields (****multiplicity geometrical*** *of .FETI and list .FETJ).*
- *The resolution of problem FETI is carried out on a vector of unknown factors,* ***Lagranges interfaces*** *(not to be confused with other Lagranges intervening in Code_Aster: conditions of Dirichlet, contacts…) (object .FETI), coinciding with these nodes of interface. With a node of interface corresponds as much of Lagranges than it is necessary to control continuity enters the under-fields. Lagrange east required for each binomial of under fields.*

Nodes
Sd 2

of interface
Under-field 1
12

23

Sd 3

12
Sd 1
13

Sd 2

1 Lagrange
3 Lagranges

Appear 1-a: Illustration of Lagranges of interfaces in 2D with 2 and 3 under-fields

Important remark on the interfaces:

For the moment, one highly disadvises the use of an interface of size N2 compared to dimension N of the problem. For example, in a 3D problem (n=3), an interface of the type segment enters a hexahedral under-field and a under-field made up of hulls.
In addition, it is to better avoid "polluting" these interfaces by loadings, conditions limits of generalized the Dirichlet type, the cracks, the zones of contact… ***Developments*** *FETI currently industrialized in the code,* ***do not ensure us of the good unfolding of things that when these interfaces are relatively virgin of any particular treatment****.*

---

2 Tree structures

SD_FETI (K19):: =record

".FDIM"
:

OJB

S V I

".FETA"
:

OJB

XD V I

".FETB"
:

OJB

XD V I

“.FETG”
:

OJB

XD V I

“.FETH”
:

OJB

S V I

“.FETI”
:

OJB

S V I

“.FETJ”
:

OJB

S V I

“.FREF”
:

OJB

S V K8

“.FLIN”
:

OJB

XD V K24

“.FLIM”
:

OJB

XD V I

“.FLII”
:

OJB

XD V I

% total objects temporary of work to all process FETI (cf remarks [§4])
“&&” //SDFETI (1: 17)/“.FINF”:
OJB

S V K24
“&FETI.INFO.STOCKAGE.FID”
:
OJB

S V I
“&FETI.INFO.STOCKAGE.FVAF”
:
OJB

S V I
“&FETI.INFO.STOCKAGE.FVAL”
:
OJB

S V I
“&FETI.INFO.STOCKAGE.FNBN”
:
OJB

S V I
“&FETI.INFO.CPU.FACN”
:
OJB

S V R
“&FETI.INFO.CPU.FACS”
:
OJB

S V R
“&FETI.INFO.CPU.ASSE”
:
OJB

S V R

SDFETI (1: 8)/“.MAILLE.NUMSD”:
OJB

S V I

LIGREL_DE_CHARGE (K19). “FEL1”:
OJB

S V K24
LIGREL_DE_CHARGE (K19). “FEL2”:
OJB

S V I
LIGREL_DE_CHARGE (K19). “FEL3”:
OJB

S V I
LIGREL_DE_CHARGE (K19). "FEL4":
OJB

S V I
LIGREL_DE_CHARGE (K19). "FEL5":
OJB

S V I

"&FETI.LISTE.SD.MPI"
:

OJB

S V I

"&FETI.LISTE.SD.MPIB"
:

OJB

S V I

3
Contents of objects JEVEUX

.FDIM*:*
S V I dim=5

Listing vector of the sizes characteristic of the cut out model.
FDIM (1) *= a number of nb_sd under-fields.*

FDIM (2) *= a number of Lagranges of interface nb_no_int.*

FDIM (3) *= a total number of meshs of the model nb_ma_tot.*

FDIM (4) *= a number of DDLs of interface nb_ddl_int.*

FDIM (5) *= a total number of nodes of the model nb_no_tot.*

.FETA:
XD V I LONG=nb_sd

Dispersed collection enumerating the list of the meshs by under-fields (meshs voluminal and associated meshs of skin to which apply a loading)
That is to say Vi=.FETA (I)
VI (J) = number of the jème mesh of the ième under-field.
The LONMAX of VI is equal to the number of meshs of the selected under-field.

.FETB:
XD V I LONG=nb_sd

Dispersed collection describing the nodes of the under-fields.
That is to say Vi=.FETB (I)
VI (2 (j-1) +1) = the number of the jème node of the ième under-field. This number is preceded by a sign if it is about a node of interface (VI (2 (j-1) +1) <0), of a sign + if not.
VI (2 (j-1) +2) = the number of DDLs until this node included. Thus a number of DDLs jème node is written:
If j=1 nb_ddl_j = VI (2),
If not nb_ddl_j = VI (2 (j-1) +2) - VI (2 (j-2) +2).
The LONMAX of VI is equal to twice the number of nodes of the selected under-field: nb_no_j = LONMAX/2.

.FETG:
XD V I LONG=nb_sd

Dispersed collection simulating the action of the operators of restriction/prediction.
That is to say Wi=.FETG (I)
Wi (2 (j-1) +1) = index of the jème Lagrange of interface of the ième under-field in the object .FETI. This number must be signed to check the continuity of the unknown field with the interface.
Can imports the convention of sign provided that its logic is respected everywhere*. One can for example make precede this index by a sign if this Lagrange with another under-field of number K > J (Wi (2 (j-1) is divided +1) <0), of one sign + if not. This convention is that retained by operator DEFI_PART_OPS [U4.23.05].*

Wi (2 (j-1) +2) = index of same Lagrange in the whole of the nodes (it is supposed coinciding with one of the nodes of interface of the grid) of the under-field chosen Vi='.FETB (I) '(thus VI (Wi (2 (j-1) +2)) < 0).

The LONMAX of Wi is equal to twice the number of Lagrange of interface of under selected field: nb_no_int_j = LONMAX/2.

.FETH:
S V I dim=nb_sd

Vector listing the numbers of DDLs per under-field (of the physical nodes and of late nodes).
That is to say X=.FETG
X (I) = a number of DDLs of the ième under-field.

.FETI:
*S V I dim= 4 * nb_no_int*

Vector describing Lagranges of interface.
That is to say Y=.FETI
Y (4 (j-1) +1) = number of the jème Lagrange of interface. It must thus be present in two negative ".FETB" (they exist K, L, m and N such as Y (4 (j-1) +1) =
- FETB (K) (2 (l-1) +1) = - FETB (m) (2 (n-1) +1))

Y (4 (j-1) +2) = its geometrical multiplicity mult_j.

Y (4 (j-1) +3) = the number of DDLs until this node included. Thus a number of DDLs is written:
If j=1 nb_ddl_j = Y (3),
If not nb_ddl_j = Y (4 (j-1) +3) - Y (4 (j-2) +3).

Y (4 (j-1) +4) = index, in object .FETJ, of the first of the mult_j under-fields comprising this Lagrange on one their interfaces. The other under-fields are with the continuation.

.FETJ:
nb_no_int
S V I dim= somme_mult =
mult_j
J =1

Vector describing the list of the under-fields containing the nodes of interface.
The access to this vector of storage indirect and is carried out via pointer .FETI (4 (J 1)+4).

.FREF:
S V K8 dim= 1 + nb_char (a number of loadings)

Listing vector of the general characteristics of partitioning for the possible ones checks (key word SOLVEUR/VERIF_SDFETI).
***FREF (1)** = name of the model,*
...

FREF (1+i) *= name of the ième loading.*

.FLIN:
XD V K24 LONG=nb_sd

For a given under-field, names of comprising LIGRELs of load of the meshs late with late nodes (condition of Dirichlet…) or not (nodal force). See also them .FEL1/3 objects with the §4.

.FLII:
XD V I LONG=nb_sd

For the ième under-field, that is to say Xi=.FLII (I) and J varying of 1 to LONMAX (.FLIN (I))
*Xi **(**2 (j-1) +1**)** = a number of late meshs of jème LIGREL of .FLIN (I),*
*Xi **(**2 (j-1) +2**)** = a number of these late meshs concerning this under-field (because one LIGREL of load can be with horse between several under-fields),*

.FLIM:
XD V I LONG=nb_sd

List absolute values of the late meshs concerning under-field I, in the order preceded by two objects preceding .FLIN and .FLIM. This object of
IN (I))
LONMAX (.FL
collection is thus length
X ((2 J -) 1 + 2)
I

J =1

---

4 Objects related

These temporary objects of the volatile base exist during a good share of a resolution FETI.

For the needs for monitoring

"&&" //SDFETI (1: 17)/".FINF"
S V K24
Character string to refine it
dim= 1
monitoring of FETI [U4.50.01].
"&FETI.INFO.STOCKAGE.FIDD"
S V I
Auxiliary vector for
dim= 2
filling of .FVAF and .FVAL.
V (1) = under-field running,
V (2) = a number of under-fields
"&FETI.INFO.STOCKAGE.FVAF"
S V I
Component counts of
dim= nb_sd+1
factorized local
"&FETI.INFO.STOCKAGE.FVAL"
S V I
Component counts of
dim= nb_sd+1
local matrices
"&FETI.INFO.STOCKAGE.FNBN"
S V I
Numbers of nodes of under
dim= nb_sd+1
fields
"&FETI.INFO.CPU.FACN"
S V R
Time (obtained via the routine
dim= nb_sd+1
UTTCPU, which is thus lower than
true spent time
(elapsed)) CPU + SYS of
local numerical factorizations.
"&FETI.INFO.CPU.FACS"
S V R
Time CPU + SYS of
dim= nb_sd+1
factorizations local symbolic systems.
"&FETI.INFO.CPU.ASSE"
S V R
Time CPU + SYS of
dim= nb_sd+1
local assemblies.
For the routines of assembly

SDFETI (1: 8)/".MAILLE.NUMSD"
S V I
Indicate the number of under-field
dim= nb_ma_tot
which a mesh belongs of
model. Initialized value with 999,
that makes it possible to test the membership
of all the meshs of the model to one

only under-field (only in
sequential mode. In parallel,
each processor reaches only
partial information and thus these
checks are invalid) and
to assemble the matrices and vectors
buildings.
For the routines of assembly in

presence of ligrel with meshs and/or
with late nodes
LIGREL_DE_CHARGE (K19). "FEL1"
S V K24
Names of projections of the ligrel
dim= nb_sd
on the under-fields concerned.
LIGREL_DE_CHARGE (K19). "FEL2" S V I
For the ième late mesh:
*dim= 2 * a number of V **(**2 (i-1) +1**)** = new number in*
late meshs of
ligrel
the projected ligrel,

*If V **(**2 (i-1) +2**)** >0 then number of*
under-field concerned, if not
*- V **(**2 (i-1) +2**)** = multiplicity of*
late mesh (DDL_IMPO on
the interface e.g.) and associated one
.FEL4.

LIGREL_DE_CHARGE (K19). "FEL3" S V I
For the ième late node
Only if late meshs with
*dim= 2 * a number of V **(**2 (i-1) +1**)** = new number in*
late nodes

late nodes of
ligrel
the projected ligrel,

*If V **(**2 (i-1) +2**)** >0 then number of*
under-field concerned, if not
- *V **(**2 (i-1) +2**) =** multiplicity of the node*
late (DDL_IMPO on the interface by
e.g.) and associated a .FEL5.
LIGREL_DE_CHARGE (K19). "FEL4" S V I
*V **(**1**)** = last index used of*

*dim= 3 * a number of vector*
late meshs
For the ième multiple late mesh
of interface
potential
*V **(**3 (i-1) +2**)** = new number in*

the projected ligrel,
*V **(**3 (i-1) +3**)** = number of under*
field concerned,
- *V **(**3 (i-1) +4**)** = old number.*
LIGREL_DE_CHARGE (K19). "FEL5" S V I
*V **(**1**)** = last index used of*
Only if late meshs with
*dim= 3 * a number of vector*
late nodes
late nodes
For the ième multiple late node
*of interface potentials V **(**3 (i-1) +2**)** = new number in*
the projected ligrel,
*V **(**3 (i-1) +3**)** = number of under*
field concerned,
- *V **(**3 (i-1) +4**)** = old number.*
For parallelism MPI

"&FETI.LISTE.SD.MPI"
S V I
Indicate in the loops on
dim= nb_sd+1
under-fields, if the processor
current is concerned with ième
under-field:
*V **(**i+1**)** = 1 the loop on it under*
field is carried out,
*V **(**i+1**)** = 0 it is jumped.*
*By convention of the loops, V **(1)***
relate to the total field and is worth
always 1.
*Into sequential, V **(I)** = 1 for any I.*
"&FETI.LISTE.SD.MPIB"
S V I
Object reverses precedent
dim= nb_sd

*V **(I)** = J under-field I is concerned with the processor J.*
*Into sequential, V **(I)** = 0 for any I.*

Notice on parallelism:

During a parallel execution, these temporary objects are declined by processor. However, according to distribution of load, each processor is concerned only by certain under-fields (cf objects "&FETI.LISTE…"). Therefore, put besides these the last two objects JEVEUX, others related objects contain only information relating to the under-fields which them interest.
For example, object SDFETI (1: 8)/".MAILLE.NUMSD" will comprise values initialized with 999 for the meshs of the under-fields concerning the other processors.

---

5 Examples

In the case test FETI002A, partionnement in four under-fields leads to SD SD_FETI following:

built named SD_FETI "SDFETI" following

====> IMPR_CO OF THE STRUCTURE OF DATA: SDFETI????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 8
==
IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:
--
SEGMENT IMPRESSION OF VALUES >SDFETI .FDIM <
*1 **-** 4 10 36 20 19*
--
IMPRESSION OF THE COLLECTION: SDFETI .FETA
SEGMENT IMPRESSION OF VALUES >SDFETI .FETA$$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: SDFETI
1 - >SD1 <>SD2 <>SD3 <>SD4 <
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETA< OC: 1
*1 **-** 1 2 3 4 5*
*6 **-** 6 25 27*

OBJECT IMPRESSION OF COLLECTION >SDFETI .FETA< OC: 2
1 - 19 20 21 22 23
6 - 24 29 31
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETA< OC: 3
1 - 13 14 15 16 17
6 - 18 30 32 33 34
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETA< OC: 4
1 - 7 8 9 10 11
6 - 12 26 28 35 36

IMPRESSION OF THE COLLECTION: SDFETI .FETB
SEGMENT IMPRESSION OF VALUES >SDFETI .FETB$$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: SDFETI
1 - >SD1 <>SD2 <>SD3 <>SD4 <
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETB< OC: 1
1 - -1 2 2 4 -3
6 - 6 -4 8 10 10
11 - 14 12 -17 14
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETB< OC: 2
1 - -1 2 -3 4 -8
6 - 6 9 8 11 10
11 - 15 12 -16 14
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETB< OC: 3
1 - -3 2 -5 4 7
6 - 6 -8 8 12 10
11 - -16 12 18 14
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETB< OC: 4
1 - -3 2 -4 4 -5
6 - 6 6 8 13 10
11 - -17 12 19 14

IMPRESSION OF THE COLLECTION: SDFETI .FETG
SEGMENT IMPRESSION OF VALUES >SDFETI .FETG$$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: SDFETI
1 - >SD1 <>SD2 <>SD3 <>SD4 <
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETG< OC: 1
1 - -1 1 -2 3 -3
6 - 3 -6 4 -10 7
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETG< OC: 2
1 - 1 1 2 2 -4
6 - 2 -8 3 -9 7
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETG< OC: 3
1 - 4 1 -5 1 -7
6 - 2 8 4 9 6
OBJECT IMPRESSION OF COLLECTION >SDFETI .FETG< OC: 4
1 - 3 1 5 1 6
6 - 2 7 3 10 6

SEGMENT IMPRESSION OF VALUES >SDFETI .FETH <
1 - 14 14 14 14

SEGMENT IMPRESSION OF VALUES >SDFETI .FETI <
1 - 1 2 2 1 3
6 - 4 4 3 3 4
11 - 6 5 3 4 8

16 - 7 3 4 10 9
21 - 4 2 12 11 5
26 - 2 14 13 8 2
31 - 16 15 16 2 18
36 - 17 17 2 20 19
--
SEGMENT IMPRESSION OF VALUES >SDFETI .FETJ <
1 - 1 2 1 2 1
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data SD_FETI

Date:
29/09/05
Author (S):
O. BOITEAU *Key*
:
D4.06.21-B Page
: 9/10

6 - 4 2 3 3 4
11 - 1 4 3 4 2
16 - 3 2 3 1 4
--
SEGMENT IMPRESSION OF VALUES >SDFETI ***.FREF*** *<*
1 - >MODM <>CH1 <

Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
Structure of Data SD_FETI

Date:
29/09/05
Author (S):
O. BOITEAU *Key*
:
D4.06.21-B Page
: 10/10

Intentionally white left page.
Data-processing handbook of Description
D4.06 booklet: Structures related to the finite elements
HT-66/05/003/A

Data-processing handbook of Description
D4.07 booklet: -
Document: D4.07.05

Structures of Data nume_ddl_gene,
vect_asse_gene, matr_asse_gene

Summary:

This document describes the structures of data associated with projection on a modal basis of type mode_meca or base_modale or established starting from a concept modele_gene resulting from sous_structuration, i.e.: nume_ddl_gene, vect_asse_gene and matr_asse_gene.

---

1 General information

A nume_ddl_gene is used to define the classification of the generalized co-ordinates of a system associated a base of projection of the mode_meca type or base_modale or starting from a concept modele_gene resulting from the sous_structuration.

*The projection of vectors or matrices, assembled according to this nume_ddl_gene on the modes of the base
of projection or on those of the modele_gene, resulting from the sous_structuration creates respectively
concepts of vectors or matrices generalized (vect_asse_gene resp. matr_asse_gene)*

The number of terms or equations of the vectors or matrices generalized depends on the number of modes taken into account in the concept nume_ddl_gene.

*The generalized matrices are stored in only one block according to a diagonal storage (in this case, as many terms as of modes) or full (in this case, its terms is arranged like one
symmetrical half-matrix in its object ".VALE").*

2 Tree structures

NUME_DDL (K14):: =record

“.NUME”: PROF_CHNO

“$VIDE”: STORAGE

PROF_CHNO (K19):: =record

“.NEQU”: OJB S V I

“.REFE”: OJB S V K24

“.DEEQ”: OJB S V I

“.LILI”: OJB S NR K24

“.NUEQ”: OJB S V I

“.PRNO”: OJB XC V I NAME ($.LILI) LONG (2)

“.ORIG”: OJB XC V I NAME ($.LILI) LONG (2)

STORAGE (K14):: =record

“.SLCS”:
STOC_LCIEL

STOC_LCIEL (K19):: =record

“.ABLO”: OJB S V I

“.ADIA”: OJB S V I

“.DESC”: OJB S V I

“.HCOL”: OJB S V I

“.IABL”: OJB S V I

“.REFE”: OJB S V K24

VECT_ASSE_GENE (K19):: =record

“.DESC”
:

OJB

S V I

“.REFE”
:

OJB

S V K24

“.VALE”
:

OJB

S V R

MATR_ASSE_GENE (K19):: =record

“.DESC”
:

OJB

S V I

“.REFE”
:

OJB

S V K24

“.VALE”
:

OJB

S V R

---

3 *PROF_CHNO*

".LILI" S
NR
K24
It is the pointer of names of ".PRNO". It contains the ligrels substructures and of connections "&SOUSSTR" and "CONNECTIONS". If generalized classification is associated a base of projection of the mode_meca type or base_modale, one considers that there is only one substructure and no connection.
Collection ".PRNO" thus contains 2 objects:
PRNO (1): numbers of the first modes of the substructures of name "&SOUSSTR"
PRNO (2): numbers of the first lagranges of the connections of name "CONNECTIONS"

".PRNO"
XC V I NAME ($.LILI) LONG (2)
This collection describes the numbers of the modes (resp. lagranges) carried by substructures (resp. connections).
It contains 2 vectors pointed respectively by the names "&SOUSSTR" and "CONNECTIONS". That is to say:
V = PRNO (1)
V (2 (isst-1) +1) = imod*
V (2 (isst-1) +2) = nb_mod*
.imod is the number of the first mode of the isstième substructure.

.nb_mod is the number of modes of the isstième substructure.
V = PRNO (2)
V (2 (ilia-1) +1) = ilag*
V (2 (ilia-1) +2) = nb_lag*
.ilag is the number of equation of the first lagrange of the iliaième substructure.
.nb_lag is the number of lagranges of the iliaième connection.

".ORIG"
XC V I NAME ($.LILI) LONG (2)
This collection describes the numbers of the substructures (resp. connections) carrying them modes (resp. lagranges).
It contains 2 vectors pointed respectively by the names "&SOUSSTR" and "CONNECTIONS". That is to say:
V = PRNO (1)
V (imod) = isst
.isst is the number of substructure of the imodième mode.
V = PRNO (2)
V (ilag) = ilia
.ilia is the number of connection of the ilagième lagrange.

".NEQU"
NEQU (1) a total number of equations.

".NUEQ"
S V I DIM = neq if neq is the number of equations of the PROF_CHNO
It is a vector containing the numbers of the equations.

".DEEQ"
*S V I DIM = 2*neq if neq is the number of equations of the PROF_CHNO*
If nueq is a number of equation (i.e addresses in object .VALE).
*V ((nueq-1) *2+1): imod*
*V ((nueq-1) *2+2): isst*
· If imod > 0 and isst > 0
nueq is the equation associated with the imodième mode with the isstième substructure.
· If imod = 1 and isst < 0
nueq is an equation of the isstième connection.

".REFE"
name of the generalized model (if it is necessary).

Data-processing handbook of Description
D4.07 booklet: -

4
STOC_LIGN_CIEL

.REFE
(1)
name of classification supporting this storage.

.DESC
(1)
a number of equations: neq
(2)
cut blocks of the matrix: t_bloc
(3)
a number of blocks necessary to the storage of the values
matrix: n_bloc (here n_bloc = 1)
(4)
maximum height of the columns of the matrix

.HCOL

S V I dim = neq
.HCOL (I)
height of the ième column

.ADIA

S V I dim = neq

.ADIA (I)
address diagonal term of the ième column in its block

.ABLO
S V I dim = n_bloc + 1 (here dim = 2)
.ABLO (1)
0
(K+1)
number of the last column of the block K (here K = 1).
note: a column can belong only to one block

.IABL

S V I dim = neq
.IABL (I)
number of the block K which contains the ième column of the matrix (here K = 1).

5
VECT_ASSE_GENE

.REFE (1)
name of the base of projection: mode_meca type or base_modale
(2)
name of the projected vector: cham_no_depl_R type

.DESC
(1)
= 1 bus vector
(2)
a number of vectors used in the base: n_vect
(3)
type of storage: = 1 so diagonal, = 2 so full

.VALE

S V I dim = n_vect
.VALE (I)
value of the ième stored term

6
MATR_ASSE_GENE

.REFE (1)
name of the base of projection: mode_meca type or base_modale
(2)
name of the projected matrix: matr_asse_depl_R type

.DESC
(1)
= 2 bus stamps
(2)
a number of vectors used in the base: n_vect
(3)
type of storage: = 1 so diagonal, = 2 so full

.VALE
S V I dim = n_termes, n_termes is worth n_vect if stokage diagonal
and n_vect (n_vect +1) /2 if full storage*
.VALE (I)
value of the ième stored term

Intentionally white left page.
Data-processing handbook of Description
D4.07 booklet: -
HT-66/03/007/A

Data-processing handbook of Description
D4.08 booklet:
D4.08.01 document
Structures of Data MACR_ELEM_STAT

1
The SD in some lines
SD MACR_ELEM_STAT represents a macro element in static under-structuring.
It contains following information:

- *definition of the external nodes of the macronutrient,*
- *references "upstream" to the GRID, MODEL, CHAM_MATER and CARA_ELEM,*
- *references "upstream" to the CHAR_MECA of conditions kinematics,*
- *a matrix of condensed rigidity,*
- *a matrix of condensed mass,*
- *condensed loadings.*

A macro element is to some extent a finite element of which the mesh support has a number unspecified of nodes.
Moreover these nodes all are not of the "physical" nodes: owing to the fact that one can impose conditions kinematics (dualized) on a MACR_ELEM_STAT, certain nodes of the macro element are of type "Lagrange".
A macro element "connait" only some options (within the meaning of the traditional finite elements): RIGI_MECA, MASS_MECA, CHAR_MECA.

2

Tree structure

MACR_ELEM_STAT (K8)
:: = record
% description geometrical and topological:
“.CONX”
: OJB
S V I
“.DESM”
: OJB
S V I
“.LINO”
: OJB
S V I
“.REFM”
: OJB
S V K8
“.VARM”
: OJB
S V R
% condensed rigidity:

“(6) .NUME”
: NUME_EQUA
“(6) .SLCS”
: STOC_LCIEL
“.RIGIMECA”
: MATR_ASSE
“.KP_EE”
: OJB
S V R
“.PHI_IE”: OJB
X V R
NAKED DISPER IDIOTS
% masses condensed:

“.MASSMECA”
: MATR_ASSE
“.MP_EE”
: OJB
S V R
% description of the loadings:

“.LICA”
: OJB

X V R
NO DISPER IDIOTS
“.LICH”
: OJB
X V K8
NO CONTIG IDIOTS
Data-processing handbook of Description
D4.08 booklet:
Index A

3
Contents of the objects

".DESM": OJB S V I LONG=10

DESM (1): (vacuum)
DESM (2): a number of external nodes grid (nbnoe
)
DESM (3): a number of nodes intern grid (nbnoi
)
(node grid = physical node or node lagrange)
DESM (4): number external ddls. total
(nddle)
DESM (5): ddls interns numbers. total
(nddli)
DESM (6): a number of CHAR_MECA kinematics.
(nbchar)
DESM (7): a number of definite loading cases.
(nbcas)
DESM (8): number external lagranges
(nlage)
DESM (9): lagranges relations numbers
(nlagl)
DESM (10): lagranges interns numbers
(nlagi)

".REFM": OJB S V K8 LONG=8+nbchar

REFM (1): model (nomo)
REFM (2): grid (noma)
REFM (3): cham_mater (to subdue)

REFM (4): cara_elem (carele)
REFM (5): nume_ddl
REFM (6): “OUI_RIGI”/“NON_RIGI”
REFM (7): “OUI_MASS”/“NON_MASS”
REFM (8): “OUI_AMOR”/“NON_AMOR”
REFM (8 +1): char_cinema_1
REFM (8 +2): char_cinema_2
REFM (8 +nchar): char_cinema_N
“.LINO”: OJB S V I

LINO contains the list of the physical nodes
external different.
The real number of these nodes (LONUTI) is also in DESM (2).
The order of the nodes in .LINO is that of appearance in .CONX
“.CONX”: OJB S V I
----
nbnoe = lonuti (LINO)
nbnoet = nbnoe + nlage + nlagl
It is the total number of external nodes.
CONX is dimensioned with 3*nbnoet
The external nodes are number in the order of appearance in initial classification
(condensed matrix).
CONX (1, inoe): ili = number of the ligrel of .LILI (nume_ddl)
containing the external node inoe
CONX (2, inoe): ino = number of inoe in the ligrel ili
CONX (3, inoe): 0 if physical node
-1 if this node of lagrange is a node “before”
-2 if this node of lagrange is a node “after”
“.VARM”: OJB S V R
LENGTH = 2

VARM (1): cut blocks matrices
Data-processing handbook of Description
D4.08 booklet:
Index A

Code_Aster ®

Version
4
Titrate:
Structure of Data MACR_ELEM_STAT
Date:

VARM (2): moment calculation rigidity, mass,…
“.PHI_IE”: OJB XD V R LONG (cste) NAKED ()

PHI_IE is the matrix: PHI_IE = K_II ** (- 1) *K_IE
or K_II: submatrix of the ddls interns.
or K_EE: submatrix of the external ddls.
or K_IE: submatrix of the couplings intern/external.
PHI_IE (iddle) is a vector length nddli.
PHI_IE (iddle) (iddli) is the coef. (I, E) of matrix PHI_IE
“.KP_EE”: OJB S V R

Stamp rigidity condensee.
KP_EE is the matrix: KP_EE = K_EE - K_EI*PHI_IE
This matrix is stored “symmetrical” by columns:
KP_EE (I, J) = KP_EE (j* (j-1) /2 +i) for j>=i
KP_EE is a vector length nddle* (nddle+1) /2
“.MP_EE”: OJB S V R

Stamp of condensed mass.
MP_EE is the matrix: MP_EE = M_EE + PHI_EI*M_II*PHI_IE
- M_EI*PHI_IE - PHI_EI*M_EI
This matrix is stored “symmetrical” like .KP_EE
“.LICH”: OJB XC V K8 LONG (cste) NO ()

Collection dimensioned with the number of cas_de_charge.
It is named by the names of cas_de_charge nomcas.
LICH (nomcas) is of dimension n_char_max+1
LICH (nomcas) (1) =/“NON_SUIV” nonfollowing loading
/“OUI_SUIV” following loading
LICH (nomcas) (1<i<=n_char_max+1) = name of the load i-1
“.LICA”: OJB XD V R LONG (cste) NO ()

Collection dimensioned with the number of cas_de_charge.
It is named by the names of cas_de_charge nomcas.
LICA (nomcas) is of dimension nddlt= nddli+ nddle
LICA (nomcas) (1<=i<=nddli) = (k_II ** - 1) *F_I
LICA (nomcas) (nddli+1<=i<=nddle) = FP_J

or FP_E = F_E - K_EI* (K_II ** - 1) *F_I
Data-processing handbook of Description
D4.08 booklet:
Index A

4
Examples: macr_elem_stat s_1 and s_123 of the test SSLP100B
4.1
Structure of Data: S_1

--
SEGMENT IMPRESSION OF VALUES >S_1 .CONX <
1 - 1 1 0 1 3
6 - 0 1 4 0 1
11 - 7 0 1 9 0
16 - 1 6 0 1 10
21 - 0 1 12 0
--
SEGMENT IMPRESSION OF VALUES >S_1 .DESM <
1 - 0 8 4 16 12
6 - 1 2 0 0 4
--
SEGMENT IMPRESSION OF VALUES >S_1 .KP_EE <
1 - 6.41345E+00 1.76677E+00 9.20402E+00 -1.81735E+00 1.25037E+00
6 - 9.57550E+00 1.00328E+00 -4.07100E+00 -1.39527E+00 6.04196E+00
...
136 - 1.03411E+01
--
IMPRESSION OF THE COLLECTION: S_1 .LICA
SEGMENT IMPRESSION OF VALUES >S_1 .LICA $$NOM <

>>>>> REPERTORY OF NAMES OF THE COLLECTION: S_1 .LICA
1 - >CHF1 <>CHF2 <
OBJECT IMPRESSION OF COLLECTION >S_1 .LICA < OC: 1
1 - 3.53553E+00 1.46447E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - 0.00000E+00 0.00000E+00 1.91342E+00 3.80604E-01 1.62212E+00
16 - 1.08386E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
21 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
26 - 0.00000E+00 0.00000E+00 0.00000E+00 2.39508E-01 9.92075E-02
31 - 2.98282E-02 1.23552E-02 -4.71006E-19 -3.57376E-19 7.14525E-03
36 - 2.95966E-03 -1.23096E-19 -1.73472E-18 7.14525E-03 2.95966E-03
41 - 2.64693E+00 1.18501E+00 2.70958E+00 1.03373E+00 8.48546E-01
46 - -3.31522E-01 1.30381E-01 -6.22318E-02 4.81890E-02 1.36198E-01
51 - 3.65592E-01 8.34435E-01 6.90624E-18 -4.58919E-18 -2.12901E-18
56 - -1.19111E-17
OBJECT IMPRESSION OF COLLECTION >S_1 .LICA < OC: 2
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - -2.00000E+01 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
16 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
21 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
26 - 0.00000E+00 0.00000E+00 0.00000E+00 -8.62497E-18 1.49690E-17
31 - 1.58841E-17 -6.06819E-17 3.01596E-01 -1.61168E+00 -1.00331E-01
36 - -8.43526E-02 -3.83034E-16 8.60423E-16 -1.00331E-01 -8.43526E-02
41 - -1.78149E-16 -2.07069E-16 1.49183E-16 6.01817E-18 1.80546E-15
46 - 2.83180E-15 -4.67445E+00 -7.31983E+00 3.49452E+00 -1.72624E+00
51 - -3.63569E-15 7.98788E-16 2.19705E+00 -4.60236E+00 3.50220E+00
56 - -2.55199E+00

IMPRESSION OF THE COLLECTION: S_1 .LICH
SEGMENT IMPRESSION OF VALUES >S_1 .LICH $$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: S_1 .LICH
1 - >CHF1 <>CHF2 <
OBJECT IMPRESSION OF COLLECTION CONTIGUE>S_1 .LICH < OC: 1
1 - >OUI_SUIV<>CHBL_1 <>CHF1_1 <> <> <> <> <
8 - > <> <> <
OBJECT IMPRESSION OF COLLECTION CONTIGUE>S_1 .LICH < OC: 2
1 - >NON_SUIV<>CHF2_1 <> <> <> <> <> <
8 - > <> <> <

SEGMENT IMPRESSION OF VALUES >S_1 .LINO <
1 - 1 3 4 7 9
6 - 6 10 12

IMPRESSION OF THE COLLECTION: S_1 .PHI_IE
OBJECT IMPRESSION OF COLLECTION >S_1 .PHI_IE < OC: 1

1 - -2.11984E-01 1.09032E-02 -9.57179E-02 -1.05534E-01 1.57605E-16
6 - 6.16106E-17 -2.98079E-02 -8.67300E-03 2.25984E-16 5.55112E-17
11 - -2.98079E-02 -8.67300E-03
OBJECT IMPRESSION OF COLLECTION >S_1 .PHI_IE < OC: 2
1 - -4.75162E-02 -4.34567E-01 -1.59188E-01 -1.36495E-01 1.80213E-16
6 - 7.87647E-17 -4.54996E-02 -1.49122E-02 2.31892E-16 1.38778E-16
11 - -4.54996E-02 -1.49122E-02
...
OBJECT IMPRESSION OF COLLECTION >S_1 .PHI_IE < OC: 16
1 - 1.88451E-17 2.27704E-17 1.11285E-16 -8.18965E-18 2.12371E-01
6 - -1.27599E-01 4.83504E-03 -1.52778E-01 -2.56578E-17 8.11851E-16
11 - 4.83504E-03 -1.52778E-01

SEGMENT IMPRESSION OF VALUES >S_1 .REFM <
1 - >MO_1 <>MA <>CHMAT <> <>S_1 <>OUI_RIGI<>NON_MASS<
8 - >NON_AMOR<>CHBL_1 <

SEGMENT IMPRESSION OF VALUES >S_1 .VARM <
1 - 4.00000E+02 0.00000E+00

4.2
Structure of Data: S_123

SEGMENT IMPRESSION OF VALUES >S_123 .CONX <

1 - 1 1 0 1 3
6 - 0 1 4 0 1
11 - 7 0 3 5 -1
16 - 1 14 0 3 3
21 - -1 3 6 -2 1
26 - 15 0 3 4 -2
31 - 3 7 -1 1 16
36 - 0 3 1 -1 3
41 - 8 -2 1 13 0
46 - 3 2 -2

SEGMENT IMPRESSION OF VALUES >S_123 .DESM <
1 - 0 8 8 24 16
6 - 1 2 0 8 0

SEGMENT IMPRESSION OF VALUES >S_123 .KP_EE <
1 - 6.03407E+00 1.82427E+00 6.67688E+00 -3.21403E+00 5.12297E-01
6 - 1.37237E+01 -9.43232E-01 -9.60998E-01 1.88753E+00 1.16316E+01
11 - -3.63182E-01 -1.33685E-01 -5.94068E+00 1.84587E-01 1.62314E+01
16 - 6.81010E-02 -1.62885E-01 -2.90313E-01 -5.64955E-01 -1.06762E-01
...
296 - 3.83854E+01 0.00000E+00 1.91927E+01 1.91927E+01 -3.83854E+01

IMPRESSION OF THE COLLECTION: S_123 .LICA
SEGMENT IMPRESSION OF VALUES >S_123 .LICA $$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: S_123 .LICA
1 - >CHF1 <>CHF2 <
OBJECT IMPRESSION OF COLLECTION >S_123 .LICA < OC: 1
1 - 3.74332E+00 3.74332E+00 1.20003E+00 1.20003E+00 1.84387E-01
6 - 1.84386E-01 5.99949E-18 -1.02727E-17 -1.45753E-06 5.29385E+00
11 - 9.28668E-08 1.69709E+00 1.85919E-08 2.60762E-01 1.15062E-17
16 - -3.02164E-18 2.64693E+00 1.18501E+00 8.48546E-01 -3.31522E-01
21 - 1.30381E-01 -6.22318E-02 6.90624E-18 -4.58919E-18 0.00000E+00
26 - -1.36198E-01 4.81890E-02 0.00000E+00 0.00000E+00 -8.34435E-01
31 - 3.65592E-01 0.00000E+00 0.00000E+00 1.19111E-17 -2.12901E-18
36 - 0.00000E+00 0.00000E+00 -1.03373E+00 2.70958E+00 0.00000E+00
41 - 3.48963E-01 4.82674E-01 1.78047E-01 1.80297E-01 1.25697E-01
46 - 1.10119E-01 8.54484E-02 7.51949E-02 9.45476E-02 5.88056E-01
51 - 1.59078E-03 2.53388E-01 -1.10150E-02 1.66748E-01 -7.25041E-03
56 - 1.13592E-01 2.98827E+00 3.63266E+00 6.07038E-01 4.45522E-01
61 - 4.78925E-01 -1.95160E-01 -6.67765E-02 -6.60203E-02 0.00000E+00
66 - -4.76652E-01 2.00653E-01 0.00000E+00 0.00000E+00 -1.14207E-01
71 - 7.44275E-01 0.00000E+00 0.00000E+00 5.34686E-04 -9.39012E-02

76 - 0.00000E+00 0.00000E+00 4.55648E-01 4.68170E+00 0.00000E+00
OBJECT IMPRESSION OF COLLECTION >S_123 .LICA < OC: 2
1 - -1.07663E-15 5.59669E-15 -7.53012E-15 1.96373E-14 4.41365E+00
6 - -1.02260E+01 7.74962E+00 -1.45510E+00 1.19912E-15 5.76725E-15
11 - 2.95592E-15 1.41995E-14 -5.80631E-06 -5.81234E+00 -8.50025E-06

---

16 - 6.29451E+00 -1.78149E-16 -2.07069E-16 1.80546E-15 2.83180E-15
21 - -4.67445E+00 -7.31983E+00 2.19705E+00 -4.60236E+00 0.00000E+00
26 - -9.19119E-01 -8.49975E+00 0.00000E+00 0.00000E+00 2.97340E-15
31 - 1.43800E-14 0.00000E+00 0.00000E+00 -4.24742E+00 1.09688E+00
36 - 0.00000E+00 0.00000E+00 9.32445E-16 4.25016E-15 0.00000E+00
41 - -4.98962E-03 -2.10948E-02 -4.25571E-03 -2.81023E-02 3.09009E-01
46 - -3.89619E-01 6.89189E-01 -4.27942E-01 1.04180E-02 -6.72636E-03
51 - 2.73092E-02 -3.78207E-02 2.65220E-02 -8.27550E-02 9.56837E-02
56 - 2.70699E-01 -2.90996E-01 -5.83448E-01 1.39967E-01 -1.07233E+00
61 - -4.62267E+00 -8.15812E+00 1.98317E+00 -5.44812E+00 0.00000E+00
66 - -8.07994E-01 -8.88717E+00 0.00000E+00 0.00000E+00 3.02188E-02
71 - -3.01191E-02 0.00000E+00 0.00000E+00 -4.37916E+00 1.39317E+00
76 - 0.00000E+00 0.00000E+00 2.79612E-03 -1.12013E-01 0.00000E+00

IMPRESSION OF THE COLLECTION: S_123 .LICH
SEGMENT IMPRESSION OF VALUES >S_123 .LICH $$NOM <
>>>>> REPERTORY OF NAMES OF THE COLLECTION: S_123 .LICH
1 - >CHF1 <>CHF2 <
OBJECT IMPRESSION OF COLLECTION CONTIGUE>S_123 .LICH < OC: 1
1 - >OUI_SUIV<>CHBL_123<> <> <> <> <> <
8 - > <> <> <

OBJECT IMPRESSION OF COLLECTION CONTIGUE>S_123 .LICH < OC: 2
1 - >NON_SUIV<>CHBL_123<> <> <> <> <> <
8 - > <> <> <

SEGMENT IMPRESSION OF VALUES >S_123 .LINO <
1 - 1 3 4 7 14
6 - 15 16 13

IMPRESSION OF THE COLLECTION: S_123 .PHI_IE
OBJECT IMPRESSION OF COLLECTION >S_123 .PHI_IE < OC: 1
1 - -1.43678E-01 3.74403E-02 -3.17532E-02 -4.50471E-02 -3.42354E-02
6 - -6.07434E-02 -2.87693E-02 -3.73962E-02 -4.32720E-02 2.30991E-02
11 - -1.52709E-02 2.34137E-02 -1.43768E-03 1.53165E-02 3.39843E-03
16 - 1.24711E-02
OBJECT IMPRESSION OF COLLECTION >S_123 .PHI_IE < OC: 2
1 - -3.47164E-02 -3.47118E-01 -7.64680E-02 -2.30246E-01 -9.49132E-02
6 - -1.30318E-01 -7.28149E-02 -8.29774E-02 -1.20224E-02 -8.98287E-02
11 - 2.93609E-02 -6.87771E-02 1.53364E-02 -6.31448E-02 1.17360E-02
16 - -4.03116E-02
...
OBJECT IMPRESSION OF COLLECTION >S_123 .PHI_IE < OC: 24
1 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
16 - 0.00000E+00

SEGMENT IMPRESSION OF VALUES >S_123 .REFM <
1 - >MO_123 <>MA_123 <> <> <>S_123 <>OUI_RIGI<>NON_MASS<
8 - >NON_AMOR<>CHBL_123<

SEGMENT IMPRESSION OF VALUES >S_123 .VARM <
1 - 4.00000E+02 0.00000E+00

28/01/1999
Author (S):
J. PELLET
Key:
D4.08.01
Page: 8/8
Intentionally white left page.
Data-processing handbook of Description
D4.08 booklet:
Index A

Code_Aster ®
Version
3.0
Titrate:
Structures of data FOND_FISS
Date:
28/04/98
Author (S):
X. DESROCHES
Key:
D4.10.01-A
Page:
1/4
Organization (S): EDF/IMA/MMN
Data-processing handbook of Description
D4.10 booklet: Structures of data for the breaking process
D4.10.01 document
Structures of data FOND_FISS
Summary:
This document describes the structure of data fond_fiss produced by operator DEFI_FOND_FISS [U4.63.01] and used by the operators of breaking process CALC_THETA [U4.63.02] and CALC_G_LOCAL [U4.63.04].
Handbook of Inforamtique Description
Booklet DD410: Strictures of data for the breaking process
HI-75/96/023 Ind A

Code_Aster ®
Version
3.0
Titrate:
Structures of data FOND_FISS
Date:

1 General information

An object of the fond_fiss type describes a bottom of crack of a grid 3D or 2D (in this case, the bottom of crack
is tiny room to a node). This concept is obligatorily produced by operator DEFI_FOND_FISS [U4.63.01].

2
Relationships to the other structures of data

A concept fond_fiss is defined on a grid, via entities NODE, GROUP_NO, MESH,
GROUP_MA describing the crack.

3
Tree structure of the structure of data fond_fiss

fond_fiss (K8)
:: =
record
'.fond
.noeu '
:
S
V
K8

'.levresup
.mail':
S
V
K8

'.levreinf
.mail':
S
V
K8

'.normale
'
:
S

V
R
'.dtan_origine
' :
S
V
R
“.dtan_extremite”:
S
V
R

4
Contents of basic objects JEVEUX

'.fond
.noeu'
:
vector (K8) containing the list of the ordered nodes of the bottom of
fissure
'.levresup
.mail'
:
vector (K8) containing the list of the meshs of the lip
higher of the crack
'.levreinf
.mail'
:
vector (K8) containing the list of the meshs of the lower lip
crack
'.normale
'
:
vector of 3 real containing the components (nx, ny, nz) of
normal in the plan of the lips (case of a crack planes)
(see convention of sign in [U4.63.01 §3.4])
'.dtan_origine
'
:
vector of 3 real containing the components of the tangent with
the structure in the beginning of the bottom of crack, in the plan of
lips
(see convention of sign in [U4.63.01 §3.5])
'.dtan_extremite
'

:
even thing that .dtan_origine at the end of the bottom of
fissure
Handbook of Inforamtique Description
Booklet DD410: Strictures of data for the breaking process
HI-75/96/023 Ind A

Code_Aster ®
Version
3.0
Titrate:
Structures of data FOND_FISS
Date:
28/04/98
Author (S):
X. DESROCHES
Key:
D4.10.01-A
Page:
3/4

5 Example

%
% DEFINITION Of a SEMI-ELLIPTIC CRACK
%
BEGINNING ();
MA = LIRE_MAILLAGE ();
FISS1 = DEFI_FOND_FISS (GRID: MA,
MELTS
: (GROUP_NO: FONFIS)
LEVRE_SUP: (GROUP_MA: GRPRES)
dtan_origine:
(0. , 1. , 0.)
dtan_extremite
:
(1. , 0. , 0.)
);
IMPR_CO (CO: FISS1);
END ();
====> IMPR_CO OF THE STRUCTURE OF DATA: FISS1????????????????
ATTRIBUTE: F CONTENTS: T BASE: >G<
A NUMBER Of OBJECTS (OR COLLECTIONS) FIND: 4
===
=

IMPRESSION OF THE CONTENTS OF THE OBJECTS FIND:

-
SEGMENT IMPRESSION OF VALUES >FISS1
.DTAN_EXTREMITE
<
>>>>>
1 - 1.00000E+00 0.00000E+00 0.00000E+00

-
SEGMENT IMPRESSION OF VALUES >FISS1
.DTAN_ORIGINE
<
>>>>>
1 - 0.00000E+00 1.00000E+00 0.00000E+00

-
SEGMENT IMPRESSION OF VALUES >FISS1
.FOND
.NOEU
<
>>>>>
1 - >NO1099
<>NO1100
<>NO1109
<>NO1110
<>NO1119
<>NO1120
<>NO1129<
8 - >NO1130
<>NO1139
<>NO1140
<>NO1149
<>NO1150
<>NO1159
<>NO1160<
15 - >NO1169 <>NO1170
<>NO1179
<>NO1180
<>NO1189
<>NO1190
<>NO1199<
22 - >NO1200 <>NO1209

<>NO1210
<>NO1219
<>NO1220
<>NO1229
<>NO1230<
29 - >NO1239 <>NO1240 <>NO1249 <>NO1250 <>NO1259 <

-
SEGMENT IMPRESSION OF VALUES >FISS1
.LEVRESUP .MAIL
<
>>>>>
1 - >MA7
<>MA8
<>MA9
<>MA10 <>MA11 <>MA12 <>MA25 <
8 - >MA26 <>MA27 <>MA28 <>MA29 <>MA30 <>MA31 <>MA32 <
15 - >MA33 <>MA34 <>MA35 <>MA36 <>MA61 <>MA62 <>MA63 <
22 - >MA64 <>MA65 <>MA66 <>MA67 <>MA68 <>MA69 <>MA70 <
29 - >MA71 <>MA72 <>MA73 <>MA74 <>MA75 <>MA76 <>MA77 <
36 - >MA78 <>MA79 <>MA80 <>MA81 <>MA82 <>MA83 <>MA84 <
43 - >MA96 <>MA100
<>MA102
<>MA104
<>MA106
<>MA108
<>MA110 <
50 - >MA112 <>MA114
<>MA116
<>MA118
<>MA120
<>MA122
<>MA124 <
57 - >MA126 <>MA128
<>MA130
<>MA132
<>MA134
<>MA136
<>MA138 <
64 - >MA140 <>MA142
<>MA144
<>MA146
<>MA148

<>MA150
<>MA152 <
71 - >MA154 <>MA156
<>MA158
<>MA160
<>MA162
<
====> FINE IMPR_CO OF STRUCTURE OF DATA: FISS1????????????????

Handbook of Inforamtique Description
Booklet DD410: Strictures of data for the breaking process
HI-75/96/023 Ind A

Code_Aster ®
Version
3.0
Titrate:
Structures of data FOND_FISS
Date:
28/04/98
Author (S):
X. DESROCHES
Key:
D4.10.01-A
Page:
4/4

Intentionally white left page.

Handbook of Inforamtique Description
Booklet DD410: Strictures of data for the breaking process
HI-75/96/023 Ind A

Code_Aster ®
Version
8.1

Titrate:
To introduce a new order

Date:
01/12/05
Author (S):
C.** Key **DURAND
:
D5.01.01-C Page
: 1/16

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D5.01 booklet: -
D5.01.01 document

To introduce a new order

Summary:

This document describes the method to introduce a new order (operator or procedure) into Code_Aster. It describes the drafting with the format "python" of the catalogue of the order and the routine
FORTRAN associated.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
To introduce a new order

Date:
01/12/05
Author (S):
C. Key DURAND
:
D5.01.01-C Page
: 2/16

Count

matters

---

1 Introduction

To introduce a new order into Code_Aster, it is necessary:

·
to write the catalogue associated with this order [§3],
·
to write associated routine FORTRAN OPxxxx [§9],
·
to follow the plan of development [A2.01.00].

We will speak here only about the first two points.

2
Vision user of an order

Let us take as example the order AFFE_MATERIAU which makes it possible to affect on a grid of material characteristics. Here a possible use of this order in the file of orders provided by the user of Code_Aster:

cham = AFFE_MATERIAU (MAILLAGE=mail,
AFFE=_F (
ALL
=
“YES”,
MATER
=
steel

))

During the use of an order, it appears:

.
the name “user” of the concept produced by the order: cham
.
the name of the order: AFFE_MATERIAU
.
key words factors: AFFE
.
single-ended spanner words: ALL, MATER, GRID
.
names “users” of concepts arguments: steel, email

.

values of the simple type (whole, real, text,…) only or in list: "YES"

From the user point of view, by writing a name on the left sign "=" of the order, one affects it name with the result of the order.

This "name user" is affected a produced concept (or structure of data) calculated by the operator and whose type is given by the supervisor. The type of the produced concept is defined in catalogue order [§3] in coherence with the catalogue accas.capy, stored with same place that catalogues of orders, which inventories all the types of concepts created and used by the code.

For example cham is the name user of the result of the order and with this name is associated it concept of the cham_mater type (see following page).
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

3
Drafting of the catalogue of order

To introduce a new order, it is necessary to create an associated catalogue in which will be indicated:

.

the name of the order,
.

its description in a few words,
.

the category of classification by families of the orders for posting in EFICAS,
·
the nature of the order: operator (production of concept), procedure (not of concept product), macro-order,
·
number xxxx of routine FORTRAN associated with this order [§4].
·
for the produced concept:
-
rules of determination of the type of the concept [§9],
-
the possibility of re-use (réentrant character).
·
for the key words [§5], [§7]:
-
if their presence is optional or obligatory, if they are excluded between them,…
-
the type of the argument,
-
the number of awaited arguments of this type,
-
the default value (if there is one of them),
-
the list of the acceptable values (possibly),
-
the beach of the acceptable values (possibly), if one awaits an entirety or a reality,
·
for the key words factors [§6]:
-
if their presence is optional or obligatory,
-
the minimum and maximum number of possible occurrences,
·
blocks: logical regrouping of key words when conditions on other key words are satisfied [§8].

Note:

One will not speak in this document about the introduction about a new macro-order [D5.01.02].

The language used to write this catalogue is the language interpreted Python: the comments are written behind character "#", one sees key words (identifying follow-ups of the character "="), brackets, of the commas to separate the key words…

*Let us take again the example of order AFFE_MATERIAU, the catalogue - i.e. the description of the order provided by its **developer** - associated is:*

*AFFE_MATERIAU=OPER (**nom=** " AFFE_MATERIAU ", **op=6**, sd_prod=cham_mater,*
***fr=** " Assignment of material characteristics to a grid ",*
*reentrant=', **UIinfo=** {“groups”: (“Modeling”,)},*

GRID =SIMP (statut=' o', typ=maillage),
MODEL =SIMP (statut=' f', typ=modele),

*AFFE =FACT (statut=' o', min=1, max=' ** ',*
***regles=** (UN_PARMI (“ALL”, “GROUP_MA”, “MESH”, “GROUP_NO”, “NODE”),),*
ALL =SIMP (statut=' f', typ=' TXM', into= (“YES”,)),
*GROUP_MA =SIMP (statut=' f', typ=grma, max=' ** '),*
*NET =SIMP (statut=' f', typ=ma, max=' ** '),*
*GROUP_NO =SIMP (statut=' f', typ=grno, max=' ** '),*
*NODE =SIMP (statut=' f', typ=no, max=' ** '),*
MATER =SIMP (statut=' o', typ=mater),

TEMP_REF =SIMP (statut=' f', typ=' R', defaut= 0.E+0),
),
);

Some remarks on this catalogue of order:

.

*it appears **reserved key words** of the catalogue, written in fat in the example, that us will try not to confuse with the key words of the order,*

·

*the type of the produced concept is specified behind the key word reserved **sd_prod**,*

In the description of the order, the reader of catalogue recognizes the reserved key words following, their significance specifies will be given throughout the document.

OPER/PROC/MACRO
To specify the type of order (production of one, zero even several concepts in the case of macros)
name
To indicate the name seen by the user to indicate the order
COp
To specify the number of high level routine FORTRAN associated order
sd_prod
To define the type of produced concept
rules
To define the logical rules of pairing or exclusion of key words
UN_PARMI/….
To define a list of key words among which the data must to find exactly once.
Fr
To describe in a sentence (in French) the role of the order, it is it contents of the bubble of assistance posted by EFICAS
UIinfo
Useful only for postings in EFICAS, to specify the family of classification of the order: Postprocessings, Modeling…
reentrant
To specify if the order creates a new concept (value "), modify an existing concept (value "O"), or potentially both (value "F")
SIMP
To specify a simple key word of the order
FACT
To specify a key word ratio control.
BLOCK
To define a block of key words of which the appearance is subjected to one "condition".
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

One can break up the writing of the catalogue of order according to following stages' [D1.02.01 §1.2]:

.
***To specify the type of the concept produced** (when there exists, i.e. for an order of type OPER):*

*To specify the type of the produced concept of an operator, should be used the reserved key word **sd_prod** (structure of produced data). For example, the assignment sd_prod=cham_mater indicate that cham_mater is the type of the produced concept of operator AFFE_MATERIAU.*

In the case of a procedure, it does not have there a produced concept (and thus not of key word sd_prod in the catalogue). For example CALC_G_THETA_T is an order of which the type of produced concept is TABL_CALC_G_TH, whereas IMPR_RESU is a procedure without produced concept:

```
CALC_G_THETA_T
=
OPER (
nom= " CALC_G_THETA_T ", op=53,
sd_prod=tabl_calc_g_th,
reentrant=',
...
```

```
IMPR_RESU = PROC (nom= " IMPR_RESU ", op=39,…
```

If the type of the produced concept depends on the arguments of the operator, it will be consulted [§9.1].

If the produced concept can be a re-used concept and nouveau riche, one will indicate it by informing it reserved key word ***reentrant*** *[§9.2].*

.

To define the name of the order:

It is written on the left sign "=" in the catalogue, on the right in the command file of the user. The rule of use is to constitute monograms using components of four characters separated by underlined white (if possible). The final choice of the name is given by the Project Leader [A2.01.02].

Generally the prefix indicates the action, the suffix the treated concept (for example AFFE_MATERIAU). Let us note some prefixes frequently employed:

AFFE:
assignments on the grid or the model,
CHALLENGE:
definitions of objects which are not fields,
CALC:
orders calling the routine CALCULATION and producing fields of
sizes.

The name of an order should not exceed 16 characters.

This name is that used by the user in a command file Aster.

.

To define the number of routine FORTRAN carrying out the order: [§4]

.

To describe the various key words: [§5], [§6], [§7]*. It is the heart of the catalogue.*

.

To close the *open bracket after key word OPER/PROC/MACRO.*
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
To introduce a new order

4
To define the bond between the catalogue and FORTRAN program associated

*The reserved key word of the reader of catalogue **COp** allows the call to the routine FORTRAN OPxxxx which*
*carry out the task of the order [§9]. The argument of **COp** is a strictly positive entirety included/understood enters*
1 and 199. The number is allotted by the team codes (cf [A2.01.02]).

On the example considered routine OP0006 will be called during the execution of the order AFFE_MATERIAU.

5
To define the attributes of the single-ended spanner words

General syntax to declare a word single-ended spanner is:

MOT_CLE = SIMP (statut=…, typ=…, into= (…,), defaut=…

min=…, max=…, val_min=…, val_max=…, validators=

...
),

*Among the attributes attached to a key word, alone **statute** and **typ** are obligatory for any key word simple:*

.
The statute:

*The definition of the statute by the word reserved **statute** is obligatory.*
The recognized statutes are only:

"O"
Obligatory: in this case the key word will have to appear obligatorily in
body of call of the ordering of the user (except if this key word is under one
key word optional factor in which case the single-ended spanner word is obligatory as soon as it
key word factor appears).
"F"
Optional: in the contrary case.

.
The type:

*The declaration of the type by the word reserved **typ** is obligatory.*
The recognized types are:

typ = "I"
for the entireties
typ = "R"
for realities
typ = "It
for the complexes
typ = Type_de_concept for the concepts
(without dimensions!)
typ = "TX"
for the texts
typ = "It
for logics

Notice on the concepts:

The type of awaited concept is a type of concept created by another order that pending order. It appears among the list of the concepts defined in the catalogue of declaration of the concepts: accas.capy.

The type of the awaited concept is not necessarily single, it can be a list. This list declares itself as follows:

MATR_ASSE = SIMP (
...
typ= (matr_asse_depl_r, matr_asse_depl_c,
...
)
,
...
)

The documentary syntax of this example is:

MATR_ASSE: m

/[matr_asse_depl_r]
/
[matr_asse_depl_c]

.
Default value for a key word:

It is possible to assign a default value to a key word not receiving an argument of type "concept". The declaration is done by the reserved word of the reader of catalogue: ***defect***

Examples:

PRECISION

=SIMP (
statut=' f', typ=' R',
***defaut=1.E-3**),*

FILE =SIMP (
statut=' f', typ=' TXM',
***defaut=** " RESULT "),*

.
List acceptable values:

So that the supervisor controls the validity of the contents of certain arguments, it is possible to declare the values of the awaited arguments. This declaration is done by the reserved word ***into***

Examples:

INFORMATION

=SIMP (statut=' f', typ=' I', defaut= 1, into= (1,2)),

The key word INFORMATION is optional, its default value is 1 and the only accepted values are 1 and 2. Documentary syntax is:

INFORMATION:
/
1
[DEFECT]

/ 2

ALL
=SIMP (
statut=' f',
typ=' TXM',
***into=** ("YES", "NOT")),*

.

A number of awaited values:

The reserved words ***min*** *and* ***max*** *make it possible to control the length of the list of the arguments waited behind the single-ended spanner word. By defect, if nothing is specified in the catalogue, one awaits one and only one value behind a single-ended spanner word (max = 1). Attention, to declare min = 1 does not bring anything and does not amount especially making the word key obligatory. If a number*
*potentially unlimited of elements is waited, syntax is max = "**".*

Examples:

NET
=SIMP (
statut=' f',
typ=ma,
max=' ** '*),*

The user can enter as many here names of meshs it wishes.

CENTER
=SIMP (
statut=' f', typ=' R', defaut= (0. , 0. , 0.),

***min=3**, **max=3**),*

A vector here is awaited (list of exactly three realities).

.
Beach of acceptable values:

For the entireties and realities, one can specify the values minimum and/or allowed maximum:

NAKED
=SIMP (
statut=' o', typ=' R',
val_min=-1E+0,
val_max=0.5E+0),

On this example, NAKED must belong to the interval [-1, 0.5]. Values given by both reserved words are included in the interval.

.
More complicated criteria:

In addition to the beaches of values and the cardinal of the list, one can impose criteria more complicated on the value provided by the user, they are the validators, defined in Noyau/N_VALIDATOR.py.

One can program the new ones, according to the needs. Those currently present are:
.
RangeVal (low, high): identical to the behavior of val_min, val_max
.
CardVal (min, max)
: identical to the behavior of min, max
.
PairVal
: the provided entireties must be even
.
EnumVal (list)
: identical to the behavior of into
.
NoRepeat

: checking of the absence of doubled blooms in a list
.
LongStr (low, high)
: checking length of a character string
.
OrdList (order)
: checking which a list is increasing or decreasing

Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

6
Case of the key words factors

The key words factors are obligatory or optional. It is possible to control the numbers minimum and maximum of occurrences of a key word factor.

*The declarations are done thanks to the key word reserved **FACT***

.

*The statute: **statute***

It is related to the key word factor.

The recognized statutes are only:

"O":

Obligatory
“F”: Optional
“of: Optional but used by optional defect, i.e for the user with the seizure but obligatory for the operation of the code. Default values of single-ended spanner words must define the syntax of all the key word factor seen of code when the user does not inform anything. The user does not need to inform the key word factor and its single-ended spanner words so that there exists and is visible of the supervisor with the execution.

.

The number of occurrences:

*As for the single-ended spanner words, the reserved words **min** and **max** make it possible to specify awaited occurrences of the key words factors. If one puts nothing, the situation by defect is max=1, the key word factor is then not répétable.*

Examples:

*MCFACT = **FACT** (statut=' f', **min=3**, **max=3**,…)*
the key word factor is obligatory and must appear three times exactly.

*MCFACT = **FACT** (statut=' f', max=' ** ',…)*
the key word factor is optional but can appear time as many as one wants.

*MCFACT = **FACT** (statut=' of, **max=1**,…)*
the key word factor is optional but if the user does not specify it, it nevertheless is taken in count and the values of the single-ended spanner words (under the key word factor) are affected by defect.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

7
To exclude or gather key words: key word rules

*The use in the catalogues of orders of the key word reserved **rules** describes below and of*
***blocks** (following paragraph) makes it possible to entirely reproduce the logic of sequence of the words keys described in the paragraph syntax of the documentation of use. There should not thus be no checking of syntax (tests on the presence or the contents of key words) on the level of routines FORTRAN op0nnn.f.*

*Reserved words, present under the key word **rules**, which follow make it possible to ensure a coherence on the simultaneous presence of the key words of the order. Behind these reserved words (EXCLUDED, UN_PARMI, TOGETHER,…), a list of key words is found (order). These key words are either of the single-ended spanner words (under the same key word factor), or of the key words factors. In the continuation this paragraph one will use nothing any more but the term “key word”.*

EXCLUDE mc1,
mc2,…, mcn
The key words are excluded mutually.

UN_PARMI mc1,
mc2,…, mcn
One of the key words of the list must be obligatorily present. Its presence excludes the others.

TOGETHER mc1,
mc2,…, mcn
The key words must appear together.

AU_MOINS_UN mc1,
mc2,…, mcn
It is necessary that at least a key word among the list is present. It is licit of in
to have several present.

PRESENT_PRESENT mc1,
mc2,…, mcn
If the key word mc1 is present then the key words mc2,…, mcn must
to be present.

PRESENT_ABSENT mc1,
mc2,… mcn
If the key word mc1 is present then the key words mc2,…, mcn must
to miss.

Note:

PRESENT_PRESENT is different from UNIT since for PRESENT_PRESENT mc2 can
to be present, without mc1 being it.
PRESENT_ABSENT different from IS EXCLUDED since for PRESENT_ABSENT mc2,…, mcn
can be present sets if mc1 misses.

Example:

***rules** = (*
***UN_PARMI** ("NODE", "GROUP_NO", "MESH"),*

***PRESENT_PRESENT** ("MESH", "NOT"),),*

NODE
=SIMP (…),

NET
=SIMP (…),

NOT
=SIMP (…),

GROUP_NO =SIMP (…),

The supervisor checks that the user gave one and only one of the key words well among NODE, GROUP_NO and MESH and, if it gave MESH, that NOT is also present.

Caution:

The key words handled in the key word ***rules*** *must be on the same level of parenthèsage that this one in the tree structure defined by the key words factors and the blocks. Several words keys* ***rules*** *can be present in the same catalogue, with the principal root of order or under key words factors, the logical conditions applying whereas to single-ended spanner words of the key words factor.*

---

8 Them blocks

The blocks are appeared as key words factors. They allow two things:

- *to translate in the catalogue of the order of the logical rules relating to the value or it type of the contents of the single-ended spanner words; whereas conditions under the key word* ***rules*** *relate that to the presence or the absence of the key words. With the help of the satisfaction of conditions, one can thus gather key words unit or affect attributes to them (defects…) private individuals,*
- *to gather the key words by families for more clearness in EFICAS. When the presence of key words is subjected to condition, will be visible with the user since EFICAS only them*

optional key words induced by this condition, with their attributes (defect,…) private individuals with the condition.

Examples:

SOLVEUR =FACT (statut=' of, min=1, max=1,

METHODE=SIMP (statut=' f', typ=' TXM', defaut= " MULT_FRONT ",
into= (“MULT_FRONT”, “LDLT”)
),

***will b_mult_front** =BLOC (**condition** = “METHOD == “MULT_FRONT””,*

fr= " Parameters of the frontal method multi ",

RENUM=SIMP (
statut=' f', typ=' TXM', defaut= " MDA ",
into= (“MANDELEVIUM”, “MDA”, “MONGREL”)
),

),

b_ldlt

*=BLOC (**condition** = “METHOD == “LDLT””,*

fr= " Parameters
of

method
LDLT ",

RENUM=SIMP (
statut=' f', typ=' TXM', defaut= " RCMK ",
into= (“RCMK”, “WITHOUT”)
),

TAILLE=SIMP (statut=' f', typ=' R', defaut=

400.
),

),
),

The blocks are named by the developer. Their name must start with "b_". In the example,
if METHOD is worth MULT_FRONT, then single-ended spanner word optional RENUM will appear with three values
possible declared under into. So on the other hand METHOD is worth LDLT, the same key word will be present
but with different possible values; moreover it will be then possible to inform the key word
simple SIZE. These key words and their respective attributes will not appear well on in EFICAS
that after the user will have affected a value with the single-ended spanner word METHOD.

*b_nomdubloc=BLOC (**condition=** " (MOTCLE1! = None) gold (Astype (MOTCLE2) ==grma) ",*

One shows on this example that the condition can be multiple (articulated by gold/and) and can carry also on the presence of the key word (MOTCLE1! = None) or the type of what it contains (Astype (MOTCLE2) == grma).

Caution:

.

*the key words handled under the **condition** of the **BLOCK** must be on the same level as it block itself in the tree structure defined by the key words factors and the blocks. Several key words **BLOCK** can be present in the same catalogue, with the principal root of order, under key words factors or in other blocks. In the example above, single-ended spanner words RENUM and TAILLE_BLOC are on the same level, lower than that of METHOD, will b_mult_front and b_ldlt, him even lower than that of the key word factor SOLVEUR. The conditions tested in the two blocks relate thus only to the key word simple METHOD of level immediately higher than the blocks themselves,*

.

it is however possible to be freed from this rule while informing for a single-ended spanner word with the root of the order, the attribute position=' global'. It will be then visible in all them conditions of blocks.

.

it is necessary to pay attention to the possible conflicts when the same key word is present under two different blocks. The conditions of activation of the two blocks must then be excluded. It is it case of the example above with single-ended spanner word RENUM: there cannot be conflict since

2 conditions METHODE=' MULT_FRONT' and METHODE=' LDLT' cannot be at the same time satisfied.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

9
To typify the produced concept and to enrich it

9.1
To typify the produced concept

*The key word reserved **sd_prod** makes it possible to carry out the declaration of the type of produced concept. If*
*order always produces the same structure of data some is the context, **sd_prod** is followed*
name of corresponding concept, already declared in the catalogue of declaration of the concepts:
***accas.capy**. All concepts being able to be produced by orders and/or to be used in*
key words are declared in this file which is managed as a catalogue of order.

Example:

In the catalogue of the order:

CALC_THETA=OPER (nom= " CALC_THETA ", op=54, sd_prod=theta_geom,…

*In the file **accas.capy**, declaration of the concept **theta_geom**, drifting of **result**:*

*class **result** (ASSD): pass*
*class **theta_geom** (**result**): pass*

In certain cases the developer wants that the operator produces a concept whose type is given dynamically (i.e with the execution) by the presence of a key word or by its type or by its value. Then, one names the structure of data produced of a name particular to the order and one add catalogue of the order at the head a "function" python which defines the rules of typing concept. The most current situations are as follows, but it is possible to write more complex "functions" for, for example, to test a single-ended spanner word under a key word factor.

Heading of the catalogue:

*def operateur_prod (MCLE1, MCLE2, MCLE3, ** args):*

yew MCLE1 == "VALEUR1"
: return
SD1

yew (MCLE2! = None)

: return SD2

yew (AsType (MCLE3) == SD3)
: return SD4

. . . . . .

raise AsException ("standard of concept result not envisaged")

Body of the order:

NOM_COMMANDE=OPER (
nom= "
NOM_COMMANDE
",
op=54,
sd_prod= ***operateur_prod****,…*

All the single-ended spanner words of the order intervening in the tests must have placed in

*argument of **operateur_prod**. In the first case, one standard the structure of produced data (i.e one assign a value to operateur_prod) according to the contents of single-ended spanner word **MCLE1**. In the second*
*case, one standard according to the presence of **MCLE2**; (**MCLE2**! = **None**) means **MCLE2** is present. In*
*third case, one standard according to the type of the concept which is behind **MCLE3**. Names of structures of*
*finally produced data, **SD1**, **SD2**, **SD4**, are taken in the catalogue of declaration*
***accas.capy operateur_prod** will have been an intermediate function whose value is given according to the context.*

Example:

*def **mode_iter_inv_prod** (**MATR_A**, **MATR_C**, **TYPE_RESU**, ** args):*
*yew **TYPE_RESU** == “MODE_FLAMB”: return **mode_flamb***
*yew AsType (**MATR_C**) == matr_asse_depl_r: return **mode_meca_c***
*yew AsType (**MATR_A**) == matr_asse_depl_r: return **mode_meca***
*yew AsType (**MATR_A**) == matr_asse_pres_r: return **mode_acou***
*yew AsType (**MATR_A**) == matr_asse_gene_r: return **mode_gene***
raise AsException (“standard of concept result not envisaged”)

MODE_ITER_INV=OPER (nom= " MODE_ITER_INV ", op=44, sd_prod=mode_iter_inv_prod,….

---

*In this case, if key word TYPE_RESU has as an argument text “MODE_FLAMB then” the concept product, **mode_iter_inv_prod**, are of the **mode_flamb** type, if not one looks at the other **yew** which must to be seen like **elseif**. If the type of the concept present behind single-ended spanner word MATR_C is*

*matr_asse_depl_r, then the produced concept, **mode_iter_inv_prod**, is of the **mode_meca_c** type.*
and will caetera.

Caution:

Not to have nasty surprises nor to harm the legibility of the catalogue, it is necessary to take care of it that only one condition only is satisfied in the function with typing of the concept. In the example above, rules of the body of the catalogue of order MODE_ITER_INV should not simultaneously authorize to have

***TYPE_RESU** == "MODE_FLAMB"*
and
*AsType (**MATR_A**) == matr_asse_gene_r*

*If it is not the case, it is the first satisfied condition met which precedes. The order of declaration of the tests **yew** of the function of typing has an importance then: to handle with precaution.*

9.2
To enrich the produced concept

*The reserved key word **reentrant** makes it possible to specify if the concept produced by an operator is created or employed again then enriched. In this last case, to announce in the command file which one re-uses a concept, the key word reserved **reuse** follow-up of the name of the concept will be present.*

Three situations are possible:

reentrant=',
the produced concept does not preexist to the call of the order in run, it is created. Example:

LIRE_MAILLAGE=OPER (sd_prod=maillage, reentrant=',

*reentrant=' **o**',*
the concept produces was necessarily generated by this operator or another. It will be enriched. Example:

CALC_META=OPER (sd_prod=evol_ther, reentrant=' o',

In this case, the structure of data evol_ther must obligatorily to be created as a preliminary by the operator of thermics to be enriched here by the metallurgical ordering of postprocessing.

*reentrant=' **f**',*
The two situations are possible. It is the case of the orders

*calculating evolutions (structures of data evol_ ***). One can to want to connect the second transitory calculation behind a first and to supplement the structure of data of the new moments of calculation obtained. Example:*

In the catalogue:
STAT_NON_LINE=OPER (sd_prod=evol_noli, reentrant=' f',

In the command file:
U=STAT_NON_LINE (…)
*U=STAT_NON_LINE (**reuse=U**,…)*

This possibility that one offers to the user must maturely be reflected and must remain an exception to the general rule which wants that one does not modify a concept provided in entry. Indeed, when one concept is modified, the concepts which had been created by using it (before the change) risk to lose the coherence which they had with him. That can thus lead to a base of data incoherent.

Today the only modifications of concepts authorized are enrichments: one adds information without modifying existing it, or complete destruction of the concept. Only exception to this rule is factorization in place of the MATR_ASSE (operator FACT_LDLT), this exception is justified by problems of obstruction of the data bases.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

---

10 Routine of use

10.1 Name of the routine

OPxxxx is the routine which carries out the associated order. The numbers of the Opxxxx.f routines are
allotted by the person in charge for documentation. xxxx is a number coded on four digits.

10.2 Two stages

The supervisor proceeds in 2 stages:

one 1st
stage
:
construction of the tree of the python objects
: command set,
key orders, words, syntactic checking python, checking of
coherence with the catalogue,
one 2nd stage:
call to OPxxxx requires execution of calculations

The call to the operators, since the supervisor, is done automatically according to the attribute
op=xxxx well informed in the catalogue, by:

CALL
OPxxxx
(IER)

10.3 Recovery of the arguments of the order

Real arguments (those which the user wrote behind the key words in his file of
orders) are recovered in the OPxxxx routine by requests made to the supervisor.

.
Requests of access to the values:

A whole of subroutines specific to each known type of the supervisor is
available:

GETVIS
recovery of whole values,
GETVR8
recovery of actual values,
GETVC8
recovery of complex values,
GETVLS

recovery of logical values,
GETVID
recovery of values identifiers (name of concepts),
GETVTX
recovery of values texts,
GETLTX
recovery lengths of the values texts,
GETTCO
recovery of the types of a concept.

.
Request of access to the result:

Subroutine GETRES makes it possible to obtain the name user of the produced concept, the standard of concept associated with the result and the name of the operator or the order.

These routines are described in [D6.03.01] Communication with the supervisor of execution.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

Intentionally white left page.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
To introduce a new macro-order

Date:
01/12/05
Author (S):
C.** Key **DURAND
:
D5.01.02-C Page
: 1/16

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D5.01 booklet: -
Document: D5.01.02

To introduce a new macro-order

Summary

This document describes how to define and use the macro-orders in python.

---

1 Introduction

This document describes the use and the development of the macro-orders in Python for Code_Aster. These macro-orders can be restored in the code and visible of the user like orders with whole share. But they can also be placed in the file of order itself without the user has to touch neither with the achievable one, nor with the catalogue of orders. Moreover, they have the advantage of being written "naturally" in the language of

order: the body of an macro-order is similar to an ordinary command file which the same sequence of orders would generate.

The developer will may find it beneficial great to read the programming of existing macro-orders, under macro bibpyt/in the repertory of installation of Code_Aster.

2
What macro?

Macro is an order which gathers the execution of several subcommands. It is usable in a command file like any other order and has its characteristic catalogue, defining its syntax. Several types of macro are possible:

·
macros specific to the supervisor, implemented in Python and FORTRAN: for example FORMULATE, INCLUDE, BEGINNING…
·
macros developers implemented in Python.

This document treats definition and use of the macros in Python.

3
To use the macros

To use the macros in Python is simple. Compared to simple orders like OPER or PROC, the only difference relate to the concepts produced by the macro one. A simple order, of type OPER, has only one produced concept which one will find on the left of the sign "=", as follows:

concept = order (word-key-simple-or-factors)

A simple ordering of type PROC does not have any produced concept and is written:

order (word-key-simple-or-factors)

An macro-order, of MACRO type, can have several produced concepts. One which one will find with left of the sign =, as for a OPER, others like arguments of the simple key words or factors. One will present the instructions for a simple key word. It extends easily to the key words *factors. Certain key words are likely to produce concepts. To ask one macro-order to produce this concept, the user will write on the right sign = following the name of key word, CO ("nom_concept"), as in the example which follows:*

MACRO_MATR_ASSE (NUMEDDL=CO ("num"))

This causes to create a produced concept of name num at exit of the order MACRO_MATR_ASSE. Its type will be given according to the conditions of call of the order. CO is a reserved name which makes it possible to create named produced concepts, not typified, before the call of the order. It is the order which will allot the good type to this concept.

4
To define an macro-order in Python

It is necessary to define:

· the catalogue itself of the key words composing the macro one,
· method of typing of the structures of produced data,
· the method Python defining the body of macro: produced "basic" orders by the macro one and their sequence.

The first two points are common with the writing of an ordinary order (except for one minor difference in the method of typing).

It is possible to restore all this in the catalogue of orders of the code and to thus return visible macro-order of all. One can also preserve his development deprived with the advantage of not having to modify the parameters of execution or the achievable one while placing these three elements directly at the head of the command file, or the important one (importation Python) since an agreed localization.

4.1
To write the catalogue of the macro-order

The catalogue of macro is similar to that of a simple order. The three differences are:

· one declares an object MACRO (and not PROC or OPER),
· the key word reserved COp does not contain an entirety (indicating the number of routine FORTRAN of
high level for OPER and PROC) but a name of method python,
· the produced concept necessarily single, is not declared on the left a sign "=".

Produced concepts can be specified as arguments of a simple key word. If a key word
simple can accept a concept produced like argument, it is necessary, in addition to the type, specify CO in
tuple typ.

Simple example of key word accepting a produced concept or an existing concept:

NUME_DDL =SIMP (statut=' o', typ= (nume_ddl, CO))

Here, the key word accepts in argument an existing concept of nume_ddl type or a concept to be produced
of a type which will be determined by the order.

4.2
To define the type of the produced concepts

The definition of the type of the produced concepts of an macro-order is carried out in a way similar to that of a simple ordering of type OPER.

If the order produces only one concept which one will find on the left of the sign = as in:

=COMMANDE has ()

one will proceed in the same way as for a simple order of type OPER.

If the macro-order can produce several concepts whose some in arguments of
key words, some additional information should be added. First of all, it is necessary absolutely
to provide a function Python, named sd_prod, in the definition of macro. Then, key words
simple containing the name user of the concept to be produced must be of type CO (reserved name).

For example:

```
def ma_macro_prod (coil, NUME_DDL, MATRIX, ** args):
yew isinstance (NUME_DDL, CO):
self.type_sdprod (NUME_DDL, nume_ddl_sdaster)
self.type_sdprod (MATRIX, matr_asse_depl_r)
return evol_noli
....
MA_MACRO =MACRO (sd_prod=ma_macro_prod,…
STAMP

=
SIMP (statut=' o', typ=CO),
....
NUME_DDL
=
SIMP (statut=' o', typ= (CO, numeddl)
),
)
```

In this case, three concepts can be produced:
· in a traditional way, a concept of the evol_noli type which will have been given by the user to left of the sign "="
· a concept of the matr_asse_depl_r type whose name will have been provided by the user behind the single-ended spanner word STAMPS
· for case NUME_DDL, the user has the choice between providing here a concept numeddl already existing or to make it produce by the macro one, in which case it determines itself its naming as in the case STAMPS.

When a key word can have in argument a concept to produce, the key word must appear in list arguments of the function sd_prod and the concept must be typified by using the method type_sdprod of the argument self-service which is the macro-order object.

NB: *This argument self-service is not present for a OPER or a PROC.*

4.3
To define the body of macro

4.3.1 Transmission of the key words the macro one with the method of construction (it body)

*The body the macro one will be defined in a function whose arguments are similar to those of function sd_prod. The first argument is the macro-order object, coil, the following are them key words necessary to express the body of macro. Useless key words to express it bodies the macro one will be ignored by the use of the argument ** args.*

Only the high level key words are transmitted: MCSIMP of first level, MCFACT. These key words are then called upon very simply by their name.

Example of function body:

*def ma_macro_ops (coil, UNITE_MAILLAGE, ** args):*
....

_NOMLMA = LIRE_MAILLAGE (UNIT = UNITE_MAILLAGE)
....

Here, UNITE_MAILLAGE is a MCSIMP the macro one, its contents (concept, list, string,… little import) is affected with the MCSIMP CONTROL UNIT LIRE_MAILLAGE.

Case of a key word factor:

*def ma_macro_ops (coil, MATR_ASSE_GENE, ** args):*
....

yew MATR_ASSE_GENE [“MATR_ASSE”]:
....

MATR_ASSE_GENE is a MCFACT the macro one, MATR_ASSE is one of its under-MCSIMP. One MCFACT is handled like a dictionary.

Easy way: in this last example, one tests very simply the presence of MATR_ASSE: if the user a key word (simple or factor did not inform), it is worth by None defect.

4.3.2 To call an order in the body the macro one

To call an order in the body the macro one, it is possible to use two methods.

The first is simplest. It is possible if the order exists in the total context of function defining the body of macro. One is in this case within the catalogue of reference code. One will make then simply:

num=NUME_DDL (METHODE=…,…)

If the order does not exist in the context, it is necessary to question the catalogue by the intermediary of the method get_cmd of the object macro-order coil to obtain this order:

NUME_DDL=self.get_cmd ("NUME_DDL")
num=NUME_DDL (METHODE=…,…)

It is essential to use the exact name of the order as name of the variable which will contain the return of the method get_cmd. Indeed this name is used to locate the line of text containing this name and thus to identify the name of the produced concept (here num).

There is no obstacle so that the orders "girls" produced by the macro-order are

they-even of the macro-orders.

4.3.3 Concept interlinks produced in the body and concepts produced by macro

The concepts produced in the body of macro are several kinds:

.
concepts named automatically and destroyed at the end of the execution the macro one order. One thus should not any more need some thereafter and it is necessary in particular to take care of it that the concept produced by the macro one does not refer there. To indicate that it is about one concept of this kind, it is enough to give him a name which starts with _ _ (double underscore)

Example:

__a=CALC_MATR_ELEM (MODELE=MODELE)
Then, as one can read it in the file of messages, an automatic name of concept, preceded by a point is generated:

.9000005=CALC_MATR_ELEM (MODELE=MODELE)

Once left macro, the object corresponding in the name of concept .9000005 does not exist any more, neither in the space of names of the supervisor, nor in the jeveux base.

.
concepts named automatically and preserved in the jeveux base at the end of macro-order. To indicate that it is about a concept of this kind, it is enough to give him a name which starts with _ (simple underscore)

Example:

_a=CALC_MATR_ELEM (MODELE=MODELE)

Then, as one can read it in the file of messages, an automatic name of concept, preceded by a underscore is generated:

_9000005=CALC_MATR_ELEM (MODELE=MODELE)

Once left macro, the object corresponding in the name of concept _9000005 does not exist in the space of names of the supervisor, it is on the other hand present under this name in the base jeveux.

This type of object answers the particular situations where the concept produced by the macro one "depends" on a concept upstream which will have to be always present: for example a model relative with a grid, one matr_asse relative with a nume_ddl.

.

concepts intended to become produced concepts of macro. To indicate that it acts of a concept of this kind, it is necessary to call the DeclareOut method of the object macro coil with, like arguments, the name of the variable of return of the order and the object resulting from key words of the macro-order.

Example with matrix, argument of a key word the macro one:

self.DeclareOut ("mm", matrix)
mm=ASSE_MATRICE (............)

.

the concept of exit the macro one (necessarily single) is treated in a similar way in indicating the concept of exit self.sd.

self.DeclareOut ("mm", self.sd)
mm=ASSE_MATRICE (............)

mm will become the concept of exit the macro one, it will bear the name given by the user in its command file (and not mm).

4.3.4 To number the macro one

In postings of the file of messages, all the orders are numbered. For

to increment this meter, it is systematically necessary to at the head invite the following method body of
macro: self.set_icmd (1)

It is particularly important to appeal this before any order “
girl
” of
macro-order because this method also initializes the total measurement of time CPU for
macro.

4.3.5 To treat the errors

As for an order in FORTRAN, it is possible to detect errors of use in
body the macro one by the Utmess utility, identical in its operation to its homonym
FORTRAN [D6.04.01]. With this intention, it is necessary to import this method since the module of the utilities
Python.

Example:

from Utilitai.Utmess importation UTMESS
...
*message= “the two ends must be *
*message=message+' of the same length in the case of symmetry *
UTMESS (“F”, “MACR_ASCOUF_MAIL”, message)

The first argument indicates the nature of the error or alarm, the second specifies the appealing routine and
the last contains the message bound for the user.

D5.01.02-C Page
: 7/16

4.3.6
postings

Postings in the files of message and result can come from the programmed part in Python like that programmed in FORTRAN. So that sequencing of these postings that is to say well respected (and thus that the aforementioned files are quite readable), it is strongly disadvised using the order Python print but rather to employ the utility posts, of the module aster.

Example:

```
importation aster
...
aster.affiche (“MESSAGE”, mon_texte_dans_une_string)
```

The first argument is worth “MESSAGE” or “RESULT” following the target file.

4.3.7 To identify the concepts produced by the macro one

In certain circumstances, it is necessary to determine if a concept is produced by macro itself or has summer produced by a preceding order. This is possible while testing if the concept in question is present or not in the list of the concepts produced by the macro one which is given by the attribute sdprods object macro coil.

Example:

```
yew nume_ddl in self.sdprods:
# the concept nume_ddl is produced by the macro one
# it is necessary to call order NUME_DDL
lnume = 1
else:
# the concept nume_ddl already exists.
lnume=0
```

Attention, sdprods does not contain the produced concept turned over by the macro one which is in the attribute sd of coil.

4.3.8 Dynamic creation of orders: a number of variable key words, contained contextual

In certain cases, according to the value of the options, the same order will be called with different key words or of the different arguments. To treat this situation and to generate dynamically order, one builds a dictionary containing the key words to write which is then transmitted in argument of the order, preceded by the characters "**". The dictionary "is then unfolded", the keys are the arguments (key words), followed contents of the key, behind the sign "=".

This Python dictionary can be built progressively with the examination of the options.

Example:

moscles= {}
moscles ["INFORMATION"] = 2

standard yew ("GROUP_MA_BORD") ==types.StringType:

motscles ["CREA_GROUP_NO"] = _F (GROUP_MA = GROUP_MA_BORD)
else:

motscles ["CREA_GROUP_NO"] = []

for grma in GROUP_MA_BORD:

motscles ["CREA_GROUP_NO"]. suspends (_F (GROUP_MA = grma))

_nomlma = DEFI_GROUP (reuse = _nomlma
GRID
=
_nomlma,
*** motscles*
)

The list of MCFACT behind key "CREA_GROUP_NO" of the dictionary motscles contains a _F object in the first case and a list of _F in the second case. The list is built by one suspends in a loop.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
To introduce a new macro-order

Note:

In this example, if CREA_GROUP_NO contains only one element, it is not a singleton but a string, from where need for importing the module standards to call upon StringType or ListType.

4.3.9 Call to an external code

If one wishes to carry out in macro third code by order EXEC_LOGICIEL or by the instruction python os.system, one must recover the way of the appealable software in a chain characters turned over by routine FORTRAN repout.f; this routine is appealable since python by the method repout of the module aster.

```
importation os.path
importation aster
way = aster.repout ()
miss3d = os.path.join (way, “miss3d”)
EXEC_LOGICIEL (…, SOFTWARE = miss3d,…)
```

5 Considerations on the use of the macros in Python

5.1 Definition of macro except catalogue

Standard method to add the definition of macro in Python for an execution of Code_Aster is to add it in the catalogue of reference of the code.

However, in certain cases:

- ***macro personal,***
-

test during the development,

it can be practical to add the definition of macro apart from the catalogue. With this intention, it is enough to
to create a module Python containing at the head the definition of macro by adding module the importation of
variables of the catalogue.

Simplified example:

```
from Cata.cata importation *
def ma_macro_prod (coil,…):
.....
def ma_macro_ops (coil,…):
....
MA_MACRO=MACRO (nom=' MA_MACRO',…)
```

Then with the use, the weather is enough in the command file to be during the importation of macro previously
defined.

Example of command file:

```
# the macro MA_MACRO is defined in the module ma_macro.py
from ma_macro importation MA_MACRO
a=MA_MACRO (...)
```

It is also possible to use the functionality of INCLUDE:

```
# the macro MA_MACRO is defined in file INCLUDE 45
INCLUDE (UNITE=45)
a=MA_MACRO (...)
```


---

6 Some errors

During the development or use of an macro-order in Python, errors of use or of definition of macro can occur. The most characteristic errors are presented below.

6.1 Invalid argument of key word detected in the body of macro

A possible user error can be detected on the level of the body of the macro-order during the phase of construction of the macros.

The user will obtain a report of the form then:

JDC.py: Construction of INVALID JdC
CR of 1st phase of construction of JDC
Stage: MA_MACRO line: 42 file:
"/home01/chris/ASTER/bugs/Pymacros/tmp.1670/ahlv100a_err20"

!!!
! THE FIRST OPTION MUST BE RIGI_MECA OR RIGI_THER OR RIGI_ACOU OR
! RIGI_MECA_LAGR
!!!
Fine Stage: MA_MACRO
fine CR of 1st phase of construction of JDC

6.2 Invalid argument of key word detected by a subcommand

Macro recovers the values of its key words to transmit them to orders called subcommands the macro one. It is possible that all the checks not having been made in definition of macro, an argument of key word of the subcommand is invalid. This error will be detected at the time of construction the macro one: Build method of the command set object J JDC.py file. The user will obtain a report of the following form:

JDC.py: Construction of INVALID JdC
BEGINNING CR validation: SansNom

Stage: MA_MACRO line: 42 file:
“/home01/chris/ASTER/bugs/Pymacros/tmp.1677/ahlv100a_err10”
Stage: NUME_DDL line: 102 file: “. /ma_macro_err1.py”
Simple key word: METHOD
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! louse is not an authorized value!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!
! The value: “louse” is not allowed for the key word: METHOD!

!!
Fine simple Key word: METHOD
Fine Stage: NUME_DDL
Fine Stage: MA_MACRO
FINE CR validation: SansNom

One can see that the order NUME_DDL which is a subcommand macro MA_MACRO is invalid because the key word METHOD is invalid.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

6.3
Error of syntax in the function body the macro one

If an error of syntax is made in the function body of macro, the user will obtain one report of the following form:
JDC.py: ERROR ACCAS - INTERRUPTION

>> JDC.py: REPORT/RATIO BEGINNING
CR phase of initialization
!!!
! error not envisaged and not traitee to prevent maintenance ahlv100a_err4.py
!
! Traceback (most recent call last): !
! Slip by “. /Eficas/Accas/commandes.py”, line 2029, in __init__!
! exec self.proc_compile in self.g_context!
! Spins “/home01/chris/ASTER/bugs/Pymacros/tmp.1634/ahlv100a_err40”, line 13, in?
!
! from ma_macro_err4 importation MA_MACRO!
! Slip by “. /ma_macro_err4.py”, line 46!
! a=!
! ^ !
! SyntaxError: invalid syntax!
!!!
fine CR phase of initialization
>> JDC.py: FINE REPORT/RATIO

6.4
Programming error in the function body the macro one

If a programming error is made in the function body of macro, the user will obtain a report of the following form:

JDC.py: Construction of INVALID JdC
CR of 1st phase of construction of JDC
Stage: MA_MACRO line: 41 file:
“/home01/chris/ASTER/bugs/Pymacros/tmp.1649/ahlv100a_err50”
!!
! impossible to build the macro MA_MACRO!
! Traceback (most recent call last): !
! Slip by “. /Eficas/Cata/asterexec.py”, line 174, in Build_MACRO!
! ier= apply (self.definition.proc, (coil,), D)!
! Slip by “. /ma_macro_err5.py”, line 44, in ma_macro_ops!
! a=1/0!
! ZeroDivisionError: integer division gold modulo by zero!
!!
Fine Stage: MA_MACRO
fine CR of 1st phase of construction of JDC

Version
8.1

Titrate:
To introduce a new macro-order

Date:
01/12/05
Author (S):
C. Key DURAND
:
D5.01.02-C Page
: 11/16

6.5
Programming error in the function sd_prod the macro one

If a programming error is made in the function sd_prod the macro one, the user
a report of the following form will obtain:
JDC.py: ERROR ACCAS - INTERRUPTION
>> JDC.py: REPORT/RATIO BEGINNING

!!
! Stage MA_MACRO line: 41 file: !
! impossible /home01/chris/ASTER/bugs/Pymacros/tmp.1301/ahlv100a_err60 D
to assign a type to the result!
! Slip by “. /ma_macro_err6.py”, line 10, in ma_macro_prod!
! a=1/0!
! ZeroDivisionError: integer division gold modulo by zero!

!!
>> JDC.py: FINE REPORT/RATIO

Important:

Moreover interpretation of the command file is stopped on the level of the error. In consequence the possible errors on the following orders will be detected only during later executions

7
An example of macro-order

One presents below an example of definition and use of macro-order in Python.
macro-order is defined in a Python module of name ma_macro.py. This module includes/understands 4
parts: a heading, a function sd_prod for the typing of the produced concepts [§7.1], a function
for the definition of the body the macro one [§7.2] and the definition of the key words the macro one [§7.3]. This
macro almost completely reproduces the functionalities of macro FORTRAN MACRO_MATR_ASSE.

Note:

For better including/understanding the paragraphs [§7.1] and [§7.2], one will be able to start by reading them
paragraphs [§7.3] and [§7.4].

7.1
Typing of the produced concepts

```
# the function ma_macro_prod makes it possible to define the type of all the concepts
# produced by the macro one. It is identical to that of MACRO_MATR_ASSE
def ma_macro_prod (coil, NUME_DDL, MATR_ASSE, ** args):
yew not MATR_ASSE: raise AsException ("Impossible to typify the concepts
results ")
yew not NUME_DDL: raise AsException ("Impossible to typify the concepts
results ")
self.type_sdprod (NUME_DDL, nume_ddl)
for m in MATR_ASSE:
opti=m ["OPTION"]
yew opti in ("RIGI_MECA", "RIGI_FLUI_STRU", "RIGI_MECA_LAGR",
"MASS_MECA", "MASS_FLUI_STRU", "RIGI_GEOM", "RIGI_ROTA",
"AMOR_MECA", "IMPE_MECA",
"ONDE_FLUI", "MASS_MECA_DIAG"): t=matr_asse_depl_r
yew opti == "RIGI_MECA_HYST": t= matr_asse_depl_c
yew opti == "RIGI_THER": t= matr_asse_temp_r
yew opti == "MASS_THER": t= matr_asse_temp_r
yew opti == "RIGI_THER_CONV": t= matr_asse_temp_r
yew opti == "RIGI_THER_CONV_D": t= matr_asse_temp_r
yew opti == "RIGI_ACOU": t= matr_asse_pres_c
yew opti == "MASS_ACOU": t= matr_asse_pres_c
yew opti == "AMOR_ACOU": t= matr_asse_pres_c
self.type_sdprod (m ["MATRIX"], T)
```

```
# the macro one does not have a produced concept turned over on the left sign =.
# It is necessary to turn over None
return None
```

7.2
Body the macro one

```
def ma_macro_ops (coil, MODEL, CHAM_MATER, CARA_ELEM, MATR_ASSE,
SOLVEUR, NUME_DDL, LOAD, INST,
** args):
"""
Example of macro in Python reproducing part of
functionalities of MACRO_MATR_ASSE
"""
# Initialization of the meter of errors
ier=0
# importation of the utility of error messages
from Utilitai.Utmess importation UTMESS
nom_macro=' MACRO_MATR_ASSE "
# One puts key word NUME_DDL in a local variable to protect it
numeddl=NUME_DDL

# One imports the definitions of the orders has to use in the macro one
# the name of the variable must be obligatorily the name of the order
CALC_MATR_ELEM=self.get_cmd ("CALC_MATR_ELEM")
NUME_DDL=self.get_cmd ("NUME_DDL")
ASSE_MATRICE=self.get_cmd ("ASSE_MATRICE")
```

 # the macro account for 1 in the classification of the orders
self.set_icmd (1)

yew SOLVEUR:
 # If the key word factor SOLVEUR is present
 # One can recover the values of the single-ended spanner words
methode=SOLVEUR [“METHOD”]
renum=SOLVEUR [“RENUM”]
else:
 # If the key word factor SOLVEUR misses
 # One assigns to the single-ended spanner words default values
methode=' MULT_FRONT'
renum=' MDA'

yew method == “LDLT”:
yew renum not in (“WITHOUT”, “RCMK”):
 # an error was detected. One increments the meter and records
 # a fatal message
ier=ier+1
UTMESS (“F”, nom_macro, “With method LDLT, RENUM must be WITHOUT or RCMK. “)

yew numeddl in self.sdprods:
 # If the concept numeddl is in self.sdprods
 # it must be produced by the macro one
 # it will thus be necessary to call order NUME_DDL
lnume = 1
else:
lnume = 0

iocc=0
for m in MATR_ASSE:
iocc=iocc+1
option=m [“OPTION”]
yew iocc == 1 and lnume == 1 and option not in (“RIGI_MECA”, “RIGI_MECA_LAGR”, “RIGI_THER”, “RIGI_ACOU”):
ier=ier+1
UTMESS (“F”, nom_macro,” the FIRST OPTION MUST BE RIGI_MECA OR RIGI_THER OR RIGI_ACOU OR RIGI_MECA_LAGR ")
 # an error was detected. One stops the construction of macro.
return ier
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

motscles= {“OPTION”: option}
yew CHAM_MATER! = None: motscles [“CHAM_MATER”] =CHAM_MATER
yew CARA_ELEM! = None: motscles [“CARA_ELEM”] =CARA_ELEM
yew CHARGES:
yew option not in (“RIGI_ACOU”, “MASS_ACOU”):
motscles [“LOAD”] =CHARGE
yew INST: motscles [“INST”] =INST
yew option == “AMOR_MECA”:
motscles [“RIGI_MECA”] =rigel
motscles [“MASS_MECA”] =masel
sigg=m [“SIEF_ELGA”]
yew sigg: motscles [“SIEF_ELGA”] =sigg
mh=m [“MODE_FOURIER”]
yew mh: motscles [“MODE_FOURIER”] =mh
__a=CALC_MATR_ELEM (MODELE=MODELE, THETA=m [“THETA”],
*PROPAGATION=m [“PROPAGATION”], ** motscles)*
yew option == “RIGI_MECA”:
 # One preserves in rigel the result at the case or
rigel=__a
yew option == “MASS_MECA”:
 # One preserves in masel the result at the case or
masel=__a
yew lnume and option in
(“RIGI_MECA”, “RIGI_THER”, “RIGI_ACOU”, “RIGI_MECA_LAGR”):
 # One declares that the produced concept of name num is actually it
concept numeddl
self.DeclareOut (“num”, numeddl)

 # One can pass from the equal key words has None. They are are unaware of
num=NUME_DDL (MATR_RIGI=__a, METHODE=methode,
RENUM=renum, TAILLE_BLOC=rbloc)
else:
num=numeddl
self.DeclareOut ("mm", m ["MATRIX"])
mm=ASSE_MATRICE (MATR_ELEM=__a, NUME_DDL=num)
 # One turns over the calculation of the errors
return ier
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

7.3
Definition of the key words

MA_MACRO = MACRO (nom= " MA_MACRO ", op=ma_macro_ops, sd_prod=ma_macro_prod,
fr= " Calculation of the matrices assembled (matr_asse_gd) for example of
rigidity, of mass ",
MODEL =SIMP (statut=' o', typ=modele),
CHAM_MATER =SIMP (statut=' f', typ=cham_mater),
CARA_ELEM =SIMP (statut=' f', typ=cara_elem),
CHARGE
=SIMP (statut=' f', typ= (char_meca, char_ther, char_acou)),
INST =SIMP (statut=' f', typ=' R'),
NUME_DDL =SIMP (statut=' o', typ= (nume_ddl, CO)),
SOLVEUR =FACT (statut=' of, min=01, max=01,
METHOD =SIMP (statut=' f', typ=' TXM', defaut= " MULT_FRONT ",

into= (“LDLT”, “MULT_FRONT”, “GCPC”)),
TAILLE_BLOC =SIMP (statut=' f', typ=' R'),

RENUM=SIMP (statut=' f', typ=' TXM', into= (“WITHOUT”, “RCMK”, “MANDELEVIUM”, “MDA”, “MONGREL”)),
),
*MATR_ASSE =FACT (statut=' o', min=01, max=' ** ',*
STAMP =SIMP (statut=' o', typ= (matr_asse, CO)),
OPTION =SIMP (statut=' o', typ=' TXM',

into= (“RIGI_MECA”, “MASS_MECA”, “MASS_MECA_DIAG”,

“AMOR_MECA”, “RIGI_MECA_HYST”, “IMPE_MECA”,

“ONDE_FLUI”, “RIGI_FLUI_STRU”, “MASS_FLUI_STRU”,

“RIGI_ROTA”, “RIGI_GEOM”, “RIGI_MECA_LAGR”,

“RIGI_THER”, “MASS_THER”, “RIGI_ACOU”, “MASS_ACOU”,
“AMOR_ACOU”)),
SIEF_ELGA =SIMP (statut=' f', typ=cham_elem_sief_r),
MODE_FOURIER =SIMP (statut=' f', typ=' I'),
THETA =SIMP (statut=' f', typ=theta_geom),
PROPAGATION =SIMP (statut=' f', typ=' R'),
),
*TITRATE =SIMP (statut=' f', typ=' TXM', max=' ** '),*
INFORMATION =SIMP (statut=' f', typ=' I', defaut=1, into= (1,2)),
) ;

7.4
Use of macro

 # This test of use is derived from the case test ahlv100a
 # the macro MA_MACRO is defined in the private module ma_macro
 # It should be imported with the order Python importation
from ma_macro importation MA_MACRO
BEGINNING (CODE=_F (NAME = “AHLV100A”))
F=500.
MAIL=LIRE_MAILLAGE ()
AIR=DEFI_MATERIAU (FLUIDE=_F (RHO = 1.3, CELE_C = (“IH”, 343. , 0. ,)))
CHAMPMAT=AFFE_MATERIAU (MAILLAGE=MAIL,
AFFE=_F (ALL = “YES”, MATER = AIR))
GUIDE=AFFE_MODELE (MAILLAGE=MAIL, VERIF=' MAILLE',
AFFE=_F (ALL = “YES”, MODELING = “3D”,

PHENOMENON = “ACOUSTIC”))
CHARACOU=AFFE_CHAR_ACOU (MODELE=GUIDE,
VITE_FACE=_F (GROUP_MA = “ENTERED”, VNOR =
(“IH”, 0.014, 0. ,)))
IMPEACOU=AFFE_CHAR_ACOU (MODELE=GUIDE,
IMPE_FACE=_F (GROUP_MA = “LEFT”, IMPE =
(“IH”, 445.9, 0. ,)))
 # If one compares the use of the macro MA_MACRO with that of
MACRO_MATR_ASSE
 # one will be able to note that there is no difference
MA_MACRO (
MODELE=GUIDE, CHARGE=IMPEACOU,
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

```
CHAM_MATER=CHAMPMAT,
NUME_DDL=CO (“NUM”),
MATR_ASSE= (
_F (MATRIX = CO (“MATASK”), OPTION = “RIGI_ACOU”),
_F (MATRIX = CO (“MATASM”), OPTION = “MASS_ACOU”),
_F (MATRIX = CO (“MATASI”), OPTION = “AMOR_ACOU”))
)
#
VECTELEM=CALC_VECT_ELEM (OPTION=' CHAR_ACOU',
CHAM_MATER=CHAMPMAT,
CHARGE=CHARACOU)
#
# IMPRESSION OF COMPLEX VECT_ELEM VECTELEM ACCORDING TO THE GRAIN NETS
#
IMPR_MATRICE (MATR_ELEM=_F (MATRIX = VECTELEM,
FORMAT = “RESULT”,
FILE = “RESULT”,
GRAIN = “MESH”))
VECTASS=ASSE_VECTEUR (VECT_ELEM=VECTELEM, NUME_DDL=NUM)
#
# _________________________CALCUL OF THE MODES_________________________________
#
# As for macro in FORTRAN, one uses the concepts produced by
the identifier
# created by the macro MATASK for CO (“MATASK”) for example.
MATASKR=COMB_MATR_ASSE (COMB_R=_F (MATR_ASSE = MATASK,
PART = “REAL”, COEF_R = 1.))
MATASMR=COMB_MATR_ASSE (COMB_R=_F (MATR_ASSE = MATASM,
```

PART = “REAL”, COEF_R = 1.))
#
MODES=MODE_ITER_SIMULT (MATR_A=MATASKR,
MATR_B=MATASMR,
CALC_FREQ=_F (OPTION = “BAND”,
NMAX_FREQ = 10,
FREQ = (1. , 1000. ,))
)

Intentionally white left page.
Data-processing handbook of Description
D5.01 booklet: - HT-66/05/003/A

Code_Aster ®
Version 4

Titrate:
To introduce a new loading of the "kinematic" type

Date: 05/05/97
Author (S):
J. PELLET

Key: D5.03.02
Page:
1/12

Mechanical department and Digital Models

Index: With
Diffusion: Developers

Data-processing handbook of Description

D5.03 booklet:

D5.03.02 document

To introduce a new loading of the type
“kinematic”

Summary:

This document presents the two utility routines making it possible to introduce the new ones easily types of boundary conditions “kinematics” (i.e of the linear relations between degrees of freedom unknown).

EDF
Direction of the Studies and Research

Electricity of France
Project Codes of Mechanics

Contents

---

1 Introduction

What one calls "loading" in Aster ("mechanical" vocabulary) is what the user defines in orders AFFE_CHAR_*. One distinguishes the loadings in general of "forces" [D5.03.01] and them loadings in "displacements" (or "kinematics").

This document explains how to introduce new loadings kinematics.

2
What a linear relation?

*This expression indicates a **linear constraint on the degrees of freedom** of the system to be studied:*

.
ddl of size TEMP_R for the thermal phenomenon,
.
ddl of sizes DEPL_R or DEPL_C for the mechanical phenomenon,
.
ddl of size PRES_C for the acoustic phenomenon.

The coefficients of this linear relation are real constants (or complexes), the second member can be real, complex or of type "function" (K8).

A linear relation can be written:

1
ddl + ddl2 +… + ddln =
1
2
N
0

where

I (or ç) (I = 1, N)
0 (or ç) (or function)

The degrees of freedom ddli are degrees of freedom carried by one or more different nodes.

Linear examples of relations:

DX (N1) = 0.
blocking of CMP "DX" of the node "N1"
TEMP (N3) = 100.
temperature imposed on 100. for the node "N3"
DY (N1) - DY (N2) = 0.
the nodes "N1" and "N2" have same displacement
"DY"
cos.
the node "
DX (N1) + sin. DY (N1) = 0.
N1 " east compels to move on the line

perpendicular with the vector (cos, sin) (in 2D).

3
How linear relations are introduced in a modeling?

The linear relations that one defined in [§2] force the solution which one seeks. They make started from what one in general calls the "boundary conditions". In Code_Aster they are one of components of the ***loads*** *(standard char_acou, char_ther, char_meca).*

These linear relations are thus introduced by the user via orders AFFE_CHAR_MECA (_F), AFFE_CHAR_THER (_F), AFFE_CHAR_ACOU, or AFFE_CHAR_CINE.

These linear relations can "dealt" with two ways:

.
one eliminates an unknown factor for each linear relation: method of elimination [D3.03.01],
.
one "dualise" the relation by adding 2 additional unknown factors to him: parameters of Lagrange [R3.03.01].

In Code_Aster, the method of elimination is used for the relations resulting from the order AFFE_CHAR_CINE. One will speak in this case about linear relations “kinematics”, although this term is not
very judicious. One limits oneself then to relations of the type:

DDL = cste

Other relations resulting from orders AFFE_CHAR_MECA, AFFE_CHAR_THER and AFFE_CHAR_ACOU
are always dualisées.

Examples of key words generating factor of the linear relations:

·
AFFE_CHAR_CINE
MECA_IMPO
·
AFFE_CHAR_MECA_F
LIAISON_OBLIQUE
·
AFFE_CHAR_THER
TEMP_IMPO
· AFFE_CHAR_MECA
LIAISON_DDL

Order AFFE_CHAR_CINE makes it possible to introduce all the simple linear relations easily (DDL = cste) that one can define.

On the other hand, although in theory (thanks to key word LIAISON_DDL), one can introduce any linear relation, the number of coefficients to be calculated can become very large. To think for example of the relations
linear that it is necessary to write for saying that 4 nodes are interdependent (connected by an indeformable solid).

The many key words making it possible the user to define these linear relations are there to facilitate it to him
work:

· LIAISON_OBLIQUE
for supports slipping into an oblique reference mark
· TEMP_IMPO
to impose a temperature
· LIAISON_GROUP
to connect nodes two to two

· ...

· and LIAISON_DDL
for the other cases…

This great number of key word (which will be able to only grow) requires to give itself software tools allowing:

·
not to duplicate a code unnecessarily,
·
to facilitate the introduction of new key words into orders AFFE_CHAR_MECA, AFFE_CHAR_ACOU and AFFE_CHAR_THER.

It is of these tools about which we will speak in the following paragraphs.

4
To introduce a new key word of type "linear relation"

We give in this paragraph a groundwork for the writing of a routine "carrying out" a key word of order AFFE_CHAR_MECA (or _THER or _ACOU). This key word factor allowing the user to define linear relations.

Are:

MFAC the key word factor
CAMFAC the name of the routine corresponding to him

The goal of routine CAMFAC is "to scan" the data of the user behind key word MFAC, of
to translate these data into linear relations and to store these relations in the load (here of char_meca type)
that the user is defining.

For that, one has two utility routines:

· AFRELA
:
to assign a linear relation to a SD of the type
LISTE_RELA (list of linear relations)
· AFRLCH
:
"to add" a SD LISTE_RELA to a SD
CHARGE

These routines force to pass by an intermediate SD (temporary) of type LISTE_RELA. That
a little the programming weighs down but presents the following advantages:

.
profits of performance, because routine AFRLCH is expensive in CPU,
·
a great flexibility to carry out the principle of overload (cf [§?]).

The groundwork of routine CAMFAC is thus the following:

SUBROUTINE CAMFAC
(CH)

*CHARACTER * (*) CH*
C in jxvar CH:
SD CHAR_MECA to be enriched
C drank:
to enrich the CH load by the definite linear relations
under the key word factor MFAC

buckle on the linear relations

.
acquisition of the coefficients of the linear relation: I

(routines GETVXX),
·
addition of the linear relation with SD LISTE_RELA
Call AFRELA (I, "&&CAMFAC.LISTE_RELA")

fine buckles

·
addition of SD LIST_RELA to the LOAD: CH
Call AFLRCH ("&&CAMFAC.LIST_RELA", CH)

END

Note:

·
SD LISTE_RELA (temporary) is specific to routine CAMFAC,
its name respects the convention of the names of objects of work: &&nom_routine,
·
the principle of overload (cf [U2.01.00 §3.7]) thus relates to only the events of
key word MFAC,
· this
SD is destroyed at the time of the call to AFLRCH.

---

5 Them routines AFRELA and AFLRCH

5.1 routine AFRELA

SUBROUTINE AFRELA (COEFR, COEFC, DDL, NODE, NDIM, DIRECT,
+ NBTERM, BETAR, BETAC, BETAF, TYPCOE, TYPVAL, LISREL)
C---
C DRANK: ASSIGNMENT Of a RELATION BETWEEN DDLS A a SD LISTE_RELA
C (IF OBJECT LISREL DOES NOT EXIST, IT EAST CREATES)
C
C---
C COEFR (NBTERM) - IN - R -: TABLE OF THE COEFFICIENTS OF THE RELATION
C THE COEFFICIENTS ARE REAL
C---
C COEFC (NBTERM) - IN - C -: TABLE OF THE COEFFICIENTS OF THE RELATION
C THE COEFFICIENTS ARE COMPLEX
C---
C DDL (NBTERM) - IN - K8 -: TABLE OF THE DDL OF THE RELATION
C---
C NODE (NBTERM) - IN - K8 -: TABLE OF THE NODES OF THE RELATION
C---
C NDIM (NBTERM) - IN - I -: DIMENSION OF THE PROBLEM (0, 2 OR 3)
C IF = 0 CHANGE NO OF REFERENCE MARK
C THE RELATION EAST GIVEN IN THE BASE
C TOTAL
C---
C DIRECT (3, NBTERM) - IN - R -: TABLE OF RELATIVE VECTORS A EACH
C TERM DEFINING THE DIRECTION OF
C COMPONENT WHICH ONE WANTS TO FORCE
C---
C NBTERM - IN - I -: A NUMBER OF TERMS OF THE RELATION
C---
C BETAR - IN - R -: ACTUAL VALUE OF THE SECOND MEMBER
C---
C BETAC - IN - C -: VALUE COMPLEXES OF THE SECOND MEMBER
C---
C BETAF - IN - K8 -: VALUE FUNCTION OF THE SECOND MEMBER
C---
C TYPCOE - IN - K4 -: TYPE OF THE COEFFICIENTS OF THE RELATION:

C = “REAL” OR “COMP”
C---
C TYPVAL - IN - K4 -: TYPE OF THE SECOND MEMBER
C = “REAL” OR “COMP” OR “FONC”
C---
C LISREL - IN - K19 -: NAME OF SD LISTE_RELA
C - JXVAR -
C---

Two cases of figure are to be considered:

has) the ddl to connect are given in the absolute reference mark: DX, DY,…
unquestionable b) ddl to be connected are given in a local reference mark.

Case A (all in the absolute reference mark):

NBTERM are the number of ddl connected by the relation.
NDIM is a vector filled with 0
DIRECT is useless.

Example 1:

*one wants to impose: 3.*DX (N1) +2.*DY (N2) -4.DRZ (N1) = “F” (foncion)*

NBTERM
=3

TYPCOE
= ' REEL'

TYPVAL
= ' FONC'

COEFR

=(3. , 2. , -4.)

NDIM

=(0 , 0 , 0)

DDL
= (“DX”, “DY”, “DRZ”)

NODE
= (“N1”, “N2”, “N1”
)

BETAF
=
“F”

Case B (local reference mark):

For each node implied in the relation, one can give a local reference mark in which the relation is simpler (the normal on a surface for example).

Example 2:

that is to say N, an unit vector of components (nx, ny, nz).

It is wanted that displacement according to N with the N3 node is null.

NBTERM
=1

TYPCOE
= ' REEL'

TYPVAL
= ' REEL'

COEFR
=(1.)

NDIM = (3)

DIRECT
= (nx, ny, nz)

DDL
= (“DEPL”)

NODE
= (“N1”)

BETAR
=
0.

Note:

.
NBTERM is not the number of terms of the final relation here (: 3).
.
When one employs (for a “term”) the possibility of a local reference mark NDIM/= the 0 name DDL must be conventionally “DEPL” or “ROTA”

Example 3:
RC
are:
n1: an unit vector of components (n1x, n1y, n1z) and

N2: an unit vector of components (n2x, n2y, n2z)

following data:

NBTERM
=3

TYPCOE
= ' REEL'

TYPVAL

= ' REEL'

COEFR
=(4.,2.,-3.)

NDIM = (3,0,3)

DIRECT
= (n1x, n1y, n1z, rbid, rbid, rbid, n2x, n2y, n2z)

DDL
= ("DEPL", "DX", "ROTA")

NODE
= ("N1", "N3", "N2")

BETAR
=
5.

describe the relation in the 7 terms:

4.* (n1x*DX (N1) +n1y*DY (N1) +n1z*DZ (N1))

+
2.*DX (N3)

+ -3.* (n2x*DX (N2) +n2y*DY (N2) +n2z*DZ (N2))
= 5.

Data-processing handbook of Description
D5.03 booklet:

Index A

Code_Aster ®
Version 4

Titrate:
To introduce a new loading of the "kinematic" type

Date: 18/09/2003
Author (S):
J. PELLET

Key: D5.03.02
Page:
8/12

5.2
routine
AFLRCH

```
SUBROUTINE AFLRCH (LISREL, LOAD)
C ----------------------------------------------------
C ADDITION Of a LISTE_RELA IN a LOAD
C
C THE IDENTICAL RELATIONS WITHIN LISTE_RELA ARE
C ELIMINEES. THE PRINCIPLE OF OVERLOAD EAST APPLIES:
C It IS the LAST SECOND MEMBER WHO IS PRESERVE.
C ----------------------------------------------------
C LISREL IN/JXVAR - K19 -: NAME OF SD LISTE_RELA
C THE LISTE_RELA EAST DESTROYED
A.C. END OF THE ROUTINE
C ----------------------------------------------------
C CHARGE IN/JXVAR - K8 -: NAME OF THE SD CHARGES
C THE LOAD EAST ENRICHED
C ----------------------------------------------------
```

6
Principle of overload

It can happen that the user defines several times the same linear relation (in a coefficient multiplier near).

Example:

3.DX (N1) -1.DY (N2) = 4.

6.DX (N1) -2.DY (N2) = 8.

3.DX (N1) -1.DY (N2) = 5.

Here, the first 2 equations are identical. Third is contradictory with the preceding ones (with cause of the second member).

If two equations of a linear system to solve have the same 1st member, one cannot reverse stamp, because the equations are not independent. It is thus necessary to eliminate all the equations which are multiples from/to each other.

One wants to be able to apply the principle of "overload" [U2.01.00 §3.7]: it is thus the last second member who is preserved.

This elimination of the "redundant" relations is made at the time or one adds the LISTE_RELA to CHARGE (routine AFLRCH). One eliminates the doubled blooms from the LISTE_RELA, one prints the eliminated relations, then one adds the relations preserved at the LOAD.

If the diagram advised is kept here [§4]: only one LISTE_RELA per key word factor, the principle of overload is thus naturally applied for each key word. The last events precede on first.

If one wanted (one does not want it today!) an overload between various key words (for example: DDL_IMPO takes precedence over FACE_IMPO), it would be enough that these 2 key words are associated the same LISTE_RELA:

CALL FACIMPO (CH, LISREL)

CALL DDLIMPO (CH, LISREL)

CALL AFLRCH (LISREL, CH)

Data-processing handbook of Description
D5.03 booklet:

Index A

7
Example: routine CALIAI

This routine treats key word LIAISON_DDL of the orders:

- AFFE_CHAR_MECA (_F)
- AFFE_CHAR_THER (_F)

```
SUBROUTINE CALIAI (FONREE, LOAD)
IMPLICIT REAL*8 (A-H, OZ)
CHARACTER*4 FONREE
CHARACTER*8 CHARGES
C -------------------------------------------------------------------
C MODIF MODELLED DATE 23/01/95 AUTHOR VABHHTS J.PELLET
C
C TO TREAT KEY WORD LIAISON_DDL OF AFFE_CHAR_XXX
C AND TO ENRICH THE LOAD (LOAD) WITH THE LINEAR RELATIONS
C
C IN: FONREE: "REAL" OR "FONC"
C IN/JXVAR: CHARGE: NAME Of a SD CHARGES
C -------------------------------------------------------------------
C ----------- COMMUN RUNS STANDARDIZE JEVEUX --------------------------
INTEGER ZI
COMMON/IVARJE/ZI (1)
REAL*8 ZR
COMMON/RVARJE/ZR (1)
COMPLEX*16 ZC
```

```
COMMON/CVARJE/ZC (1)
LOGICAL ZL
COMMON/LVARJE/ZL (1)
CHARACTER*8 ZK8
CHARACTER*16 ZK16
CHARACTER*24 ZK24
CHARACTER*32 ZK32
CHARACTER*80 ZK80
COMMON/KVARJE/ZK8 (1), ZK16 (1), ZK24 (1), ZK32 (1), ZK80 (1)
CHARACTER*32 JEXNOM, JEXNUM
C---------------- FINE COMMUN RUNS STANDARDIZE JEVEUX ----------------------
C
COMPLEX*16 BETAC
CHARACTER*7 TYPCHA
CHARACTER*8 BETAF
CHARACTER*8 K8BID, MOTCLE, MOGROU, MOD, NOMA, NOMNOE
CHARACTER*16 MOTFAC
CHARACTER*19 LISREL
CHARACTER*24 WK., GROUMA, NOEUMA
CHARACTER*19 LIGRMO
C ---------------------------------------------------------------------
C
MOTFAC = “LIAISON_DDL”
MOTCLE = “NODE”
MOGROU = “GROUP_NO”

LISREL = “&&CALIAI.RLLISTE”
CALL GETFAC (MOTFAC, NLIAI)
IF (NLIAI.EQ.0) RETURN
C
BETAC = (1.0D0,0.0D0)
C
C
CALL DISMOI (“F”, “TYPE_CHARGE”, LOAD, “LOAD”, IBID,
+ TYPCHA, IER)
CALL DISMOI (“F”, “NOM_MODELE”, LOAD, “LOAD”, IBID, MOD, IER)
CALL DISMOI (“F”, “NOM_MAILLA”, LOAD, “LOAD”, IBID, NOMA, IER)
C
NOEUMA = NOMA//“.NOMNOE”
GROUMA = NOMA//“.GROUPENO”
C
```

Data-processing handbook of Description

D5.03 booklet:

Index A

Code_Aster ®
Version 4

Titrate:
To introduce a new loading of the “kinematic” type

Date: 18/09/2003
Author (S):
J. PELLET

Key: D5.03.02
Page:
10/12

```
C -- CALCULATION OF NDIM1: NO. MAXIMUM TERMS Of a LIST
C GROUP_NO OR OF NODE
C ------------------------------------------------
NDIM1 = 0
C 10 I=1, NLIAI
CALL GETVID (MOTFAC, MOGROU, I, 1,0, K8BID, NENT)
NDIM1 = MAX (NDIM1, - NENT)
CALL GETVID (MOTFAC, MOTCLE, I, 1,0, K8BID, NENT)
NDIM1 = MAX (NDIM1, - NENT)
10 CONTINUOUS

WK. = “&&CALIAI. ” //MOTFAC
CALL WKVECT (WK., “V V K8”, NDIM1, JJJ)

C -- CALCULATION OF NDIM2 AND CHECKING OF THE NODES AND GROUP_NO
C NDIM2 EAST THE MAXIMUM NUMBER OF NODES IMPLY IN ONE
C LINEAR RELATION
C ----------------------------------------------------
NDIM2 = NDIM1
C 20 IOCC = 1, NLIAI
CALL GETVID (MOTFAC, MOGROU, IOCC, 1, NDIM1, ZK8 (JJJ), NGR)
NBGT = 0
C 30 IGR = 1, NGR
CALL JEEXIN (JEXNOM (GROUMA, ZK8 (JJJ+IGR-1)), IRET)
IF (IRET .EQ. 0) THEN
```

```
CALL UTMESS (“F”, MOTFAC, “THE GROUP” //ZK8 (JJJ+IGR-1)/
+ “DOES NOT FORM PART OF THE GRID: ” //NOMA)
ELSE
CALL JELIRA (JEXNOM (GROUMA, ZK8 (JJJ+IGR-1)), “LONMAX”,
+ N1, ")
NBGT = NBGT + N1
ENDIF
30 CONTINUOUS
NDIM2 = MAX (NDIM2, NBGT)

CALL GETVID (MOTFAC, MOTCLE, IOCC, 1, NDIM1, ZK8 (JJJ), NNO)
C 40 INO = 1, NNO
CALL JENONU (JEXNOM (NOEUMA, ZK8 (JJJ+INO-1)), IRET)
IF (IRET .EQ. 0) THEN
CALL UTMESS (“F”, MOTFAC, MOTCLE//"//ZK8 (JJJ+INO-1)/
+ “DOES NOT FORM PART OF THE GRID: ” //NOMA)
ENDIF
40 CONTINUOUS
20 CONTINUOUS
C
C
C -- ALLOWANCE OF TABLES OF WORK
C ------------------------------------
CALL WKVECT (“&&CALIAI.LISTE1”, “V V K8”, NDIM1, JLIST1)
CALL WKVECT (“&&CALIAI.LISTE2”, “V V K8”, NDIM2, JLIST2)
CALL WKVECT (“&&CALIAI.DDL”, “V V K8”, NDIM2, JDDL)
CALL WKVECT (“&&CALIAI.COEMUR”, “V V R”, NDIM2, JCMUR)
CALL WKVECT (“&&CALIAI.COEMUC”, “V V It, NDIM2, JCMUC)
CALL WKVECT (“&&CALIAI.DIRECT”, “V V R”, 3*NDIM2, JDIREC)
CALL WKVECT (“&&CALIAI.DIMENSION”, “V V I”, NDIM2, JDIME)
C
C
C
C BUCKLE ON THE LINEAR RELATIONS
C ----------------------------------
C 50 I = 1, NLIAI
CALL GETVR8 (MOTFAC, “COEF_MULT”, I, 1, NDIM1, ZR (JCMUR), N2)
CALL GETVTX (MOTFAC, “DDL”, I, 1, NDIM1, ZK8 (JDDL), N1)
C
```

```
C EXCEPTION: IF KEY WORD DDL DOES NOT EXIST IN AFFE_CHAR_THER,
C IT IS CONSIDERED THAT THE LINEAR RELATIONS CARRY
C ON THE DDL "TEMP"
IF (N1.EQ.0.AND.TYPCHA (1: 4) .EQ. "THER") THEN
N1 = N2
C 60 K=1, N1
ZK8 (JDDL-1+K) = "TEMP"
60 CONTINUOUS
ENDIF

IF (N1.NE.N2) THEN
CALL UTDEBM ("F", "CALIAI", "THE NUMBER OF DDLS APPEARING IN"
& // ' THE CONNECTION NR " IS NOT EQUAL TO THE NUMBER OF COEF_MULT: ')
CALL UTIMPI (", ", 1, N1)
CALL UTIMPI (", ", 1, N2)
CALL UTFINM ()
ENDIF

C -- RECOVERY OF the 2ND MEMBER:
C ------------------------------
IF (FONREE.EQ. "REAL") THEN
CALL GETVR8 (MOTFAC, "COEF_IMPO", I, 1, 1, BETA, NB)
ELSE IF (FONREE.EQ. "FONC") THEN
CALL GETVID (MOTFAC, "COEF_IMPO", I, 1, 1, BETAF, NB)
ELSE
CALL UTMESS ("F", "CALIAI", "CASE NOT ENVISAGED")
ENDIF
```

```
C
C
CALL GETVID (MOTFAC, “GROUP_NO”, I, 1,0, ZK8 (JLIST1), NG)
IF (NG .NE.0) THEN
C
C
C -- CASE OF GROUP_NO:
C --------------------
NG = - NG
CALL GETVID (MOTFAC, “GROUP_NO”, I, 1, NG, ZK8 (JLIST1), NR)
INDNOE = 0
C 80 J = 1, NG
CALL JEVEUO (JEXNOM (GROUMA, ZK8 (JLIST1-1+J)), “It, JGR0)
CALL JELIRA (JEXNOM (GROUMA, ZK8 (JLIST1-1+J)), “LONMAX”,
+ NR, '')
C 90 K = 1, NR
IN = ZI (JGR0-1+K)
INDNOE = INDNOE + 1
CALL JENUNO (JEXNUM (NOMA//“.NOMNOE”, IN), NOMNOE)
ZK8 (JLIST2+INDNOE-1) = NOMNOE
90 CONTINUOUS
80 CONTINUOUS
C
C ASSIGNMENT OF THE RELATION WITH THE LISTE_RELA:
C -------------------------------------------
CALL AFRELA (ZR (JCMUR), ZC (JCMUC), ZK8 (JDDL), ZK8 (JLIST2),
+ ZI (JDIME), ZR (JDIREC), INDNOE, BETA, BETAC, BETAF,
+ FONREE, FONREE, LISREL)
C
C
ELSE
C
C
C CASE OF NODE:
C -------------
CALL GETVID (MOTFAC, “NODE”, I, 1, 0, ZK8 (JLIST1), NBNO)
IF (NBNO .NE. 0) THEN
NBNO=-NBNO
CALL GETVID (MOTFAC, “NODE”, I, 1, NBNO, ZK8 (JLIST1), NR)
ENDIF
C
```

```
C ASSIGNMENT OF THE RELATION WITH THE LISTE_RELA:
C -------------------------------------------
CALL AFRELA (ZR (JCMUR), ZC (JCMUC), ZK8 (JDDL), ZK8 (JLIST1),
+ ZI (JDIME), ZR (JDIREC), NBNO, BETA, BETAC, BETAF,
+ FONREE, FONREE, LISREL)
ENDIF
C
50 CONTINUOUS
C
C
C -- ADDITION OF THE LISTE_RELA TO THE LOAD:
C --------------------------------------
CALL AFLRCH (LISREL, LOAD)
C
C
C -- SPARE:
C -----------
CALL JEDETR (WK.)
CALL JEDETR ("&&CALIAI.LISTE1")
CALL JEDETR ("&&CALIAI.LISTE2")
CALL JEDETR ("&&CALIAI.DDL")
CALL JEDETR ("&&CALIAI.COEMUR")
CALL JEDETR ("&&CALIAI.COEMUC")
CALL JEDETR ("&&CALIAI.DIRECT")
```

```
CALL JEDETR ("&&CALIAI.DIMENSION")
C
END
```

Data-processing handbook of Description
D5.03 booklet:

Index A

Code_Aster
Version 4

Titrate:
To introduce boundary conditions on new a ddl
Date: 05/05/97
Author (S):
***J. PELLET**, Mr. BONNAMY*

Key: D5.03.03
Page: 1/4

Mechanical department and Digital Models

Index:
With
GIST
Diffusion: Developers

Data-processing handbook of Description

D5.03 booklet:

D5.03.03 document

To introduce boundary conditions on new a ddl

.

Concretely, to introduce new a ddl and to allow to block a ddl (or to impose conditions of connection), there is no FORTRAN to modify but it is only necessary of to make evolve/move catalogues (catalogues elements, orders and sizes).

.

With regard to the elements of Lagrange, the finite element allowing to introduce conditions on a ddl must have as a name:

D_DEPL_R_nom_ddl
in mechanics
D_TEMP_R_nom_ddl
in thermics
D_PRES_C_nom_ddl
in accoustics

the name of a ddl is limited to 7 characters.

-
1ére stage for the developer is thus to create the new element of associated Lagrange with its new ddl while taking as a starting point the the existing elements (the only difference with these elements is the name of the ddl, it is almost a recopy).

-
2éme stage is the inscription with the catalogue of the sizes of this new ddl "to the end" of size DEPL_R in mechanics, of size TEMP_R in thermics, the size PRES_C in accoustics.

EDF
Direction of the Studies and Research

Electricity of France

Code_Aster
Version 4

Titrate:
To introduce boundary conditions on new a ddl
Date: 18/09/2003
Author (S):
***J. PELLET**, Mr. BONNAMY*

Key: D5.03.03
Page: 2/4

Important remark:

New the ddl must be registered following the existing ddl but right before the parameter of Lagrange named LAGR.

.
to put possibly up to date the catalogues of orders AFFE_CHAR_XXX

Order
Keys
Key word factor
AFFE_CHAR_CINE (_F)
[U4.25.05]
MECA_IMPO

THER_IMPO

ACOU_IMPO
AFFE_CHAR_MECA (_F)
[U4.25.01]
DDL_IMPO

FACE_IMPO
AFFE_CHAR_MECA_C

[U4.25.06]
DDL_IMPO
AFFE_CHAR_THER (_F)
[U4.25.02]
TEMP_IMPO
AFFE_CHAR_ACOU
[U4.25.04]
PRES_IMPO

.

Once these three operations carried out, new the ddl can be used under conditions with limits.

Example joining together the three stages to introduce new the ddl, GRX (warping of beams).

% ADDITION D_DEPL_R_GRX TYPELEM DATES…
ELEMENT D_DEPL_R_GRX % DDL BINDS IN MECHANICS (CMP GRX)
NET SEG3 001
CHART 4
MTEMPSR = INST_R E 1 IDEN 1 INST
MDDLMUR = DDLM_R E 1 IDEN 1 A1

MDDLIMR = DDLI_R E 1 IDEN 1 C
MDDLIMF = DDLI_F E 1 IDEN 1 C
CHAMNO 2
MGEOMER = GEOM_R NR 3 DIFF 3 X Y Z
First stage
0 0
MDEPLAR = DEPL_R NR 3 DIFF 1 GRX
1 LAGR 1 LAGR
CHAMELEM 0
VECTOR 1
MVECTUR = VDEP_R MDEPLAR
STAMP 1
MMATUUR = MDEP_R MDEPLAR MDEPLAR
OPTION 3
MECA_DDLM_R 0002
....
CONVERT 0

%
% MODIF SIZE COMPELEM DATES…
% CATALOGUES SIZES
%
SIZES _ 1 ière 93
%
Second stage
CASECT K8 1 NAME
....
DEPL_R R 9 DX DY DZ
DRX DRY DRZ GRX LAGR
....
LISTMA K16 2 TRANS LISTMA
%
SIZES _ 2 ième _MEMBRE 0
%
GRANDEURS_ELEMENTAIRES 12
%
....

END

Data-processing handbook of Description
D5.03 booklet:

% MODIF AFCHAME ORDERS DATE…
CHARGE = AFFE_CHAR_MECA (NUMERO__: 07
MODEL: “STATUTE: O, TYPE: CO (MODEL)”
....
ROTATION: “STATUTE: F, TYPE: L_R8”
DDL_IMPO: (STATUT__: “STATUTE: F, MIN: 01”
UN_PARMI__: ANY GROUP_NO NODE
ALL: “STATUTE: F, TYPE: TXM, IN: YES”
GROUP_NO: “STATUTE: F, TYPE: L_CO ()”
NODE: “STATUTE: F, TYPE: L_CO ()”
AU_MOINS_UN__: DX, DY, DZ, DRX, DRY, DRZ, GRX
DX: “STATUTE: F, TYPE: R8”
....
GRX: “STATUTE: F, TYPE: R8”
)
FACE_IMPO: (STATUT__: “STATUTE: F, MIN: 01”
UN_PARMI__: GROUP_MA, MESH
....
Third stage
NET: “STATUTE: F, TYPE: L_CO ()”
AU_MOINS_UN__: DX, DY, DZ, DRX, DRY, DRZ, GRX, DNOR, DTAN
EXCLUS__: DNOR, DX EXCLUS__: DNOR, DY EXCLUS__: DNOR, DZ
....
GRX: “STATUTE: F, TYPE: R8”
DNOR: “STATUTE: F, TYPE: R8”
DTAN: “STATUTE: F, TYPE: R8”

)
LIAISON_DDL: (STATUT__: “STATUTE: F, MIN: 01”
UN_PARMI__: GROUP_NO, NODE
GROUP_NO: “STATUTE: F, TYPE: L_CO ()”
NODE: “STATUTE: F, TYPE: L_CO ()”
DDL: “STATUTE: O, TYPE: L_TXM”
COEF_MULT: “STATUTE: O, TYPE: L_R8”
COEF_IMPO: “STATUTE: O, TYPE: R8”
)
FORCE_NODALE: (STATUT__: “STATUTE: F, MIN: 01”
...................)
....

Data-processing handbook of Description
D5.04 booklet: -
Document: D5.04.01

To introduce a new law of behavior

Summary:

The purpose of this document is to provide to the developers the principal elements necessary to establishment (or modification) of a law of behavior in Code_Aster. It describes the modifications with
to carry out on the catalogue of orders, as well as new routine FORTRAN to create to integrate it behavior either explicitly (method of RUNGE-KUTTA), or in an implicit way, in environment PLASTI (method of NEWTON) or in an optimized way.

Data-processing handbook of Description
D5.04 booklet: -
HT-66/05/003/A

Count

matters

1
Essential stages to introduce a new law of behavior

1.1
Writing of Doc. R

Initially, it is necessary to write the reference material of the law of behavior connecting in one not given the constraints to the deformations. One restricts here in the continuous mediums 2D and 3D. method remains valid for models with local nonlinear behavior in plane constraints, such that hulls, plates and pipes, or modelings with nonlinear behavior monodimensional, like the multifibre beams, the bars, the grids, via the method OF BORST [R5.03.03]. This method is applicable in a general way in small deformations, whatever it behavior, provided that the tangent operator of the new behavior is accessible and effective.

To solve the nonlinear problem posed on the structure, the document [R5.03.01] described the algorithm used in Aster.

With each iteration N of the method Newton [R5.03.01 § 2.2.2.2] one must calculate the nodal forces
R (one) = ***QT*** *N*
N
I
*I (options RAPH_MECA and FULL_MECA) constraints I being calculated from displacements linked via the relation of behavior. One must build too the tangent operator to calculate **K** nor (option FULL_MECA).*

*With the first iteration, one calculates **K** i-1.*

*The calculation of **K** i-1 (option RIGI_MECA_TANG), which is necessary to the phase of initialization*

*[R5.03.01 § 2.2.2.1] corresponds to the calculation of the tangent operator deduced from the problem of speed
below.*

*This operator is not identical to that which is used to calculate **K** nor by option FULL_MECA, to
run of the iterations of Newton. Indeed, this last operator is tangent with the problem discretized of
implicit way.*

***1.2
Modification of the catalogue of DEFI_MATERIAU***

*The goal of DEFI_MATERIAU is to introduce parameters of behavior. These parameters
can be common to several relations of behavior.*

*It is possibly necessary to add in the catalogue of DEFI_MATERIAU a key word corresponding factor
with the type of behavior which one wants to model and under this key word factor, key words
representing the parameters of this type of behavior.*

Important remark:

*From a data-processing point of view, the key words factors must be of K10 (chain of
characters limited to 10 characters), and the key words under unclaimed are limited to 8 characters.
In practice, that means that if key word is longer, only the first 8 characters
will be used. It thus has a collision risk with other key words having the 8 first
joint characters.*

1.3

Modification of the catalogues of STAT_NON_LINE and DYNA_NON_LINE

In the catalogues of these orders, one gives under the key words factors COMP_INCR or COMP_ELAS, the name of the relation of behavior after the key word RELATION.

One gives also the number of variables intern associated with this relation after the name with this relation.

The name of this relation of behavior can be different from the name of the type of behavior given in DEFI_MATERIAU.

1.4
Writing of a routine relating to a point of integration of an element

The point of integration is a point of Gauss in the case of continuous mediums, or a point of integration in the thickness, for hulls, for example.

The arguments of entry are:

· the increment of total deflection,
· the tensor of constraints at the moment of preceding calculation,
· the variables intern at the moment of preceding calculation; for example P
, p, Xi,
· the option of calculation: 3 options must be calculated: "RIGI_MECA_TANG", "RAPH_MECA" and "FULL_MECA".

The arguments of exit are according to the option of calculation:

· the tensor of the constraints reactualized (RAPH_MECA and FULL_MECA),
· reactualized internal variables (RAPH_MECA and FULL_MECA),
· the coherent matrix of behavior tangent or of speed (FULL_MECA and RIGI_MECA_TANG).

Important remarks:

The tensors deformation, constraints at the previous moment, and increment of deformation, given in arguments of entry, are such as the components except diagonal (shearing for constraints, and distortion for the deformations, are multiplied by 2 before call to the routine of integration of the behavior. Consequently, components of shearing of tensor of constraints at exit must also be multiplied by same coefficient 2.

One describes here the integration of a new behavior under the assumption of the small deformations. The assumptions available in Code_Aster on the deformations are:

· SMALL: in this case of the tensors deformations are calculated linearly by report/ratio

with displacements, on the initial geometry (Assumption of the Small Disturbances:
HPP);
· PETIT_REAC: the deformations are calculated linearly starting from displacements
on the reactualized geometry. Nothing changes in the integration of the behavior;
· GREEN
: in this case the provided deformations are the deformations of
GREEN-LAGRANGE. Under the assumption of small deformations (but the large ones
displacements), the behavior is expressed in a way similar to the behavior
HPP, but connects this time the deformations of GREEN-LAGRANGE to the constraints of
PIOLA-KICHHOFF of 2nd species. The transformation of constraints PK2 into
forced of Cauchy is managed by the appealing routines of NMCOMP. [R5.03.22].
· SIMO_MIEHE: in this case the arguments of entry correspond to the gradient of
transformation F at the previous moment and of gradient of the transformation enters
configuration at the previous moment and the current configuration F
. But in this case it
is necessary to formulate the model of behavior in great transformations, and
one cannot use formulation HPP like previously any more [R5.03.21].

1.5 Connection of this routine in routine NMCOMP (inelastic behavior) or NMCPEL (elastic behavior)

Shunting is done according to the name of the relation which was given under the key word RELATION of
COMP_INCR or COMP_ELAS.

2 Modifications of the catalogues of orders

2.1 DEFI_MATERIAU

One introduces into the catalogue of order DEFI_MATERIAU a key word factor under which one goes to be able to give the parameters necessary to the description of the behavior of material.

This key word factor is K16, whose only 10 characters are significant.

Examples:

· to describe an elastic behavior, a user will employ the key word factor ELAS
in DEFI_MATERIAU: to subdue = DEFI_MATERIAU (ELAS =…),
· to describe an elastoplastic behaviour with linear work hardening, a user goes
to employ the key words factors ECRO_LINE and ELAS: to subdue = DEFI_MATERIAU
(ELAS =…, ECRO_LINE =…).

Under the key words factors defining the behavior of material, one gives the key words which go to correspond to the names of the parameters of the law and after which one gives the values of these parameters.

These key words are of K8.

The values of the parameters are either of the real numbers, or of the functions (thus of K8).

Examples:

· for an elastic material, one must give the Young modulus E and the naked Poisson's ratio.
One has as follows:
to subdue = DEFI_MATERIAU
(ELAS= _F (
E = yg, [R]

NAKED = naked, [R]

)

)

· for an elastoplastic material with linear work hardening, one must give the characteristics rubber bands and the linear curve of work hardening which is defined by the elastic limit SY and the slope
*
traction diagram D_SIGM_EPSI, i.e.
*
One has as follows:

to subdue = DEFI_MATERIAU
(
ELAS= _F (
E = yg,
[R]

Naked = naked,
[R]

)
ECRO_LINE
=
_F (
SY
=
sy,
[R]
D_SIGM_EPSI
=
dsde,
[R]

),

)

Data-processing handbook of Description
D5.04 booklet: -
HT-66/05/003/A

2.2
STAT_NON_LINE, DYNA_NON_LINE, DYNA_TRAN_EXPLI

The non-linear laws in the case of are placed. It is necessary to modify the catalogues of the orders STAT_NON_LINE and DYNA_NON_LINE by giving the name of the relation after the key word RELATION under
the key words factors COMP_INCR or COMP_ELAS.

Examples:

· in the case of a relation of behavior of elastoplasticity of von Mises with work hardening isotropic linear, one a:

statnl = STAT_NON_LINE (

MODEL = MOD,
CHAM_MATER
=
chmat,

COMP_INCR = _F (RELATION = "VMIS_ISOT_LINE"),

...)

· in the case of a relation of behavior of elastoplasticity of Von Mises with work hardening linear kinematics, one a:

statnl = STAT_NON_LINE (

MODEL = MOD,
CHAM_MATER
=
chmat,

COMP_INCR = _F (RELATION = "VMIS_CINE_LINE"),

...)

It should be noticed that these two relations use the same parameters of DEFI_MATERIAU but them behaviors are different and the numbers of the variables intern are different:

2 for VMIS_ISOT_LINE: p,
7 for VMIS_CINE_LINE: X,

·
p indicates the cumulated plastic formation,
·
X indicates the tensor of recall (it is symmetrical, it thus has 6 components),
·
indicate an indicator of plasticity in a given point:
- if
X = 1 the point is "plastic",
- if
= 0 it are not it.

3
Modifications of the routines

3.1
In which (S) routine (S) to intervene?

To carry out the calculation of the behavior in a point of integration [§1.4], i.e. the calculation of constraints and of the internal variables, and the calculation of the tangent behavior, three solutions are possible in Code_Aster:

· is to use the architecture of environment PLASTI. It is about a whole of routines (routines of shunting and routines utility) allowing to introduce a new model "with little expenses ", i.e. by defining some specific routines. On the other hand, PLASTI does not allow to obtain models optimized in time calculation. This environment is described in [§3.3]. This framework general is used in particular for the integration of the models of behavior of the monocrystals [R5.03.11];
· is, which is not advised to obtain a good convergence, but can be faster in a phase of test, to use the architecture of integration clarifies by the method of RUNGE-KUTTA [R5.03.14]. Indeed simplicity comes owing to the fact that only the equations differentials describing the evolution of the variables intern are to be programmed, since in it case it does not have there a tangent operator;
· is to create a complete routine of integration of the behavior, which by the means of the example of the other existing routines, often makes it possible to obtain powerful models (for example, by reducing the system to be solved with only one scalar equation, not linear). This process is described with [§3.2.2].

In version 8, it will be also possible to define a new behavior in the formalism of Zmat (module of behavior of the Zebulon code) via the coupling Aster-Zmat. The goal of this functionality is of prototyper new models, but not to use it on calculations of structures of important flying bridge, because time calculation is increased to a significant degree. The interface will be accessible only within the framework from the partnership School of the Mines of Paris EDF R & D.

3.2 Programming of law while passing by the routine of shunting NMCOMP or NMCPEL

3.2.1 Principle

***One places oneself in the case of the routine NMCOMP which makes the integration of the laws of behavior
incremental (thus relating to COMP_INCR).***

Routine NMCOMP is called on the level of the calculation of the elements, that is to say TE.

***In fact NMCOMP is not called directly by TE but by called routines themselves
by TE.***

These routines are:

***· NMPL2D and NMPL3D for the solid elements 2D and 3D in small deformations,
· NMGP2D and NMGP3D for the solid elements 2D and 3D in great deformations
(SIMO_MIEHE),
· DKQNLI and DKTNLI for elements DKQ and DKT,
· VDXNLR for the thick hulls 3D,
· TE0329 for the hulls 1D,
· TUFULL for the elements pipes.
· NMCO1D for the behaviors 1D of the elements BARS, multifibre Poutres, Grilles.***

***The calculations carried out on the level of NMCOMP relate to a given point of integration of an element
given.***

: 8/16

These calculations consist in determining at the current moment the tensor of the constraints, the internal variables
and behavior tangent stamps it.

The goal of the play is to write a routine making these calculations, this routine being called by NMCOMP.
disconnection towards this routine in NMCOMP is done starting from a test consisting in comparing variable COMPOR (1) with the name of the relation of behavior given under the key word RELATION of
COMP_INCR and treated in the routine.

If one defined a relation of name "LOUSE", i.e one wrote:

statnl = STAT_NON_LINE (

MODEL = MOD,
CHAM_MATER
=
chmat,

COMP_INCR = _F (RELATION = "LOUSE"),

...)

One will write a routine NMTOTO which will be called by NMCOMP in the following way:

IF (COMPOR (1) (1: 4) .EQ' TOTO') then CALL NMTOTO (…,…)

ELSE

...

ENDIF

Let us take NMTOTO like a generic routine to carry out the integration of a law of behavior.

The arguments at exit of NMTOTO will be:

Standard name

Significance
SIGP (6)
R
constraints at the current moment
VIP (NBVARI)
R
variables intern at the current moment
DSIDEP (6,6)
R
stamp behavior tangent. It is one
square matrix dimensioned "into hard" 6 X 6 for
continuous mediums 2D and 3D

The arguments in entry of NMTOTO will be:

Standard name
Significance
NDIM
I
Dimension of space (2 or 3, addresses material
coded).

· TYPMOD (1) is the type of modeling: 3D,
D_PLAN, AXIS or C_PLAN,
· TYPMOD (2) is equal to "INCO" for
incompressible elements.
COMPOR (3)
K16
Table of 3 K16 relating to relation of
behavior.

· COMPOR (1) is the name of the relation of
behavior,
· COMPOR (2) is the number of internal variables
by point of integration,
· COMPOR (3) is K16 indicating one
assumption on the deformations.
Data-processing handbook of Description
D5.04 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
7.4

crit (5)
R
Table of local criteria of convergence.

· crit (1): maximum iteration count with convergence,
· crit (2): type of tangent matrix at the moment t+dt,
· if crit (2) = 0, one has a formulation in speed and the matrix is symmetrical,
· if crit (2) = 1, one has a formulation incremental and the matrix can be not-symmetrical,
· crit (3) is the value of tolerance of convergence,
· crit (4) is the number of increments for local recutting of the step of time,
· if crit (4) = - 1, 0 or 1, it does not have there recutting,
· crit (5) is the type of local integration for law of behavior,
· if crit (5) =0, integration is Euler-implicit,
· if crit (5) = 1, one makes an integration of RUNGE_KUTTA.
instam
I
Moment of preceding calculation
instap
I
Moment of calculation
TM
R
Temperature at the moment of preceding calculation

TP
R
Temperature at the moment of calculation
TREF
R
Temperature of reference
EPSM (6)
R
Deformations at the moment of preceding calculation (see notice has).
LIFO (6)
R
Increment of deformation, i.e., it acts of B.U in HPP (see remark has).
SIGM (6)
R
Constraints at the moment of preceding calculation
VIM (NBVARI)
R
variables intern at the moment of preceding calculation; NBVARI is in entirety entered "into hard" the routine, clean with the relation of behavior and not one variable
option
K16
Option of calculation asked.
There are the choice between:

· RIGI_MECA_TANG: this option is useful at the time of prediction, internal variables and them constraints are not calculated, (see notice b),
· FULL_MECA: the tangent matrix is reactualized at each iteration and one updates them constraints and internal variables,
· RAPH_MECA: the matrix is not reactualized tangent; one updates the constraints and them internal variables.

Data-processing handbook of Description
D5.04 booklet: -
HT-66/05/003/A

Note:

a. As specified previously, the tensors deformation, constraints at the moment precedent, and increment of deformation, given in arguments of entry, are such as components except diagonal (shearing for the constraints, and distortion for the deformations, are multiplied by 2 before call to routine NMCOMP. This was bench to facilitate calculations of standards intervening in a certain number of nonlinear behaviors, depend on the second invariant of the tensors. It is thus necessary to take into account this characteristic in the routine of integration with to write.
Consequently, components of shearing of the tensor of constraints at exit must also be multiplied by same coefficient 2. This does not have in theory of consequence on the matrix of elasticity, nor on the tangent matrix.

B. The argument OPTION is important because it makes it possible to determine calculations to carry out.
In particular, option RIGI_MECA_TANG is intended to calculate only one matrix tangent of prediction, to build starting from DSIDEP. It is necessary to take guard in programming not to use in this case arguments SIGP and VIP, of which the place memory is not allocated for this option.

Routine NMTOTO will be organized in the following way:

subroutine NMTOTO (NDIM, IMATE, TYPMOD, COMPOR, crit, instam, instap, TM, TP, TREF, EPSM, LIFO, SIGM, VIM, OPTION, SIGP, VIP, DSIDEP)

· Lecture of the characteristics of material (elastic and different) and calculation of these characteristics at the moments instam and instap by using routine RCVALA.

For example, these characteristics can be E, v, E, S
T
y.
One thus will calculate E - v - E -, S
,
,
+
+
+
+
T
y (i.e at the moment instam) and E, v, E, S
T
y (i.e at the moment
instap).

· When one will handle the constraints and the deformations, one will not make loops of 1 with 6 but of the loops of 1 to NDIMSI.

NDIMSI = 4 for the 2D
NDIMSI = 6 for the 3D

· Calcul of the threshold (for the laws with thresholds).

· For options FULL_MECA and RAPH_MECA: calculation of the constraints and the internal variables.

· For options FULL_MECA and RIGI_MECA_TANG: calculation of the matrix of behavior

tangent
or

&

.

3.2.2 Example of a routine realization the integration of a law of behavior: NMCINE

NMCINE carries out the integration of a relation of behavior of elastoplasticity of von Mises with linear kinematic work hardening.

For the integration of this relation, one will refer to Doc. [R5.03.02].

The arguments of NMCINE appear among those of generic routine NMTOTO described in [&3.1].

One a:

subroutine NMCINE (NDIM, IMATE, COMPOR, CRIT, INSTAM, INSTAP, TM, TP, TREF, EPSM, LIFO, SIGM, VIMP, OPTION, SIG, VIP, DSIDEP).

This routine is organized in the following way:

· Calcul of the elastic characteristics of material with temperatures TM and TP.
For that, one uses routine RCVALA in the following way:

```
C
C READING OF THE ELASTIC CHARACTERISTICS OF THE MATERIAL (TIME AND +)
NOMRES (1) = ' E'
NOMRES (2) = ' NU'
NOMRES (3) = ' ALPHA4


CALL RCVALA (IMATE, "ELAS", 1, "TEMP", TM, 2, NOMRES, VALRES, CODRET, FB2)


CALL RCVALA (IMATE, "ELAS", 1, "TEMP", TM, 1,
+
```

NOMRES (3), VALRES (3)
=
0.D0

IF (CODRET (3) .NE. "OK") VALRES (3) = 0.D0
EM
=
VALRES (1)
NUM
=
VALRES (2)
DEUMUM
=
EM (1.D0+NUM)
TROIKM
=
EM (1.D0-2.D0*NUM0
ALPHAM
=
VALRES (3)

CALL RCVALA (IMATE, "ELAS", 1, "TEMP", TP, 2, NOMRES, VALRES, CODRET, FB2)

CALL RCVALA (IMATE, "ELAS", 1, "TEMP", TP, 1?
+
NOMRES (3), VALRES (3), CODRET (3), BL2)

IF (CODRET (3) .NE. "OK") VALRES (3) = 0.D0
E
=
VALRES (1)
NAKED
=
VALRES (2)
LAMBDA
=
E*NU/551.D0-2*NU) * (1.D0+NU))
DEUXMU
=
E (1.D0+NU)
ALPHAP
=
VALRES (3)

One rather uses the coefficients of Lamé and μ and the model of compressibility K.

· Calcul of the characteristics of work hardening AND, SY and C at the temperatures TM and TP; for AND and SY,
one uses routine RCVALA like previously:

C
C READING OF the CHARACTERISTICS Of WORK HARDENING
NOMRES (1) = ' D_SIGM_EPSI'
NOMRES (2) = ' SY'

CALL RCVALA (IMATE, "ECRO_LINE", 1, "TEMP", TM, 2,
+
NOMRES, VALRES, CODRET, FB2)
DSDEM=VALRES (1)
SIGYM=VALRES (2)
*CM=2.D0/3.D0*DSDEM/(1.DO-DSDEM/EM)*
NOMRES (1) = ' D_SIGM_EPSI'
NOMRES (2) = ' SY'
CALL RCVALA (IMATE, "ECRO_LINE", 1, "TEMP", TP, 2,
+
NOMRES, VALRES, CODRET, FB2)
DSDE=VALRES (1)
SIGY=VALRES (2)
C
=
*2.D0/3.D0*DSDE/(1.D0-DSDE/E)*

· Calcul of the constraint of test and its standard within the meaning of von Mises:

~
~-
~
S.E. = +
2

To affect the terms of ~
S.E. , one makes a loop of 1 to NDIMSI like one saw it with [§3.1]:

```
C
C CALCULATION OF THE ELASTIC CONSTRAINTS
D0
110
K=1,3
DEPSTH (K)

= LIFO (K) (ALPHA* (TP-TREF) - ALPHAM* (TM-FREF))
DEPSTH (K+3)

=
LIFO (K+3)
110 CONTINUOUS
EPSMO = (DEPSTH (1) + DEPSTH (2) + DEPSTH (3) /3/D0
C 115 K=1, NDIMSI

DEPSDV (K) = DEPSTH (K) EPSMO * KRON (K)
115 CONTINUOUS
C CALCULATION OF THE CRITERION OF VON MISES OF SIGEL
C ONE SEES HERE the INTEREST OF the COEF RACINE (2) ON
C SHEARINGS.
SIGMO = (SIGM (1) + SIGM (2) + SIGM (3) /3/D0
SIELEQ = 0.D0
C 114 K=1, NDIMSI

SIGDV (K) = SIGM (K) SIGMO*KRON (K)

SIGDV (K) = DEUXMU/DEUMUM*SIGDV (K)

SIGEL (K) = SIGDV (K) + DEUXMU*DEPSDV (K)
```

*SIELEQ = SIELEQ + (SIGEL (K) - C/CM*VIM (K))** 2*
114 CONTINUOUS
*SIGMO = TROISK/TROIKM*SIGMO*
*SIELEQ = SQRT (1.D5D0*SIELEQ)*

· Calcul of the threshold of plasticity

Threshold = ~
S.E. - sy
THRESHOLD = SIELEQ - SIGY

· For options RAPH_MECA and FULL_MECA, calculation of the constraints and the variables intern with
the current moment

-
if threshold < 0
One is in the elastic range and the increments of the variables intern are null:

C
C CALCULATION OF THE ELASTOPLASTIC CONSTRAINTS AND THE INTERNAL VARIABLES

IF (OPTION (1: 9) .EQ. "RAPH_MECA". GOLD.
+

OPTION (1: 9) .EQ. "FULL_MECA" THEN
IF
(SEUIL.LT.0.D0)
THEN
VIP (7)
=
0.D0
DP
=
0.D0
SIELEQ
=
1.D0
A1
+
0.D0
A2

+
0.D0

- if
threshold

One is in the elastoplastic field and one calculates the increments of the constraints and of internal variables.

ELSE
VIP (7)
=
1.D0
DP
=
THRESHOLD (1.5D0 (DEUXMU+C))*
A1
=
*(DEUXMU/(DEUXMU+c)) * (SEUIL/SIELEQ)*

*A2 = (C/(DEUXMU+c)) * (SEUIL/SIELEQ)*
ENDIF
PLASTI=VIP (7)

C 160 K = 1, NDIMSI

*SIGDV (K) = SIGEL (K) A1*SIGEL (K) - VIM (K) *C/CM)*
SIGP (K) =
SIGDV (K)
+
*(SIGMO+TROISK*EPSMO) *KRON (K)*
VIP (K)
=
*VIM (K) *C/CM*
+
A2 (SIGEL (K) - VIM (K) *C/CM)*
160 CONTINUOUS
ENDIF
Data-processing handbook of Description
D5.04 booklet: -

HT-66/05/003/A

· For options RIGI_MECA_TANG and FULL_MECA, calculation of the matrix of behavior tangent:

RIGI_MECA_TANG calculates
&

FULL_MECA calculates

But one shows in our case that & =
if p = 0, which corresponds well to
the use of RIGI_MECA_TANG at the time of the phase of prediction.

The matrix is thus calculated
.
This calculation is not the object of this document and is too long to be exposed clearly here [R5.03.02].

It is thus admitted that one has affected the square matrix dsidep with the values of
.
Note:

Particular case of the plane constraints.

To write the tangent matrix, the fact is used that, when one writes:

=

zz = 0

One deduces zz from it according to
xx,
yy and
xy, and this expression of zz is injected
in the other relations.

Therefore in the treatment of the tangent matrices in the case of plane constraints,
one finds the instructions following:

C - 8.3 CORRECTION FOR THE PLANE CONSTRAINTS:

IF (CPLAN) THEN

C 136 K=1, NDIMSI
IF
(K.EQ.3)
GO
TO
136
C
137
L=1,
NDIMSI
IF
(L.EQ.3)
GO
TO
137
DSIDEP
(K, L) =DSIDEP (K, L)
+
-
*1.D0/DSIDEP (3,3) *DSIDEP (K, 3) *DSIDEP (3, L)*
137 CONTINUOUS
136 CONTINUOUS
ENDIF

3.3

Programming of law in environment PLASTI

3.3.1 Introduction

By means of computer, one passes by the routine NMCOMP which calls routine REDECE. PLASTI is called by REDECE.
Environment PLASTI is described in documentation [R5.03.10]: `Relation of behavior élasto-viscoplastic of the LMARC'.
Data-processing handbook of Description
D5.04 booklet: -
HT-66/05/003/A

3.3.2 Algorithm of resolution of the quasi-static problem

One seeks to check the balance of the structure at every moment. In incremental form, it is about one nonlinear problem whose variational formulation in the case of the small deformations can to put in the form:

To find U
such as:

((U + U
), T) () D = L (T)

kinematically

Drunk

= ud (T)
acceptable and T

where U indicates the field of displacement, Bu
ud
=
(T) corresponds to the boundary conditions in
displacement and L (T) are the loading at the moment T.

One is thus led to solve, for each increment of time T
:

Ft+t (U + U
T
) = 0 on the basis of a state with balance F = 0
0

U
being the increment of
U
solution on
T
, C being known

The diagram general adopted by Aster to solve this discretized total system is a method of Newton which is written, K being an indication of iteration:

F

D (U
K) = - F (U
K)
U
K

U
+1 = U
+ D

K
K
(U
K)

This diagram requires, starting from the estimate of displacements to the interation K, to calculate in each point of Gauss:

T T
+ which checks the law of behavior

MR. C
t+ T =

the operator of tangent behavior
T + T

F

T

with

= K =
K
K =
B
Data base
U
E
E
E

E

3.3.3 Environment
Plasti

It is thus necessary, with each total iteration and in each point of Gauss, to integrate them equations of the model for calculation T T
+ and to calculate the operator of tangent behavior.

An environment was created in Code_Aster with an aim of parameterizing the establishment of models elastoviscoplastic presenting a function threshold (field of elasticity).

This algorithm:

· manages the choices of integration elastic or (visco) plastic,
· proposes various routines to contribute to the resolution of the nonlinear system (local) formed by the equations of the model,
· updates the variables at the end of the increment,
· calls the routines user for the calculation of the operator of tangent behavior.

The step to establish a new model can be schematized in the following way:

Writing of the equations of the model of speed
y = F (y, T)
Choice of a diagram of integration
Writing of the system discretized R (y) = 0
Writing routines specific to the models:
· recovery of the data materials,
· evaluation of the function threshold,
· evaluation of the operator of tangent behavior
· routine for the resolution of the system R (y) = 0
(the algorithm proposes a method of Newton for one implicit nonlinear system)

+ Modification of the routines of shunting of the algorithm

3.3.4 Formalization of the equations to be solved

One has to solve the following equations:

· The law of behavior connecting the increment of the constraints to the increment of the total deflections
with internal variables (cumulated total deflection, center of the surface of load,…).

That is to say G (
p
, p
, v
ari.). = 0 éq 3.3.4-1

· laws of evolution of the various internal variables:

That is to say L (
p
, p
, v
ari,)
... = 0

éq 3.3.4-2
· the criterion of plasticity
F (, X, p
I
) = 0
That is to say - X - R (p
I
) éq 3.3.4-3
·
vari indicates the variables intern others that p and p,
·
Xi center of the surface of load is an example of component of vari,
· The increment of cumulated plastic deformation p is calculated with [éq 3.3.4-3].

The equation [éq 3.3.4-1] to 6 unknown factors (6 components of the symmetrical tensor of the constraints).

The equation [éq 3.3.4-3] to 1 unknown factor: p.
The number of unknown factors relating to the equation [éq 3.3.4-2] is equal to the component count of
variables intern others that the deformation (visco-) plastic and the deformation (visco-) plastic cumulated.

One has to solve, as indicated in [§3.2.2]:

F (y
) = 0

with yT = (vari p)

One solves this system by a method of Newton, that is to say:

F D [yk] = - F (yk)
yk+1

y

1
= y
K + D

K +
(y
K)

In addition, one has to calculate the tangent matrix.

It is considered that the system F (y
) = 0 are checked at the end of the increment. One disturbs F according to one small variation. One regards a variable and not as a parameter.

The system remains with balance and one thus checks that dF = 0.

F
F
F
F
That is to say D
+ D
+

+
= 0

D vari
D p

vari

p

One is thus led to use the same matrix jacobienne which was used for to calculate F (y
) = 0 bus
one can write:

F D (y)

= X
y

with yT = [vari p]

and X = [Hd 0

]0

By successive substitutions and eliminations, one obtains Kd = Hd

from where the required tangent operator

[- K1H]

=
T +t

The expression of K-1 is difficult to determine, also uses one a solvor LU to evaluate it.

Data-processing handbook of Description
D5.04 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
List utilities

Date:
01/12/05
Author (S):
J. Key PELLET
:
D6.00.01-B Page
: 1/16

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D6.00 booklet: -
Document: D6.00.01

List utility routines of Code_Aster

Summary:

We give in this document a list of approximately 500 utilities of Code_Aster. For each one of them, one
give a very short description of its function.

---

1 mode
of employment

This document must allow the developers new functionalities in Code_Aster to answer
the following question:

"does there exist in the code a routine which does that which I need? ".

The answer (if it is positive) will be the name of this routine as well as a short description of this one (2 lines of French). To use this routine with profit, other documents will have either to be consulted (D5 or D6), or if these routines are not documented (rather frequent case), to consult the text (and the comments) of sound source.

To be able to quickly traverse the list of these utilities, we classified them in "packages". One can sometimes to find the same routine in several packages. The packages are associated the types of the objects handled by the routines. For example, a routine of resolution of an assembled system will be found in packages: matr_asse and cham_no. The objects handled in these routines are not always Structures of Data Aster; one can also handle variables FORTRAN: scalars or tables. Us let us define in the table of the paragraph following the "types" of objects considered thereafter.

How to make live this document?
This document useful (I hope for it) for the community of the developers must live by them. The author their request thus to communicate (by mél if possible) their note: to him
Which are the utilities to be added to this document (new or forgotten)?
Which are the utilities whose function is not clear enough (or erroneous)?
Which are the utilities to remove list (removed code or to disadvise)?

2
List packages

PACKAGE
definition
BLAS/LAPACK
This "package" was removed. Libraries BLAS and LAPACK are now "presupposed" for Code_Aster.
CARA_ELEM SD cara_elem

FIELD
Field:
SD cham_no, SD cham_elem, SD chart
SD cham_no_s, SD cham_elem_s
CHARGE
SD char_meca, SD char_acou, SD char_ther, SD char_cine
SD liste_rela, SD liste_charge
COMPOR
laws of behavior
DEBUG
“debugging”
OTHERS
date, hour, measurement of the execution times,…
ELT_COQUE
finite elements of type “hull”
ELT_ISO
finite elements “isoparametric”
ELT_POUTRE
finite elements of type “beam” or “pipe”
ELT_TOUS
finite elements “general” (what is handled by routines)
ENVIMA
Constants depending on the object computer: IRIX, SOLARIS,…
FILE
files of input/output
FUNCTION
Function:
FUNCTION of FORTRAN 77
SD function, SD tablecloth
SIZE
physical size: names of the components, entireties coded,…
JEVEUX
Objects JEVEUX: simple objects or collections
INITEL
Initialization dse ELREFE.
LIGREL SD
ligrel
GRID SD
grid
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

MATERIAL
Characteristics of a material:
SD MATER, SD cham_mater, SD materc (coded material "used in the routines
)
MATR_ASSE
SD matr_asse, SD solvor
MESSAGE
alarm or error messages (UTMESS)
or messages of the type "INFORMATION" (key word INFORMATION of the orders)
MPLEIN
"full" matrix; i.e tables FORTRAN with 2 indices
NUME_DDL
Classification of the unknown factors of a linear system:
SD nume_ddl, SD prof_chno, SD nume_equa, SD storage
PREPOST
"pre" and "post" graphs treatment: GIBI, IDEAS
LOCATE
change of reference mark, rotation
RESUELEM
Matrices (or vectors) elementary:
SD resuelem, SD vect_elem, SD matr_elem
RESULT SD
result
RUPTURE
breaking process
SD
Structure of data of Code_Aster (i.e together of objects JEVEUX)
SUPERVISOR
communication of the orders with the supervisor: routines GETXXX

COUNT SD
count
TITRATE
titrate (or subtitles) associated with a structure of data
TYPE_F77
*Fortran77 types: REAL*8, COMPLEX*16, INTEGER, CHARACTER and vectors FORTRAN of these types*
VARI_COM SD vari_com

3
List utilities

>>>>>>>
CARA_ELEM
RECUDE
recovery of the diameter external of a tubular structure from the data provided by a concept of the cara_elem type

>>>>>>>
FIELD
ALCART
to allocate a SD chart [D6.10.01]
ALCHML
to create a "virgin" SD cham_elem
ASASVE
to assemble the elementary vectors coming from the loads
ASCAVC
to produce the second member of with loads kinematics
ASCOVA
to combine assembled vectors
ASSVEC
To assemble elementary vectors to make of it a second member (SD cham_no)
BARYCH
combination linaire of cham_no or cham_elem
CALCULATION
to make elementary calculations corresponding to an OPTION on the elements of a SD ligrel.
CALVCI
Calculation of the second member of to loads kinematics
CARCES

to transform a SD chart into a SD cham_elem_s
CELFPG
to recover the list of the names of the families of PG of a cham_elem (ELGA)
CELCES
to transform a SD cham_elem into SD cham_elem_s
CELVER
to check that a SD cham_elem has certain properties
CESCES
to change the discretization of a cham_elem_s (ELNO/CART/ELGA)
CESCNS
to transform a SD cham_elem_s into a SD cham_no_s
CESCRE
to create a SD cham_elem_s
CESEXI
to test the existence of a CMP of a point of a mesh of a SD cham_elem_s
CESRED
"to reduce" a SD cham_elem_s on a list of meshs and/or a list of CMPS.
CHLIGR
to convert a cham_elem into another cham_elem on another ligrel.
CHPCHD
to change the geometrical support of a field (NOEU/ELNO/ELGA/CART)
CHPNUA
to transform a cham_no into a SD cloud to be able to project it on another grid (method "NUAG_DEG_0/1")
CHSFUS
to amalgamate several SD cham_elem_s (or SD cham_no_s) (by addition or overload of CMPS)
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

: 4/16

CHSRAZ
to put at "undef" the CMPS of a cham_no_s or a cham_elem_s
CHSUT1
to modify the nouns of the size and the CMPS of a cham_no_s or a cham_elem_s
CNOCNS
to transform a SD cham_no into SD cham_no_s
CNOMAX
Calculate the max of the L2 standard of displacement DX DY DZ for a cham_no_depl_R.
CNSCES
to transform a SD cham_no_s into a SD cham_elem_s
CNSCNO
to transform a SD cham_no_s into SD cham_no
CNSCRE
to create a SD cham_no_s
CNSPRJ
to project a cham_no_s on another grid
CNSRED
"to reduce" a SD cham_no_s on a list of nodes and/or a list of CMPS.
COCHGD
combination linaire of cham_no or cham_elem
CRCHNO
creation of a SD cham_no
CRCNCT
to create a constant cham_no on all the nodes of a grid.
CSMBGG
calculation of the contribution to the second member of the imposed ddl when they are treated by elimination (SD char_cine)
IRCH19
to print a field (cham_no or cham_elem)
MAJOUR
update of a field of displacement following an increment by holding account the possible ones great rotations
MCCONL
to take account of the conditioning of Lagrange on the second member
MCMULT
carry out the product of a matrix by NR vectors (so complex)
MECACT
to create 1 SD constant chart [D6.10.01]
MECARA
recovery of the name of the fields in a cara_elem
MECHTE

to recover the field of temperature and the field of temperature of reference to one moment given
MEGEOM
recovery of the field of geometry in 1 model or 1 list of loads
MEMAX
to extract the "max" or the "min" from a CMP on a whole of elements of a cham_elem
MEMOY
to calculate the average (balanced) of a CMP on a whole of elements of a cham_elem
MESOMM
to make the sum (on the meshs of a grid) of the values of a cham_elem
MRCONL
to take account of the conditioning of the terms of Lagrange on the second member
MRMULT
carry out the product of a matrix by NR vectors (real case)
NMDORC
Treatment of key words factor BEHAVIOR/COMP_INCR/COMP_ELAS
NOCART
to note a couple (entitée_affectée, size) in a SD chart [D6.10.01]
NUACHP
to transform a SD cloud into a cham_no
PJ2DCO
to create a SD corresp_2_mailla to be able to use PJEFPR (case 2D)
PJ3DCO
to create a SD corresp_2_mailla to be able to use PJEFPR (case 3D)
PJ4DCO
to create a SD corresp_2_mailla to be able to use PJEFPR (case 2,5D)
PJ6DCO
to create a SD corresp_2_mailla to be able to use PJEFPR (case 1,5D)
PJEFPR
to project a cham_no on another grid (method "ELEM")
PRONUA
to project a SD cloud on another grid (method "NUAG_DEG_0/1")
SDCHGD
to change the type R/C of a cham_no or a cham_elem.
TECART
"to finish" a SD chart: to manage a "fine overload" of the affected CMPS [D6.10.01]
UTCH19
to extract a value (CMP) from a SD cham_elem
UTCHDL
to recover the number of a CMP in a SD cham_elem
UTNCMP
recover the number and the names of the CMPS of a field
VTCMBL

Linear combination of cham_no
VTCOPY
Copy values d'1 cham_no in another cham_no having possibly another classification.
VTCREA
Allowance of a cham_no
VTCREB
Allowance of a cham_no
VTCREM
Allowance of a cham_no
VTGPLD
add a field of displacement to a field of geometry: X2 = X1 + U
ZERLAG
to put at zero DDLS of Lagrange in a SD cham_no
ZEROSD
determine if a field is completely "virgin" (0.)

>>>>>>>
CHARGE
AFLRCH
to write in a load the linear relations of a SD liste_rela

AFRELA
to write a linear relation in a SD liste_rela
ASCAVC

to produce the second member of with loads kinematics
ASSCHC
to modify one matr_asse to take account of the elimination of the ddls constrained by SD char_cine
CALVCI
Calculation of the second member of to loads kinematics
COCHRE
check on a list of loads the presence of only one distributed load
CORICH
to manage a possible bond between a field and a load to be able to apply one later to him "FONC_MULT"
CSMBGG
calculation of the contribution to the second member of the imposed ddl when they are treated by elimination (SD char_cine)
MEDOM1
input and checking of the coherence of the mechanical data of the problem
MEDOME
input and checking of the coherence of the mechanical data of the problem
MEGEOM
recovery of the field of geometry in 1 model or 1 list of loads
NMDOME
Checking of the data of a mechanical non-linear problem

>>>>>>>
COMPOR
LCDEVI
calculate the diverter of a tensor of order 3
LCDIMA
calculate the difference of 2 square full matrices
LCDIVE
calculate the difference of 2 vectors of realities
LCEQVE
copy of a vector of realities
LCEQVN
copy of a vector of realities
LCHYDR
calculate the spherical part of a tensor
LCINMA
initialization of a square matrix
LCINVE
initialization of a real vector
LCINVN

initialization of a real vector
LCIV2E
calculation of the second invariant of a tensor of deformation
LCIV2S
calculation of the second invariant of a tensor of constraint
LCNRTE
normalizes of the second invariant of a tensor of deformation
LCNRTS
normalizes of the second invariant of a tensor of constraint
LCOPIL
operator of flexibility for a linear elastic behavior
LCOPLI
operator of rigidity for a linear elastic behavior
LCPRMM
product of 2 square matrices
LCPRMV
*product stamps square * vector*
LCPRSC
scalar product of 2 vectors
LCPRSM
multiply a square matrix by a scalar
LCPRSV
multiply a vector by a scalar
LCPRTE
tensorial product of 2 vectors
LCQEQV
test the equality of 2 vectors
LCSOMA
calculate the sum of 2 square matrices
LCSOVE
calculate the sum of 2 vectors
NICOMP
integration of the laws of nonlinear behavior for the incompressible elements into small deformations
NMDORC
Treatment of key words factor BEHAVIOR/COMP_INCR/COMP_ELAS

>>>>>>>
DEBUG

Note:
To compare the execution of 2 versions of the code giving of the different results (for example debug and nodebug), one can start impressions very useful and not too bulky

by positioning DBG=.TRUE in the routine calcul.f
IMPTOU
to print on listing the “signature” of all objects JEVEUX present on a basis
JEIMPM
print the segmentation of the memory [D6.02.01]
JEIMPR
impression of the repertory of one or more classes [D6.02.01]
JEPRAT
impression of the system objects or the attribute objects of collection [D6.02.01]
JEUNDF
to put at “undef” an object JEVEUX
JXVERI
test the coherence of the segmentation memory of JEVEUX [D6.02.01]
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

DBGOBJ
To print in files 5 numbers characterizing an object JEVEUX: contents + some attributes
UTTCPU
measure time CPU (to use and system) soup between 2 instructions [D6.01.03]

>>>>>>>
OTHERS
JJMMAA
writing of the name of the author and the creation date of this file
UTTCPU

measure time CPU (to use and system) soup between 2 instructions [D6.01.03]

>>>>>>>
ELT_COQUE
CQ3D2D
calculation of the co-ordinates 2D of a triangle or a quadrangle starting from its co-ordinates 3D passage in the reference mark of the plan of the triangle or the quadrangle with teta=angle between axis X and
dimension A1A2
DKQBF
stamp B at the point qsi eta for element DKQ
DKTBF
stamp B at the point qsi eta for element DKT
DSQBFA
stamp BFA at the point qsi eta for element DSQ
DSQBFB
stamp BFB at the point qsi eta for element DSQ
DSQCIS
matrices BCB and BCA at the point qsi eta for element DSQ
DSQDIS
stamp YEAR of shearing for element DSQ
DSTBFA
stamp BFA at the point qsi eta for the DST element
DSTBFB
stamp BFB at the point qsi eta for the DST element
DSTCIS
Matrices BCA and YEAR of shearing for the DST element
DXBSIG
*calculation of internal forces B*SIGMA to the nodes of the element due to the stress field SIGMA defined in the points of integration for the elements: DST, DKT, DSQ, DKQ and Q4G*
DXEFGT
efforts generalized of thermal origin at the points of integration for the COQUEes elements DST, DKT, DSQ, DKQ and Q4gG
DXEFRO
passage of the efforts or generalized deformations of the intrinsic reference mark of the element to the reference mark
room of COQUEe
DXMATE
calculation of the matrices of rigidity of inflection, membrane, coupling membrane-inflection and shearing for an isotropic or multi-layer material
DXMATH
calculation of the matrices of rigidity of inflection, membrane, coupling membrane-inflection and

shearing for an isotropic or multi-layer material
DXQBM
stamp BM membrane at the point qsi eta for elements DKQ and DSQ
DXQPGL
construction of the matrix of total passage --> local for a mesh triangle DKQ or DSQ
DXREPE
calculation of matrices T1VE and T2VE of passage of a matrix of the reference mark of the variety to the reference mark
element and T2VE reverse T2EV for all the options of post COQUEe treatment
DXROEP
Recovery density of material and thickness of the plate
DXSIRO
passage of the constraints or deformations of the intrinsic reference mark of the element to the local reference mark of
COQUEe
DXTBM
stamp BM out of membrane for the elements DKT and DST
DXTPGL
construction of the matrix of total passage --> local for a mesh triangle DKT or DST
GQUAD4
geometrical magnitudes on the QUAD4
GTRIA3
parameter setting of elements DKT (TRIA3)
JQUAD4
jacobien at a point on the QUAD4
Q4GBC
stamp BC at the point qsi eta for element Q4G

>>>>>>>
ELT_ISO
BMATMC
to calculate the matrix B connecting the first order deformations to displacements for a point
of integration
BSIGMC
to calculate the forces intern B*sigma with the nodes of the element
BTDBMC
to calculate the Bt*D*B product giving the elementary matrix of rigidity

CONNEC
initialization of the Iso-P2 elements
DFDM1D
calculation of derived from the functions of form compared to an element running in a point from

gauss for the elements 1D
DFDM2D
calculation of derived from the functions of form compared to an element running in a point from gauss for the elements 2D
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
List utilities

Date:
01/12/05
Author (S):
J.** Key **PELLET
:
D6.00.01-B Page
: 7/16

DFDM3D
calculation of derived from the functions of form compared to an element running in a point from gauss for the elements 3D
DMATMC
calculation of the matrix of HOOKE for the isoparametric elements for materials
isotropic, orthotropic and isotropic transverse
DPFCH3
calculation of derived from the functions of form compared to an element running in a point from gauss for the nonisoparametric elements 3D
EPSAMC
construction of the vector of the anelastic deformations defined in each point of integration has to leave the user data for the current element
EPSIMC
construction of the vector of the initial deformations defined in each point of integration has to leave
user data for the current element
EPSTMC
calculation of the thermal deformations for the isoparametric elements
EPSVMC
calculation of the mechanical deformations (i.e eps_totales - eps_thermic) at the points
of integration for the isoparametric elements
GAUSS
calculation of the weights and points of gauss
PGSEG2
calculation of the weights of gauss and the co-ordinates of the points of gauss for a segment, like

the derivative of the functions of form
SUBACV
calculation of the base counter-alternative (dimension 3)
SUMETR
calculation of the metric tensor (2x2) and sound jacobien
VFF2DN
calculate the normal and the weight of a point of Gauss of an element SEG in 2D
VFF3D
calculate the weight of a point of Gauss of an element SEG in 3D.
DFDMIP calculation of derived from the functions of form and the jacobien 2D, AXI, 3D

NMGEOM
calculation of the elements kinematics in a point of Gauss (possibly into large transformations)
NMMABU calculation of the matrix B (LIFO = B.DU)
2D, 3D
AXI and GREAT DEFORMATIONS remain has to make if need be
NMEPSI calculation of the deformations kinematics
2D, AXI, 3D, LARGE
NMEPSB calculation of the regularized deformations and their gradients
2D, 3D

>>>>>>>
ELT_POUTRE
CARCOU
to recover the geometry of the elements pipe (elbow)
DEELPO
recovery of the diameter external of an element of BEAM
FUN1
calculation of the surface or the constant of torsion equivalent of a right BEAM to section variable under the linear assumption of variation of the co-ordinates
FUN2
calculate the moment of inertia equivalent of a right BEAM to variable section under the assumption
of linear variation of the co-ordinates
GDFINT
for an element of BEAM in great displacement, the contribution of the point of gauss calculates number KP with the internal forces
GDJRG0
for an element of BEAM in great displacement, calculates, at the points of gauss, the jacobien

and rotation of the principal axes of inertia in position of reference stamps it, compared to general axes of co-ordinates
GDMB
for an element of BEAM in great displacement, the contribution of the displacement calculates of node with the matrix of deformation B at the point of gauss KP
JPD1FF
calculation of the functions of form of deformations generalized for the element BEAM 6 ddl with 3
points of gauss
JSD1FF
calculation of the functions of form of deformations generalized for the element BEAM 7 ddl with 3
points of gauss
POEFGC
calculation of the elementary vector complex generalized effort, for the elements of BEAM of Euler
and of Timoshenko
POEFGR
calculation of the elementary vector real generalized effort, for the elements of BEAM of Euler and of
Timoshenko
POMASS
calculate the matrix of mass of the elements of BEAM
PORIGI
calculate the matrix of rigidity of the elements of BEAM
POUEX7
Treatment of the eccentricity of the elements of BEAM
PTENCI
Calculate the kinetic energy for the elements of BEAM, discrete and bars
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

PTENPO
Calculate the deformation energy for the elements of BEAM, discrete and bars
PTKA01
calculate the matrix of stiffness of the element of right BEAM with constant section
PTKA02
calculate the matrix of stiffness of the element of right BEAM with variable section
PTKA10
calculate the matrix of stiffness of the curved element of BEAM
PTKA21
calculate the matrix of stiffness of the element of right BEAM with constant section to 7 ddl by node
PTMA01
calculate the matrix of mass of the element of right BEAM
PTMA10
calculate the matrix of mass of the curved element of BEAM

>>>>>>>
ELT_TOUS
TEATTR
to recover the value of an attribute associated with a TYPE_ELEMENT
LEATTR
to test if an attribute has a given value (on-layer of TEATTR)
FGEQUI
calculation of the equivalent sizes in constraint and deformation
JEVECH
To recover the address of the local field corresponding to a parameter
ELREF4
to recover the addresses of the tables containing the values of the functions of form (and theirs derived) on a family from points from integration (+ dimensions, matrix of passage Gauss - > Node)
ELREF5
to recover the addresses of the tables containing the co-ordinates and weight of the points of one family of points of integration.

NBDIM
turn over the dimension of a type of element given
NBNOEU
turn over the number of nodes associated with a type of element
NBNOSO
turn over the number of nodes tops associated with a type of element
PPGAN2
passage of the values at the points of gauss to the values with the nodes tops and the nodes mediums by average value
UTELVF
to recover the values of the functions of form on a family of points of integration when one is not in an elementary routine of calculation. (If not, ELREF4 should be used).
TECACH
to recover the characteristics of a champ_local: address, length,…
TECAEL
to recover the characteristics of a élément_fini: name of the associated mesh,…

>>>>>>>
ENVIMA
ISMAEM
possible maximum entirety [D6.01.01]
ISNNEM
entirety NaN [D6.01.01]
R8DEPI
*give the actual value 2*PI [D6.01.01]*
R8DGRD
Conversion degree/radian [D6.01.01]
R8GAEM
*range: numbers such as range ** 2 is representable out of machine [D6.01.01]*
R8MAEM
the largest reality [D6.01.01]
R8MIEM
the smallest reality [D6.01.01]
R8NNEM
reality NaN [D6.01.01]
R8PI
to give actual value pi. [D6.01.01]
R8PREM
relative precision of the real numbers [D6.01.01]
R8RDDG

Conversion radian/degree [D6.01.01]
R8VIDE
to give the value of an "impossible" reality (can be used to test if real were affected or not) [D6.01.01]

>>>>>>>
FILE
ULDEFI
defines logical association unit - local name (FILE) - name file (NOM_SYSTEME), made call to ULOPEN for the ASCII files
ULOPEN
carry out association, open FORTRAN and positioning for the ASCII files
ULCLOS
carry out the release and it "closed" for the ASCII files
ULPOSI
positions (NEW, OLD, SUSPEND) in the file of the ASCII type (in FORTRAN 77 it is not unfortunately not possible to position at the time of OPEN and extensions to the standard are not always allowed on all the platforms)
ULINIT
initialize the structure of data stored in the commun runs
ULIMPR
print the contents of the structure of data
ULISOP
return an entirety not no one if the logical unit were affected and if the associated file is open. local name is also returned.
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
List utilities

Date:
01/12/05
Author (S):
J. *Key* ***PELLET***

IUNIFI routine IUNIFI intended to recover the number of logical unit associated a local name (FILE) is preserved to ensure compatibility, but rests now on
new structures of data.

>>>>>>>
FUNCTION

foot-note:
when one needs a "null" function (for example like function by defect in orders), one can always use the function "&FOZERO" created by routine debut.f and thus available at any moment.

FIINTE
to evaluate a function interpreted in a point (i.e calculation of F (X))
FOATTR
overload the attributes of a concept of the function type
FOC1MA
to calculate the maximum ones of a concept of the function type
FOCRCH
recovery of a function in a structure tran_gene for a node of shock
FOCSTE
creation of a concept of constant the function type
FODERI
obtaining the value of the function and its derivative for a linear function per piece
FOEC2F
writing of the couples (parameter, result) of a concept of the function type
FOEC2N
writing of the values (parameter, function) of a TABLECLOTH
FOIMPR
impression of a concept of the function type on a file
FOINRI
interpolation for complex function has variable real
FOINT0
handing-over has zero of the common used by the routine foint2
FOINT2
interpolation for real function with real variable

FOINT3
interpolation for function complexes with real variable
FOINTE
to evaluate a function (i.e calculation of F (X, y, Z,…))
FOINTN
interpolation in the tablecloths
FOINTR
interpolation-extrapolation of a whole function
FOLOCX
research of the place of X in the ordered vector growing
FONBPA
to recover the list of the names of the parameters of a SD function
FOPRO1
to recover the prolongations and type of interpolation of a concept of the function type
FOZERO
to create a null function
REFODE
recovery of the value of a function and its derivative for a function of the temperature linear by piece
TBEXFO
to extract a function from a table by indicating 2 columns in opposite. [D6.06.01]
ZEROCO
resolution of F (X) =0 by a method of cord
ZEROF3
to seek the zero of a function: method of BRENT
ZEROFO
to seek the zero of a function

>>>>>>>

SIZE

DEC2PN
to decode a coded entirety bases 2 of them
DGMODE
to find the descriptor size associated with a local mode with chart, cham_no, or cham_elem, in form “iden”
DIGDEL
To recover the number of scalars representing the size for a local mode
EXISDG

to decode a coded entirety
IPOSDG
the position d'1 component in a descriptor size DG makes
IRCCMP
to find the number and the names of the components of a list presents in a size
ISCODE
to code an entirety coded on the first 30 powers of 2 (not of power 0)
ISDECO
to decode an entirety coded on the first 30 powers of 2 (not of decoding on power 0)
ISGECO
to manage the addition or the subtraction of both entirety coded out of the first 7 powers whole of 2
NBCMP
turn over the number D whole coded for a size
NBEC
turn over the number D whole coded for a size
SCALAI
turn over the type of a size: reality, entirety, character.
VERIGD
to check the coherence of a list of CMPS of a size

>>>>>>>
INITEL
ELRACA
various dimensions of a ELREFE
ELRAGA
description of the families of integration of a ELREFE
ELRFVF
functions of forms of a ELREFE
ELRFDF
derived from the functions of forms of a ELREFE

>>>>>>>
JEVEUX
CHLICI
to check that a character string is licit within the meaning of JEVEUX
COCOPG
To recopy a collection the JEVEUX in another largest
IMPTOU
to print on listing the "signature" of all objects JEVEUX present on a basis
JACOPO
to recopy a piece of object JEVEUX in another
JECREC
to create a collection JEVEUX [D6.02.01]
JECREO
to create an object simple JEVEUX [D6.02.01]
JECROC
to declare a new object in a collection (or in a repertory of name) [D6.02.01]
JEDEMA
décrémente the mark and releases the marked objects [D6.02.01]
JEDETC
destruction of a whole of objects JEVEUX [D6.02.01]
JEDETR
to destroy an object JEVEUX (simple or collection) [D6.02.01]
JEDISP
return in a table the lengths max available [D6.02.01]
JEDUPC
to duplicate a whole of objects JEVEUX [D6.02.01]
JEDUPO
to duplicate 1 object JEVEUX [D6.02.01]

JEECRA
assignment of an attribute of an object JEVEUX [D6.02.01]
JEEXIN
test the existence of an object JEVEUX [D6.02.01]
JEIMPA
to print the attributes of an object JEVEUX [D6.02.01]
JEIMPM
print the segmentation of the memory [D6.02.01]
JEIMPO
to print an object JEVEUX [D6.02.01]
JEIMPR
impression of the repertory of one or more classes [D6.02.01]
JELIBE
to release an object JEVEUX of the memory [D6.02.01]
JELIRA
to consult an attribute of an object JEVEUX [D6.02.01]
JELSTC
to find the names of the objects whose name contains a character string given, present on a basis JEVEUX [D6.02.01]
JEMARQ
increment the current mark [D6.02.01]
JENONU
return the number associates has a name (hash-coding JEVEUX) [D6.02.01]
JENUNO
return the name associates has a number (hash-coding JEVEUX) [D6.02.01]
JEPRAT
impression of the system objects or the attribute objects of collection [D6.02.01]
JERAZO
handing-over has zero of the associated segment of values A an object JEVEUX [D6.02.01]
JEUNDF
to put at "undef" an object JEVEUX
JEVEUO
to recover the address of an object JEVEUX [D6.02.01]
JEVEUT
recover an object in memory in a permanent way (mark = -1) [D6.02.01]
JEXATR
recovery cumulated lengths of the objects of a contiguous collection [D6.02.01]
JEXNOM
function of access to the objects of the named collections (or of the pointers of names) [D6.02.01]
JEXNUM

function of access to the objects of the numbered collections (or of the pointers of names) [D6.02.01]
JUVECA
enlarging of an object simple JEVEUX [D6.02.01]
JXVERI
test the coherence of the segmentation memory of JEVEUX [D6.02.01]
TBEXVE
to extract objet_jeveux containing a column from a SD counts. [D6.06.01]
TSTOBJ
To recover 5 numbers characterizing an object JEVEUX: contents + certain attributes
WKVECT
to create an object JEVEUX of the type vector [D6.02.01]

>>>>>>>
LIGREL
ADALIG
to reorganize the grels of a SD ligrel so that they have sizes adapted to the management of memory.
CALCULATION
to make elementary calculations corresponding to an OPTION on the elements of a SD ligrel.
CHLIGR
to convert a CHAM_ELEM into another CHAM_ELEM on another ligrel.
EXLIM1
Creation of a ligrel starting from a list of meshs
EXLIMA
Creation of a ligrel starting from a list of meshs

INITEL
to initialize the elements types present in the ligrel and to create objects .PRNM and/or .PRNS ligrel
NBELEM
turn over the number of elements of a GREL of a SD ligrel
NBGREL
turn over the number of GREL of a SD ligrel
NOLIGR
To add finite elements in a SD ligrel
TYPELE
To determine the type of the finite elements of a GREL of a SD ligrel

>>>>>>>
GRID
CESRED
“to reduce” a SD cham_elem_s on a list of meshs and/or a list of CMPS.
CHPNUA
to transform a cham_no into a SD cloud to be able to project it on another grid (method “NUAG_DEG_0/1”)
CNCINV
construction of the table of connectivity reverses of a SD grid
CNSPRJ
to project a cham_no_s on another grid
CNSRED
“to reduce” a SD cham_no_s on a list of nodes and/or a list of CMPS.
CRLINU
transform a list of names of nodes into a list of numbers of late meshs for
NOCART
EXLIM1
Creation of a LIGREL starting from a list of meshs
EXLIMA
Creation of a LIGREL starting from a list of meshs
EXMANO
extraction of the numbers of the meshs of the type SEG2 of which one of the ends is a node of

number given
GETVEM
to check the coherence of a list of entities of the grid given by the user
GMGNRE
to fill the list of node subjacent with the list with mesh
MEGEOM
recovery of the field of geometry in 1 model or 1 list of loads
NUACHP
to transform a SD cloud into a cham_no
PACOAP
to sort 2 lists of nodes so as to put in opposite the nodes of the 2 lists
PACOOR
to give the list of the co-ordinates of the nodes of a mesh
PADIST
to calculate the distance between 2 nodes
PANBNO
to calculate the number of nodes tops, nodes of edges, interior nodes of a mesh of a given type
PJ2DCO
to create a SD corresp_2_mailla to be able to use PJEFPR (case 2D)
PJ3DCO
to create a SD corresp_2_mailla to be able to use PJEFPR (case 3D)
PJ4DCO
to create a SD corresp_2_mailla to be able to use PJEFPR (case 2,5D)
PJEFPR
to project a cham_no on another grid (method "ELEM")
PRONUA
to project a SD cloud on another grid (method "NUAG_DEG_0/1")
RELIEM
to recover the list of the nodes (or the meshs) given behind key words.
UTNONO
return the 1st node (or the 1st mesh) of a GROUP_NO (or of a GROUP_MA)
VERIMA
to check the coherence of a list of entities of the grid given by the user
VTGPLD
add a field of displacement to a field of geometry: X2 = X1 + U

>>>>>>>
MATERIAL
MATELA

recovery of the values of E, NAKED, ALPHA in a material
RCADMA
recovery of the metallurgical components of a material
RCCOMA
obtaining the complete behavior of a material
RCCOME
obtaining the complete behavior of a material
RCFODE
obtaining the value of the function and its derivative for a function of the temperature
linear by piece
RCFONC
interpolation on a function of the type R (P)
RCMFMC
creation of the chart of material coded starting from the cham_mater
RCPARE
checking of the presence of a characteristic in a given behavior
RCTRAC
determination of the Young modulus and the function of work hardening starting from the curve of
traction of a given material
RCVADA
obtaining the value of the coefficients of material and their derivative compared to
temperature
RCVALA
obtaining the value of a real parameter of an element of a relation of behavior of one
material given, starting from an address of material coded by giving the list explicitly of
variables of order of which can depend the functions on material.

RCVALB
obtaining the value of a real parameter of an element of a relation of behavior of one material given, starting from the designation of the point of Gauss (and under-point).
RCVALC
obtaining a parameter complexes of an element of a relation of behavior of one material given
RCVALE
obtaining the value of a real parameter of an element of a relation of behavior of one material given, starting from a name of coded material

>>>>>>>
MATR_ASSE
ATA000
*construction of a SD matr_asse by calculation of the product: At * A where A is a matrix rectangular*
AJLAGR
add the lagrange in the matrix of mass from the matrix of stiffness
ASASMA
to assemble the elementary matrices of rigidity and Dirichlet
ASMATR
to assemble elementary matrices in an assembled matrix
ASSCHC
to modify one matr_asse to take account of the elimination of the ddls constrained by SD char_cine
COPMAT
copy of one matr_asse in a full matrix
CRESOL
To create a SD solvor
CRSOLV
To create a SD solvor by "defect" for method LDLT
EXTDIA
extraction of the diagonal of a matrix
FLEXIB
to calculate the matrix of associated residual flexibility has a cyclic problem with Mac interface Neal or none

JACOBI
resolution of the problem reduced to the eigenvalues by the decomposition of generalized Jacobi
MCCONL
to take account of the conditioning of Lagrange on the second member
MCMULT
carry out the product of a matrix by NR vectors (so complex)
MRCONL
to take account of the conditioning of the terms of Lagrange on the second member
MRMULT
carry out the product of a matrix by NR vectors (real case)
MTCOMB
linear combination of matrices
MTCONL
linear combination of the conditioning of Lagranges of the matrices
MTCOPY
recopy the values of the matrix in another matrix
MTDEFS
definition of the structure of a matrix
MTDSC2
recovery of the address of an object of a SD matr_asse
MTDSCR
allowance/desallocation of the descriptors of a SD matr_asse
MTEXIS
to check the existence of a matrix
PRECON
pre conditioning of one matr_asse for the use of RESO_GRAD
PRERES
to factorize one matr_asse (LDLT/MULT_FRONT) or to manufacture a matrix of pre conditioning (GCPC)
RESGRA
resolution by a method of gradient combines (GCPC) for a stored matrix "MORSE"
RESOUD
Resolution of a linear system
TLDLGG
"to factorize" a matrix (LDLT or MULT_FRONT)

>>>>>>>
MESSAGE
INFBAV
to put the mechanism INFORMATION in talkative mode [D6.04.01]

INFMAJ
update for the key word INFORMATION [D6.04.01]
INFMUE
to put the mechanism INFORMATION in dumb mode [D6.04.01]
INFNIV
Reference the level of impression and the logical unit of impression [D6.04.01]
UTDEBM
to begin an alarm or error message [D6.03.01]
UTFINM
to close an alarm or error message [D6.03.01]
UTIMPI
to print an entirety in alarm or error message [D6.03.01]
UTIMPK
to print a character string in alarm or error message [D6.03.01]
UTIMPR
to print a reality in alarm or error message [D6.03.01]
UTMESS
to print an alarm or error message [D6.03.01]
UTEXCP
As UTMESS but allows the lifting of an exception python in the command file.
UTDEXC
As UTDEBM but allows the lifting of an exception python in the command file.

>>>>>>>
MPLEIN
AMPPR
to add a real full matrix to a real full matrix
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

COPMAT
copy of a MATR_ASSE in a full matrix
MAVEC
*passage stamps full (m*m) > half-matrix column vector (N)*
MTGAUS
resolution by the method of Gauss of a linear system
PMAT
product of square matrices
PMAVEC
product stamps square full by a vector
PMPPR
product of two matrices stored full with taking into account with transposition by the intermediary of indicator
PRMAMA
product of rectangular full matrices
PROMAT
product of two full matrices
UTBTAB
*fact produces it full matrices: BT * A * B*
VECMA
transform a symmetrical matrix (triangular) into a square matrix

>>>>>>>
NUME_DDL
CHEDDL
to seek the row of a ddl starting from its type and the node
CRPRNO
creation and allowance of a structure prof_chno
NUMBER
To create a SD nume_ddl
POSDDL
give the number of the ddl associated with the node and its component
PTEDDL
to recover the numbers of equation corresponding to certain names of CMPS

PTEEQU
to create object .DEEQ of a SD prof_chno
RGNDAS
to find the name of the node and the component corresponding to a number of equation in one assembled system

>>>>>>>
PREPOST
ECRTES
writing of at the head of a dataset SUPERTAB
GICOOR
to create the collection which gives the permutation of the nodes of the meshs (ASTER--> GIBI)
INISTB
initialization of the names of meshs ASTER-TRIFOU according to graphic code I-DEAS 4.0
IRADHS
adherences IDEAS
IRGAGS
seek sizes IDEAS present in a size

>>>>>>>
LOCATE
ANGVX
Calculate the 2 nautical angles starting from a vector
ANTISY
calculate a matrix of rotation in R3
CANOR2
calculate the normal with a SEG2 (in 2D)
CANOR3
calculate the normal with a TRIA3 (in 3D)
CANORM
to calculate the normal with a mesh in a node with or without standardization of this vector
CHGREP
Change of reference mark: total room and vice versa
CHMALG
passage of the local reference mark to the total reference mark of the elementary matrices
CQ3D2D
calculation of the co-ordinates 2D of a triangle or a quadrangle starting from its co-ordinates 3D passage in the reference mark of the plan of the triangle or the quadrangle with téta=angle between axis X and

dimension A1A2
CTETGD
calculation of the matrix téta allowing to pass from the ddl of the right interface to those of the interface
left
GLOLOC
change of reference mark for a modal dynamic system
GRIROT
calculation of the matrix of passage of the reference mark of orthotropism towards the local reference mark of the element roasts
INTET0
to calculate the matrix of rotation for DX, DY, DZ, DRX, DRY and DRZ
LOCGLO
passage of the local reference mark to the total reference mark for a modal dynamic system
MAROTA
calculate the matrix of rotation corresponding to the vector rotation
MATPGL
Construction of the matrix of local total passage
MATRO2
calculation of the matrix rotation for a curved beam
MATROT
calculation of the matrix rotation for a right beam
MUDIRX
calculate the cosine directors of the matrix of passage of the reference mark of the element to the reference mark of
reference as well as the 3 normalized directions of the reference mark of the element
ORIEN2
orientation of a trihedron defined by 3 points
ORTREP
recovery of the user data defining the reference mark of orthotropism relating to the element running
PROJMG
passage BASE_MODALE - > reference mark physical
Handbook of Data-processing Description
D6.00 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

REFLTH
calculate the passage of the terms of conductivity of the reference mark of reference to the reference mark of the element
UTPSGL
total passage - > Local for a symmetrical elementary matrix (triangular)
UTPSLG
local passage - > Total for a symmetrical elementary matrix (triangular)
UTPVGL
total passage - > Local for a vector
UTPVLG
local passage - > Total for a vector

>>>>>>>
RESUELEM
ASASMA
to assemble the elementary matrices of rigidity and Dirichlet
ASASVE
to assemble the elementary vectors coming from the loads
ASMATR
to assemble elementary matrices in an assembled matrix
ASSVEC
To assemble elementary vectors to make of it a second member (SD cham_no)
CALCULATION
to make elementary calculations corresponding to an OPTION on the elements of a SD ligrel.
CESVAR
to create a SD cham_elem_s (DCEL_I) allowing to extend the cham_elem (VARI_R) calculated by the routine CALCULATION.

MEAMME
calculation of the elementary matrices of AMOR_MECA or RIGI_MECA_HYST
MEDIME
calculation of the elementary matrices of the elements of Lagrange (mechanical)
MEDITH
calculation of the elementary matrices of the elements of Lagrange (thermal)
MEMAME
calculation of the elementary matrices of MASS_MECA
MEMARE
to create and initiate object .REFE_RESU of the SD matr_elem (or SD vect_elem)
MERIME
calculation of the elementary matrices of RIGI_MECA (elastic)
MERIMO
calculation of the elementary matrices of the elements of the model and the elementary terms of residue (STAT_NON_LINE)
MERITH
calculation of the elementary matrices of RIGI_THER

>>>>>>>
RESULT
BMNBMD
to restore the modes number and of deformations of a SD base_modale
BMNODI
to recover the deformations of interface in a SD base_modale
CTETGD
calculation of the matrix téta allowing to pass from the ddl of the right interface to those of the interface
left
DCAPNO
to recover the address of a .VALE of a cham_no from its type and sequence number in
a made up result
DYARCH
seizure of the key word factor FILING (in a SD result)
EXTMOD
to extract from a concept mode_meca the deformation for one or more ddl. The lagranges are remove.
FOCRCH
recovery of a function in a structure tran_gene for a node of shock
IMBAMO
to print the results relating to the modal base

IRECRI
writing of a structure of data result on a file
IRPARA
impression of the parameters of a structure of data result
IRPARB
determination/checking of the parameters of a structure of data result
IRTITR
impression of the title of a SD result
NDARCH
filing of displacements, speeds, accelerations, forced
PROJMG
passage base_modale - > reference mark physical
RSADPA
recovery of addresses JEVEUX of the parameters of calculation or the variables of access of one structure of data result for the sequence number given and the list of variables of reference symbols [D6.05.01]
RSAGSD
redimensioning of a structure of data result [D6.05.01]
RSBARY
To interpolate a field between 2 moments of a SD result [D6.05.01]
RSCRSD
Creation of a structure of data result [D6.05.01]
RSEXCH
recovery of the name of the field of a structure of data result [D6.05.01]
RSEXIS
Existence of a structure of data result [D6.05.01]
RSEXPA
Existence of a parameter (or a variable of access) in a structure of data result [D6.05.01]
RSINCH
Interpolation of a field of a structure of data result [D6.05.01]
RSINDI
to find a reality (or a complex) in a parameter list of a SD result [D6.05.01]
RSINFO
impression (on listing) of the structure of a SD result [D6.05.01]
Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

Code_Aster ®
Version

RSNOCH
To note a field in the structure of data result [D6.05.01]
RSNOPA
recovery of the number of variables of access and the number of parameters as well as theirs names of a structure of data result [D6.05.01]
RSORAC
Recovery of the sequence numbers of a structure of data result starting from a variable access [D6.05.01]
RSRUSD
To destroy the fields of a structure of data result starting from a sequence number [D6.05.01]
RSUTNU
Recovery of the sequence numbers of a structure of data result starting from a variable access [D6.05.01]
RSUTN2
Like RSUTNU, but filter sequence numbers found by checking the existence of a field symbolic system on these sequence numbers.
RSUTN1
Like RSUTNU, but filter sequence numbers found by checking the existence of one parameter (or of a variable of access) on these sequence numbers.

>>>>>>>
RUPTURE
GABSCU
for each node of the bottom of crack one calculates his curvilinear X-coordinate
GDFONC

calculation of the gradients for the calculation of the rate of refund of energy in 2D
GDINOR
calculation of the direction of the field theta in the case or the normal in the plan of the lips appears in
the sd fond_fiss
GDIREC
for each node of the bottom of crack, one calculates the direction of the field theta
MEALPH
to create 1 field of propagation

>>>>>>>
SD
GNOMSD
To obtain a name validates for "hidden" SD.
COPISD
to duplicate a structure of data under another name [D6.07.05]
DETRSD
to destroy a structure of data [D6.07.05]
DISMOI
to put a question about a SD [D6.07.05]
EXISD
To test the existence of a SD [D6.07.05]
IMPRSD
to print (listing) a structure of data [D6.07.05]
UTIMSD
To print (dump) the contents of the objects of a SD [D6.07.05]

>>>>>>>
SUPERVISOR
GCNCON
to obtain the name of a SD (K8) which is not in conflict with the other names of SD
GETFAC
turn over the number of occurrences of a key word factor [D6.03.01]
GETLTX
turn over the length of the chains of a key word of type "text" [D6.03.01]
GETMFA
turn over the ième key word control ratio [D6.03.01]
GETMFM
turn over the nbval first words key of a key word factor with their type [D6.03.01]

GETMNB
turn over general information on an order [D6.03.01]
GETRES
turn over the name and the type of the result of an order [D6.03.01]
GETTCO
turn over the type of a SD user [D6.03.01]
GETVC8
turn over the list of the arguments of a key word of type “complexes” [D6.03.01]
GETVID
turn over the list of the arguments of a key word of “identifying” type [D6.03.01]
GETVIS
turn over the list of the arguments of a key word of “whole” type [D6.03.01]
GETVR8
turn over the list of the arguments of a key word of “real” type [D6.03.01]
GETVTX
turn over the list of the arguments of a key word of type “text” [D6.03.01]

>>>>>>>
COUNT
TBAJLI
To add a line to a SD counts [D6.06.01]
TBAJPA
To add parameters in a SD counts [D6.06.01]
TBCRSD
to create a SD counts [D6.06.01]
TBCRSD
to declare a new SD counts [D6.06.01]
TBEXFO
to extract a function from a SD counts by indicating 2 columns in opposite. [D6.06.01]
TBEXIP
Existence of a parameter in a SD counts [D6.06.01]
TBEXTB
To extract a under-table from a SD counts [D6.06.01]
TBEXVE
to extract objet_jeveux containing a column from a SD counts. [D6.06.01]
TBLIVA
Reading of a cell of a SD counts [D6.06.01]
TBNULI
Return the number of a line of a SD counts [D6.06.01]

Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

Code_Aster ®
Version
8.1

Titrate:
List utilities

Date:
01/12/05
Author (S):
J.** Key **PELLET
:
D6.00.01-B Page
: 16/16

>>>>>>>
TITRATE
IRTITR
impression of the title of a SD result
TITRE2
to create a subtitle
TITRATE
to create a title

>>>>>>>
TYPE_F77
AGGLOM
to create a table with 8 K24 to gather arguments of the type CHARACTER
ALMULR
product of N real numbers with test of the overflow and the underflow with office plurality of value
former or given zero have
CODE
writing an entirety in a character string
CODREE
writing a reality in a character string
DESAGG
fact opposite of AGGLOM
EXTRAC
extraction in a table containing of the vectors at successive moments of the vector
possibly interpolated at the moment wish
FOVERF
checking of the character growing of the values in a vector
FREQOM
calculate the frequency associated with the pulsation
GCNCO2
to obtain a character string by incrementing of a number
GGUBS
generator of numbers (pseudo) random uniformly left again between (0,1)
INDIIS
turn over the row of an entirety in a vector of entirety
INDIK8
turn over the row of K8 in a vector of K8
INDK16
turn over the row of K16 in a vector of K16

INDK24
turn over the row of K24 in a vector of K24
INDK32
turn over the row of K32 in a vector of K32
INDK80
turn over the row of K80 in a vector of K80
INITVE
to initialize a vector with zero
KNDIFF
to make the difference between 2 lists of character strings LK3 = LK1 - LK2
KNDOUB
to check that there are no doubled blooms in a list of character strings
KNINCL
to check that a list of character strings is included in another
KNINDI
turn over the row of K* in a vector of K*
LIIMPR
to print a list of entireties or realities
LSAME
test the equality of 2 character strings independently of their breakages
LXCAPS
a character string puts in capital letters
LXLGUT
turn over the length uses of a character string (without the white)
LXLIIS
decode a character string to read an entirety there
LXSCAN
decode a character string in words of various types: entirety, reality, text,…
NORMEV
a vector normalizes and turns over its initial standard
OMEGA2
calculate the pulsation associated with the frequency
ORDIS
rearrangement of a list of entireties by ascending order
ORDR8
to find the order ascending of a list of realities, not of modification about entry but determination of a pointer of order
PROVEC
calculation of the vector product of two vectors
PSCAL

calculation of the scalar product of two vectors
PSCVEC
calculation of the scalar product of two vectors
R8INIR
initialization of a real vector
SNLIRE
reading of a lexeme in a character string
UTLISI
utility of logical operations on the lists of entireties: union, intersection, singleton
UTREMT
scan for a word in a list of words
UTTRII
To sort a list of entireties
VDIFF
calculate the difference between 2 vectors: Z = X - Y
VECMA
transform a symmetrical matrix (triangular) into a square matrix

>>>>>>>

VARI_COM
VRCINS
manufacture of the field of variables of order at a given moment.
RCVARC
recovery of a variable of order on a point of Gauss (in a routine of calculation elementary)

Data-processing handbook of Description
D6.00 booklet: - HT-66/05/003/A

Code_Aster ®
Version
4
Titrate:
Management memory: JEVEUX
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE, J.R. LEVESQUE
Key:
D6.02.01-A
Page:
1/40
Organization:
EDF/IMA/MMN

Data-processing handbook of Description

D6.02 booklet: Management of structures of data

Document: D6.02.01

Management memory: JEVEUX

Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

Code_Aster ®
Version
4
Titrate:
Management memory: JEVEUX
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE, J.R. LEVESQUE
Key:
D6.02.01-A
Page:
2/40

Contents

---

1 Presentation general

1.1 Introduction

The management of the storage areas assigned to each structure of data constitutes a share

important of the organization of an application of scientific computation written in language FORTRAN. Standard FORTRAN 77 proposes a very reduced number of possibilities of establishment of these structures of
data:
· variable simple,
· fixed table of dimension,
· common zone of fixed size shared between several units of program.
These possibilities are insufficient to carry out a dynamic management of the resources.
In addition, for any application leading to quantities of data of a volume higher than that of the memory available on a given machine, the writer of the application must take in charge the overflows report and to program all the exchanges between the main memory and auxiliary auxiliary storage (files).
Software package JEVEUX makes it possible to deal with, directly in the source text FORTRAN several levels of operations:
· the standardization of types FORTRAN usable on machines different (entirety, reality, complex, logical, character). The criteria of portability are those definite with the software package ENVIMA (Manual of Development - D6.01.01 Booklet: the descriptor of environment machine),
· the creation of structures of data items (objects JEVEUX) of size defined in execution, shareable between several units of programs and car-documented,
· the assumption of responsibility of all the transfers between main memory and auxiliary auxiliary storage.

Note:

R1:
This document presents the whole of the functions of the Jeveux software package. Some these functions was not known of the programmer "lambda numbers" (those which must be called that by the supervisor [§4.20]).
It also should be noticed that the large majority of the uses of Jeveux in Aster relates to the management of "simple" objects (of the vectors). The reader in a hurry (or frightened by the whole of this document) will be able to be convinced that management of one vector by Jeveux is very simple by looking at the example of [§5.1].
R2:
The developer of Code_Aster which uses Jeveux must (in addition to this document having taken knowledge of another document: "Use of Jeveux" [D2.06.01].

1.2
Objects JEVEUX: simple object and collection
1.2.1 Basic concepts
Object JEVEUX:

Software package JEVEUX makes it possible to manage two types of structures of data accessible per name to
centre of the application. The use wanted that one calls these structures of data objects JEVEUX.
By definition, an object JEVEUX is:
· is a simple object,

· is a collection of objects.
An object JEVEUX is the unit consisted a descriptor and one or more segment (S) of values.

Descriptor:

A descriptor is a whole of information made up of a name and various attributes.
descriptor makes it possible to reach, starting from the name, (X) the segment (S) of values of object JEVEUX.
attributes describe the structure of object JEVEUX.

Name:

A name consists of a succession of alphanumerics limited to 24 characters (with the name is associated, by a table and a function of coding, an entirety giving access the object by associative addressing).

Attribute:

The attributes are the parameters (in a fixed number for the simple objects) making it possible to define and of
to describe the structure of data (variable, table). For example the type of the values and the length of table belong to the whole of the attributes.

Segment of values:

A segment of values is a continuous succession of words or bytes (according to the unit of addressing of machine) used to store values in main memory or auxiliary auxiliary storage (file).

Note:

Except exception (repertories of names), a segment of values is a succession of values of
even standard.

1.2.2 The object
simple

The simple object is the structure of source data of the software package. A simple object is consisted of descriptor and of only one segment of values. The number of attributes of a simple descriptor of object is fixed for a version of the software package.

Note:

The number of attributes is identical for all the simple objects.

Simple object = Name + Attributes + 1 Segment of values

The name of a simple object consists of 24 natures taken among the following:

· capital letters of A to Z and figures from 0 to 9 (standard FORTRAN);

· four special characters:

the white
« «
the point
« . »
the underlined white
"_"
and commercial one
"&"

The principal attributes are as follows:

· the **kind**: described the structure of the simple object

E
simple element (variable)
V
vector (table with an index)
NR
repertory of names (this kind will be defined in the following paragraph)

The kind makes it possible to associate the segment of values the traditional concepts of variable, of table FORTRAN or to define a more complex structure of data.

· the **type**: defines type FORTRAN of the scalar variables of the object

I

entirety
(type FORTRAN INTEGER)
R
reality
(type FORTRAN REAL*8)
C
complex
(type FORTRAN COMPLEX*16)
L
logic
(type FORTRAN LOGICAL)
K8, K16, K24,
K32,
or K80
characters
(type FORTRAN CHARACTER)

· the **length**: the length of the associated segment of values defines (the number of scalars).

1.2.3
collection
Collection:

A collection is a structure of data making it possible to share certain attributes of the whole of the objects composing it. It authorizes indifferently the access by name or number to objects of collection and to manage objects variable length.

Two types of collection are possible:

· one comprising a segment of values per object of collection: the dispersed collection,
· the other only one segment of values for all the objects of the collection: the collection contiguous.

Note:

A collection cannot be built a posteriori by regrouping of simple objects.
A collection is carried out starting from the descriptor of collection and one or more segments of values.

Descriptor of collection:

The descriptor of collection is the whole of information defining the collection: the name and them attributes.
The attributes common to all the objects of the collection are in particular: the kind, the type, length if it is constant, etc….
The attributes specific to each object of collection are managed separately. From the point of view of the user, an object of collection has all the attributes of a simple object, it could be used of same manner.

Collection = Name + Attributes + 1 or several Segments of values

The name of a collection makes up in a way identical to that of a simple object.
The specific attributes of a collection are:

· the access:
who defines the access mode to the objects of the collection: the access can be carried out by name (collection known as is then named), by number (the collection known as is numbered),
· storage:
who describes the organization in memory of the values associated with the objects with the collection, them
values can contiguous (only one segment of values), or be dispersed (one segment of values per object of collection).
All the objects of a contiguous collection are followed in the segment of values associated,
· the length:
it is constant if all the objects of the collection share the same length, it is variable if the objects are different lengths,

· the **maximum number** of objects of the collection.
Two attributes of collection can be shared between several collections. In the case of one collection of objects variable length, the vector lengths can be re-used for another collection containing the same number of objects. In the same way, two collections can share the names objects of collection. These objects are called pointer external.

Types of collection:

The numbered collection is made up of objects whose key of access is a variable entirety of 1 with numbers maximum objects of the collection. In a numbered collection all the objects preexist.
The named collection uses an ordinary person object of repertory kind of names which contains list names of objects of the collection managed by associative addressing. This repertory of names can to be defined by the user or to be created automatically. The names of the objects are inserted in the order chronological of creation of the names. One can reach an object of collection named by his name and/or by the sequence number of insertion.

---

2 Management of the objects

2.1 attributes

2.1.1 General

The attributes to which the user has access are, with two exceptions, nonmodifiable directly.
They are affected during the creation of the descriptor (for example the standard) then fixed until destruction
this descriptor. The system manages in addition to the attributes which evolve/move during the application (by
example the address memory of the segment of values).
One distinguishes three categories of attributes for objects JEVEUX:

· generic attributes: classify, kind, type,…
· attributes of collection.

· attributes of the objects of collection.
All the attributes are consultable constantly by the user.
In the continuation of the document, for each attribute, one indicates:
· the reference symbol by which one can consult it,
· type FORTRAN of each attribute with following conventions:

Code	Type	Declaration FORTRAN
I	entirety	INTEGER
R	reality	REAL*8
C	complex	COMPLEX*16
L	logic	LOGICAL
Ki	character	CHARACTER*i
K*	chain	CHARACTER* (*)

2.1.2 Generic attributes

Attributes affected by the user (nonmodifiable after creation of the descriptor):

CLAS	K1	classify fastening of the object at a base of data
GENR	K1	kind of the object: · E variable simple · V vector, · NR repertory of names of the type K8, K16 or K24.
TYPE	K1	type FORTRAN of the object: I, R, C, L or K

LTYP
I
length of the type:
· managed automatically for types I,
R, C and L,
· standardized for the characters with the values
8, 16, 24, 32 and 80
length of the object:
LONMAX
I
component count of the object of kind V
NOMMAX
I
a maximum number of names of the object of kind NR

Modifiable attributes constantly by the user:

These attributes are not essential to the operation of the software package. They allow the user to supplement the description of the objects JEVEUX which it handles.

LONUTI
I
component count of the object of kind V
used
DOCU
K4
four free characters left with the user

Attributes affected and managed by the system and consultable by the user:

These attributes are described as information, it can be consulted at the time of the development of one application (search for error,…).

NOMUTI
I
a number of names of the object of kind NR indeed
used
DATE
I
entirety comprising the month, the day, the year, the hour and
minutes of the completion of the work having carried out
last modification of the segment of values
(last unloading on the basis of data)
ORIG
K8
not used
IADM
I
address memory: relative position in the memory
JEVEUX of the segment of values
IADD
I
address disc
: the number contains of
the recording of the segment of values in
base data and the relative position in
the recording (stored in 2 entireties for
each object)
LONO
I
length in unit of addressing of the object (place
occupied by the segment of values measured in
length of the type)
USE
K16
use of an object JEVEUX: at the same time one contains
information on the use in writing or reading,
the state (déchargeable, removable,…) segment of
values when it is present in memory and them
trade marks at the time of the first request in
reading and in writing
The latter are used only in-house with JEVEUX.

2.1.3 Attributes of collection

Common attributes affected by the user or managed by the system:

The attributes common (CLAS, GENR, TYPE, LTYP, DOCU, etc) to the objects of collection are accessible in the same way that the attributes from a simple object and answer the same rules.

Attribute affected and managed by the system and consultable by the user:
NUTIOC
I
a number of objects actually created in
collection
Note:
This attribute is updated only at the time of the call of the routine of creation of object of collection JECROC.
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

Attributes affected by the user and nonmodifiable:

ACCESS
K*
type
access:
NAME
or NUMBERS (see
the assignment of this attribute by JECREC)
STORAGE
K*
Mode of storage of the values:
CONTIG: all the objects are contiguous in
segment of values
DISPERSE: the objects are filed with the fur and with
measure needs
MODELONG
K*
Mode of definition length of the objects of
collection:
CONSTANT: all the objects are of the same length
(attribute length of the object of reference)
VARIABLE: each object can have a length
different
L O NR T
I
overall length of a contiguous collection
NR MR. A X O C
I
numbers maximum objects of the collection

2.1.4 Attributes of the objects of collection

The rules of name of attribute of the objects of collection are the same ones as for the simple objects.

2.2
Associated data bases
With each class it is possible to associate a data base on disc. This possibility is not
by no means obligatory: one can work without writing and reading on disc (what can lead to
to saturate the memory available and to the stop with the application).
The number of classes on which one can work simultaneously is limited to 5. It is possible to open
or to constantly close a class during the application.
The opening of a preexistent data base makes it possible to recover the whole of information
stored on the latter. At the end of the work, it is possible to destroy a whole class.
A data base is a file of direct access; it is defined by:
· a name of class (K1),
· a local name of the file (K8),
· a block length,
· a number of blocks.
Note:
In Code_Aster, all the classes are associated bases [§3.3].

2.3
The access to the values
The user reaches the values of an object JEVEUX by a relative address in a definite table
by one of variables FORTRAN of reference of the suitable type. These variables of reference (ZI,
ZR,…) are placed in standardized commun runs (cf Usage of JEVEUX [D2.06.01] [§ 5.1]).
Note:
The returned address remains valid as long as the user did not declare that it did not use any more the object.
Cut memory of work defined for work
Achievable module
Zone managed by JEVEUX

System
Variables declared in
commun runs of reference
Address object of the K16 type
Address object of the type R
ZR
ZK16
LK16
LR
ZR (LR)
era
1 component of the values of the object of the real type
ZK16 (LK16 - 1 + I)
ième component of the values of the K16 object

2.4

Concepts of management of the memory

The access to objects JEVEUX is carried out by name. These names are stored in a simple object of kind repertory of names accessible only by the software package and managed by a method of addressing associative.

To order an application, the user must gather objects JEVEUX in one or more classes opened as a preliminary. With each class a catalogue is associated: it is the unit made up repertory of names and attributes of objects JEVEUX of the class.

The class one of the attributes is defined in the creation of the descriptor of an object JEVEUX. The research of name is carried out among all the classes open to this moment.

Note:

A name cannot thus appear more once among the unit of the open classes.

With each class, it is possible to associate a file of direct access (or bases data), which will contain at the end of the application, the descriptors and the segments of values of all objects JEVEUX class. This type of file gives access quickly the various recordings, an index describing the position of each one of them.

The exchanges between the main memory and the data bases associated with the classes are entirely taken charges some by the software package. When the working area in main memory is saturated, this one seeks, among the segments of values declared unutilised, the place necessary to establishment of a segment of values whose presence in memory was claimed. In end from application, all objects JEVEUX present in memory are discharged in the data base associated, as well as the catalogues (simple objects "system"), which allows the re-uses later.

The file of direct access will be valid only if the index were brought up to date during closing, it is thus essential to properly stop the application while passing by routine JEFINI.

Any segment of values in memory is framed by eight entireties (four front, four behind) allowing to manage the chaining of the various segments of values, to indicate their use (free, used in reading, déchargeable,….), to store the associated identifier, and to ensure a protection

---

partial with the overflows (the software package controls the integrity of the values contained by these entireties at the time of any request).

2.5 Requests of use of the objects

Software package JEVEUX provides various subroutines and functions making it possible to manage all them objects. The various following requests are distinguished:

· the creation of the descriptor of an object JEVEUX or an object of collection,

· the allowance of the object: search for place in memory for the segment of values associated, initialization of the values or second reading on the basis of data, finally supply with the unit of program appealing of a relative address of the segment of values compared to one variable of Z* reference. This allowance can be only formulated in reading or in read/write;
· consultation of the descriptor which makes it possible to recover the value of one dynamically attribute;
· impressions of the descriptor, the segment of values, the catalogue or the state of memory managed by JEVEUX;
· the release which puts an end to the allowance of the object and returns the segment of values unutilised;
· destruction of the descriptor and the segment of values of an object JEVEUX or an object of collection.

2.6
Characteristics of the collections

The dimensioning of a collection depends on its type. For each type, one describes it below mode of definition of the segment (S) of values associated.

2.6.1 Contiguous collection variable length

One defines:
· is the overall length of the segment of values by attribute LONT (overall length) of collection. One can in this case create the collection and define (without particular order) later on the length of each object before using it;
· is the length of whole or part of the objects of the collection by giving the vector of lengths managed by the user, or by bringing up to date the attribute length for each object, overall length of the segment of values will be calculated by the software package.

name
attributes
1
2
3
4
LONT
contiguous collection variable length

Note:
· it is advised to define the length of all the objects of a contiguous collection before first request of allowance in memory,
· the length of the segment of values LONT is fixed at the time of the first request of allowance in memory,
· a contiguous collection variable length cannot thus be increased after the first access,
· all the objects created in a contiguous collection are managed together,
· the associated segment of values can be used like a vector of values while being unaware of cutting into object of collection.

2.6.2 Contiguous collection constant length
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

Code_Aster ®
Version
4
Titrate:
Management memory: JEVEUX
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE, J.R. LEVESQUE
Key:
D6.02.01-A
Page:
15/40

One defines the length common to all the objects of collection; the length of the segment of values will be equal to the product constant length by the maximum number of objects of the collection.

2.6.3 Dispersed collection variable length

One defines:

· raising it of the number of objects,
· the length of each object, progressively with the needs and without particular, front order the allowance of the object.

Note:

The obstruction manpower of this type of collection is limited to the cumulated length of the objects actually created.

name
attributes
dispersed collection variable length

Note:

For a dispersed collection, one can use an object (of which the length was defined) front to have finished defining the whole of the objects.

2.6.4 Dispersed collection constant length

One defines:

· raising it of the number of objects,
· the length common to all the objects.

2.7
Release of the segments of values

A request of access in reading or writing on a segment of values associated with an object JEVEUX cause a loading in memory of the contents of the associated segment of values. When one manages a finished memory capacity, it arrives one moment when it is not possible any more to find of place to charge

a new object. It is then necessary to cause unloadings on disc or to eliminate from the segments
values become useless. This mechanism cannot be completely taken charges some by JEVEUX:
programmer must have indicated the objects concerned as a preliminary. But some care must
to be observed: several units of program can use the same address simultaneously
memory associated with an object. The setting in memory of a segment of values is accompanied by
the assignment of a whole mark which measures the level of depth in the calls of under
programs for each new request. The release can be carried out only if the level of call to
this moment is identical to the mark associated with the segment with values. One calls current mark it
level of call in progress. The placement of this mechanism imposes a rule of use of JEVEUX
very strict: any routine which calls JEVEUO or WKVECT must carry out a call to JEMARQ like
first achievable instruction and a call to JEDEMA like last achievable instruction.
routine JEMARQ makes it possible to bring up to date the value of the current mark, by incrementing it of 1, affected
with all the segments of values charged later on. Routine JEDEMA releases all the segments
values associated with the current mark then décrémente the latter.

Certain configurations tolerate exceptions to this rule: they are for example loops on
blocks of matrices. It is then necessary progressively to release the objects, it is the routine
JELIBE which is used in this case.
It is sometimes necessary to have permanently or all the length of a ordering of Code_Aster
certain objects (for the Supervisor of execution, the coded material), of the specific requests are
used, which affects a special mark.
The value of the current mark can be consulted at any moment using routine JEVEMA.

3 Environment of application

3.1 Print files

Impression of the descriptors, the segments of values, the catalogue, the state of the managed memory

by JEVEUX and error messages is directed towards logical units defined in beginning
of application (cf § 3.3).

3.2

Assistance with the development

The errors met and detected by the software package cause the impression of an error message, and in certain cases the data bases opened by the application close. The software package prints then the state of the memory which it manages and catalogues of all the open classes.

Note:

JEVEUX uses its own routines of impression of messages different from those from Code_Aster to avoid calls dynamically récurcifs.

All objects JEVEUX, safeguarded as a preliminary, are recoverable in a later execution.

3.3 Environment

Code_Aster

Three data bases are usable within the application of Code_Aster:

base "TOTAL"

associated the class "G"

base "VOLATILE"

associated the class "V"

base "LOCAL"

associated the class "It

Note:

A fourth base of name "BASEELEM" is used to store and read again the catalogue compiled elements.

The impressions of error messages are carried out by the utilities of impression of message with following conventions:

· programming error (misuse of JEVEUX): message of class "with
dead halt;

· error of exploitation (access to a non-existent object,…) : message of class "F" with attempt
of open data base closure.

Utilities of impression of error messages UT…. of Code_Aster can sometimes be employees in order to communicate with the Supervisor and to stop the code properly by validating them concepts created. Names of print units indicated at the time of the call of the routines of impression (JEIMPO, JEIMPR,…) are as follows:

· RESULT: results of calculation,

· MESSAGE: messages of control and error messages.

4

Provided subroutines and use

4.1 Preliminaries

In this chapter, one indicates by:

nom_os
a simple name of object

nom_co
a name of collection

nom_oc
a name of object of collection

num_oc
a number of object of collection

During the description of the arguments of the routines, one specifies the type of each argument and if it must
to be provided in entry (in) or to be recovered at exit (out). The modified arguments are announced by (VAr).

4.2
Functions of access for the objects of collection

The access to objects JEVEUX and the objects of collection is carried out by name. In the case of one named collection, it is necessary to pass, like argument, one of the two groups
information:

· the name of the collection and the name of the object of collection,
· the name of collection and the sequence number of the object of collection in the repertory.

To limit the number of routines user and to standardize the arguments of call, one introduced functions of the type FORTRAN CHARACTER *32. Those return a made up character string
name of collection and an interpretable suffix by the software package. They affect a commun run which ensure the transfer towards the routine called of the name of object of collection, the sequence number in repertory.

In order to synchronize well the call with the name of collection and the assignment of the commun run, it is
obligatory to call these functions in the arguments of the routines concerned.

Note:
These functions should be used only at the time of the call of a routine of request on one
object.

Functions of the type CHARACTER*32 (to be declared in any subroutine of call):
The three following functions must be called only like argument of the routines
acting on objects of collection
CHARACTER*32 FUNCTION JEXNOM (name, nom_oc)
in
name
K24
name of repertory or collection
in
nom_oc
K*
name of object of collection.
CHARACTER*32 FUNCTION JEXNUM (name, num_oc)
in
name
K24
name of repertory or collection
in
num_oc
I
number of object of collection.
Example of use: CALL JEVEUO (JEXNUM (NOMCO, NUMO), "It, JTAB)
In the case of a contiguous collection, a function of the type FORTRAN CHARACTER*32 is used for
to reach the vector cumulated lengths.
This vector of entireties contains n+1 values for a collection of N objects: component VI of it
vector provides the relative address of object I in the segment of values of the collection; the length
from this object is obtained by the Vi+1 difference - VI.
CHARACTER*32 FUNCTION JEXATR (nom_co, arg)
in
nom_co
K24
name of collection
in
arg
K8
"LONCUM"
Notation: one will indicate, from now, a name of object JEVEUX, or a call to the one of these
functions within the arguments of a routine by nom_o.
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

Code_Aster ®

4.3 The creation of the descriptors

The descriptor of a simple object or a collection is created in general by several calls.
first call makes it possible to create the descriptor by defining the name of object JEVEUX, and while indicating
values of the obligatory attributes (nonmodifiable later on):

· the class of fastening,
· the kind,
· the type.

In the case of a collection, one must define moreover:

· the access,
· storage,
· mode of definition length,
· the maximum number of objects.

Creation of the simple descriptor of object

SUBROUTINE JECREO (nom_os, lis_at)
in
nom_os
K24
simple name of object.
in
lis_at
K*
text defining STANDARD CLAS GENR and if need be LTYP,
when it is provided, the length of the type must be stuck to
type.
For example: “G V K16”. The first character must be not white, at least a white separates each
value of attribute; the capital letters are obligatory.

Creation of the descriptor of collection

SUBROUTINE JECREC (nom_co, lis_at, access, stock, modlon, nmaxoc)
in
nom_co

K24
name of collection
in
lis_at
K*
text defining STANDARD CLAS GENR and if need be LTYP,
when it is provided, the length of the type must be stuck to
type.
in
access
K*
“NO”: collection named with repertory interns (names of 8
characters to the maximum), objects of collection created by name
and accessible then by name or number
“NO nom_repertoire”: named collection using an object
simple of kind NR provided by the user and shareable enters
several collections, objects of collection created, by name,
are accessible by name or number (only one white must
to separate NO from nom_repertoire).
“NAKED”: sequentially numbered collection, objects of
collection are accessible only by number
in
stock
K*
mode of storage of the collection:
· “CONTIG” contiguous objects of collection in the segment
values,
· “DISPERSES” independent objects of collection the ones
others.
in
modlon
K*
mode of definition length of the objects of collection
· “CONSTANT”: all the objects of the collection are of
identical length;
· “VARIABLE”: each object of the collection can
to have a different length;
· “nom_vecteur_longuor”: each object can have
a different length, definite like component
of a simple object of kind V managed by the user and
shareable between several collections
in
nmaxoc

I
numbers maximum objects of the collection
Note:
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

It is strongly advised not to use the names “CONSTANT” and “VARIABLE” like
name of object JEVEUX.

Assignment of an attribute (after creation of the descriptor)

It is often necessary to supplement the assignment of the attributes of the descriptor (definition of length of a vector, overall length of a collection,…).

SUBROUTINE JEECRA (nom_o, nom_at, ival, cval)
in
nom_o
K24
name of object JEVEUX
in
nom_at
K*
name of attribute (cf annexes C: list attributes).
in
ival
I
whole value for an attribute of the whole type.
in
cval
K*
text for an attribute of the type CHARACTER.

Note:
The two arguments ival and cval are obligatory and cval must be different from the chain

vacuum.
CALL JEECRA (nom_o, “DOCU”, 0, “JPJP”)
CALL JEECRA (nom_o, “LONMAX”, 500, K8BID)

4.4 Insertion of a name in a repertory or creation of an object of collection

It is the same routine which is used to create a name of object of collection and to insert a name in a simple object of repertory kind.

SUBROUTINE JECROC (JEXNOM (nom_o, name))
in
nom_o
K24
simple name of object of repertory kind,
or name of named collection
in
name
K*
name of object to be inserted in the repertory,
or name of object of collection
SUBROUTINE JECROC (JEXNUM (nom_co, num_oc))
in
nom_co
K24
name of collection
in
num_oc
I
number of object of collection

Note:

For a numbered collection, this call brings up to date attribute NUTIOC.

4.5

The request of allowance

It is via routine JEVEUO that the user recovers a relative address by
report/ratio with a Z* variable measured in the length of the type of object JEVEUX. That allows him then to use the associated segment of values.
At the time of this call if the segment of values is not present in memory, the software package carries out one
search for place in main memory. If object JEVEUX does not have an image on disc, the segment values is initialized according to the type of the object (zero or white). In the contrary case one recovers on
disc preceding values.
One calls **use the** condition of access (“E” or “It) to the segment of values.
The segment of values remains allocated with the affected use as long as the user did not carry out new request and as long as the call to JEDEMA on the good level was not carried out. A request in

writing on an object allocated in reading will modify the use of it. A request in reading on an allocated object
in writing will not affect the use.
SUBROUTINE JEVEUO (nom_o, concealment, jtab)
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

---

in
nom_o
K24
access to the segment of values defined by:
nom_os: simple object,
nom_co: contiguous collection,
JEXNOM (nom_co, nom_oc): object of named collection,
JEXNUM (nom_co, num_oc): object of numbered collection,
JEXATR (nom_co, “LONCUM”) vector cumulated lengths
of contiguous collection.
in
concealment
K*1
condition of access or use of the segment of values,
“E” in read/write (allows to modify the segment of values),
“In reading (not of modification of the segment of values).
out
jtab
I
address first value of the object in table FORTRAN
associated the Z* variable corresponding to the type of the object
For a table, for example of vector kind and real type, one can use, indifferently, the object
JEVEUX, as of its allowance, in:

· reaching directly the ième value by:
ZR (jtab-1+I),
· passer by its address with an unspecified routine by:
ZR (jtab).

Note:

· It is dangerous to transmit the value of the address relating jtab directly to one routine;
· request JEVEUO is prohibited on the objects of repertory kind.

4.6

The request of permanent allowance during an order

JEVEUT makes it possible to allocate objects JEVEUX in their affecting a mark different from the mark current being worth -1. The objects will not be affected by the calls to JEDEMA but will have to be released explicitly by a call to JELIBZ.

The syntax of call is identical to that of JEVEUO.

4.7

The request of permanent allowance during the execution

This request is exclusively reserved to the Supervisor. JEVEUS makes it possible to allocate objects JEVEUX in their affecting a mark different from the current mark being worth -3. They will not be released
that at the end of the execution.

The syntax of call is identical to that of JEVEUO.

4.8

Surcouche of creation and allowance of an object of vector kind

WKVECT is a surcouche JEVEUX making it possible to allocate a vector (creation and presence in memory), it allows "the economy" of the call to three subroutines JEVEUX.

SUBROUTINE WKVECT (nom_os, lis_at, length, jtab)

in
nom_os
K24
simple name of object
in
lis_at
K*
text defining STANDARD CLAS GENR LTYP, GENR is worth here obligatorily V.
in
length
I
whole value associated the attribute LONMAX, length of vector.
out
jtab
I

address first value of the object in the table
FORTRAN associated with the Z* variable corresponding to the type with
vector.
The allowance of the object is carried out by defect as a first writer (cel=' E').
Note:
· no object of name nom_os must exist in the bases (under penalty of stop of
program JEVEUX).
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

Example of use:
Allowance on the basis of VOLATILE vector (V) of work (V) of REAL*8 (R) length
100; the name of the table is &&OP000.TAMPON.
CALL WKVECT (“&&OP000.TAMPON”, “V V R”, 100, LDEC)
4.9
Surcouche of enlarging of an object of vector kind
SUBROUTINE JUVECA (nom_os, length, jtab)
in
nom_os
K24
simple name of object
in
length
I
new value associated with the attribute LONMAX, length of
vector.
VAr
jtab
I
address first value of the object in the table
FORTRAN associated with the Z* variable corresponding to the type with
vector.
This routine makes it possible to modify the size of a simple object of vector kind. The new vector is affected with the same class as the old one.
Note:
· the object must be in memory (call to preliminary JEVEUO),
· attribute LONUTI is affected with the same value as LONMAX,
· the values are recopied,
· the value of the shift jtab is modified.
4.10 Recopies of objects JEVEUX
Recopy of an object JEVEUX (simple object, collection or object of collection):

SUBROUTINE JEDUPO (nom_cli, nom_in, nom_clo, nom_ou, dup_co)
in
nom_cli
K1
name of the class of the object to be recopied (``)
in
nom_in
K24
name of object JEVEUX to be recopied
in
nom_clo
K1
name of the class of the object receptacle (``)
in
nom_ou
K24
name of object JEVEUX receptacle
in
dup_co
L
used only for the collections if the classes are
different,
if =.TRUE. : one recopies the external pointers with one
collection,
if =.FALSE. : the external pointers are preserved

Note:

The receptacle object is destroyed if there exists before.

Recopy of a whole of objects JEVEUX:

Instead of working on only one object, it is possible to provide to certain routines a chain of characters being used to select the objects JEVEUX whose name contains this chain.

SUBROUTINE JEDUPC (nom_cli, souchi, ipos, nom_clo, soucho, dup_co)
in
nom_cli
K1
name of the class of the objects to be recopied
in
souchi
K*
character string to be identified in the names of objects
in
ipos
I
position in the names of the chain to be identified

in
nom_clo
K1
name of the class of the receptacles objects
in
soucho
K*
chain to substitute in names origins to obtain the name
object JEVEUX receptacle
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

Code_Aster ®
Version
4
Titrate:
Management memory: JEVEUX
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE, J.R. LEVESQUE
Key:
D6.02.01-A
Page:
22/40

in
dup_co
L
used only for the collections if the classes are
different,
if =.TRUE. : one recopies the external pointers with one
collection,
if =.FALSE. : the external pointers are preserved

4.11 Requests of release

These requests lead to a writing on disc (differed or not) according to the condition of access chosen at the time of the request of allowance and the rules of release [§2.3]:
one calls release the stop of the request in progress on an object JEVEUX:
· fine of a writing, with writing differed on disc,
· fine of reading, which does not imply any writing on disc.
The releases are carried out at the time of the call to JEDEMA. The calls to routine JELIBE are in rule general prohibited.

Note:

The access in reading makes it possible to avoid the writings on disc at the end of the use (saving of time).

Management of the access:

The access authorizations affected at the time of the first call to JEVEUO in reading or writing are managed using the following routine:

SUBROUTINE JEMARQ ()

JEMARQ increments the value of the current mark. Its call is obligatory in any unit of program charging in memory of objects JEVEUX.

Implicit release of all the objects in charge in a unit of programming

SUBROUTINE JEDEMA ()

JEDEMA releases all the affected objects of the current mark and décrémente the value of the mark current. Its call is obligatory in any unit of program charging in memory of the objects JEVEUX.

Release of an object JEVEUX or an object of dispersed collection:

SUBROUTINE JELIBE (nom_o)

in
nom_o
K24
name of object JEVEUX (object simple, collection or object of dispersed collection)

Its call is tolerated under certain conditions.

Release of the whole of objects JEVEUX:

SUBROUTINE JELIBZ ()

The call to this routine causes the release of the whole of the objects JEVEUX which were charged in memory by a call to JEVEUT (they are affected of a mark being worth -1).

4.12 Requests of existence

The requests of existence make it possible to check the existence of the descriptor of an object JEVEUX or one object of collection, but also the presence of a name in a repertory. They also allow to recover the sequence number of a name in a repertory of names.

Existence of an object JEVEUX or an object of collection:

SUBROUTINE JEEXIN (nom_o, iret)

in
nom_o
K24
name of object JEVEUX
out
iret
I
code return of the routine
iret = the 0 descriptor of the object does not exist,
iret the 0 descriptor exists

4.13 The passage of the sequence number to the name and screw poured

Obtaining the sequence number starting from the name:

SUBROUTINE JENONU (JEXNOM (nom_o, name), num)

in
nom_o
K24
name of collection or simple object of repertory kind
in
name
K*
name of object of collection or name
out
num
I
num = the 0 descriptor of the object does not exist,
num the 0 descriptor exists

Note:

· If the sought name does not appear in the repertory, the returned number is 0.
· This call is useless in the case of a numbered collection.

Obtaining the name starting from the sequence number:

SUBROUTINE JENUNO (JEXNUM (nom_o, num), name)

in
nom_o
K24
name of collection or simple object of repertory kind
in
num
I
sequence number of insertion

out
name
K
name = name of the object corresponding to the number num
Note:
· If the sought number does not appear in the repertory, the returned name is white.
· This call is impossible in the case of a numbered collection.

4.14 Destruction of the descriptors

It is possible to destroy the descriptor of an object JEVEUX and by extension the name and the attributes of an object of collection. This destruction is accompanied by the destruction of the segments by values associated present in memory and returns inaccessible those present on disc.

SUBROUTINE JEDETR (nom_o)
in
nom_o
K24
name of object JEVEUX (object simple, collection or object of
named collection)

The place released in the catalogue or the repertory of names becomes immediately reusable for another descriptor. The place, possibly used on disc, will be recoverable only by one operation of “retassage” of the file.

Note:
· It is not possible to destroy an object of numbered collection,
· for a named collection, the destruction of an object brings up to date attribute NUTIOC.

One can also use the following routine to destroy a whole of descriptors.

SUBROUTINE JEDETC (nom_cl, souch, ipos)
in
nom_cl

K1
name of the class or ``to treat all the classes
opened.
in
souch
K*
character string to be identified in the whole of the names
contents in the repertory of one or more classes.
in
ipos
I
position in the name of the chain to be identified.
One seeks all the descriptors whose name contains under chain souch with the position ipos
in the classes defined by the parameter nom_cl and one destroys the descriptors thus located.
It is preferable to use routine JEDETR when one explicitly knows the name of the objects with
to destroy, the call in a loop or a routine of low level can prove very expensive.

4.15 The recovery of the size of the storage areas available

SUBROUTINE JEDISP (nbp, lplace)
in
nbp
I
a number of sought positions.
in
lplace
V_I
cut in unit of addressing of the various zones available.
JEDISP returns, by descending order in the vector of entireties lplace, the size of the nbp more
great segments of liquid assets for an allowance (JEVEUO), at the moment of the call.
values obtained remain valid under the condition of the only use of the requests of allowance concerning
segments of values lower or equal lengths.
Thus lplace (1) returns the size of the largest object than one could allocate, lplace (2) that of
larger object than one could allocate after having used the zone corresponding to lplace (1).

Note:

The allowance of a collection can involve the loading in memory of the attributes objects and
to make thus null and void the values obtained by JEDISP.

4.16

consultations

Reading of an attribute of an object JEVEUX or an object of collection

SUBROUTINE JELIRA (nom_o, nom_at, ival, cval)
in
nom_o
K24
name of object JEVEUX

in
nom_at
K*
name of attribute (cf annexes C: list accessible attributes)
out
ival
I
whole value for an attribute of the whole type.
out
cval
K*
text for an attribute of the character type.
Note:

In addition to the attributes described with [§2], it is possible to recover with value XOUS the type of object JEVEUX associated with the descriptor, “S” simple object, “X” collection.

Key:
D6.02.01-A
Page:
26/40

Reading of the value of the current mark

SUBROUTINE JEVEMA (mark)
out
mark
I
value of the current mark

Research of the list of the names of descriptors present in a class

SUBROUTINE JELSTC (nom_cl, souch, ipos, nbmax, l_nom, nbval)
in
nom_cl
K1
name of the class or ``to treat all the classes
opened.
in
souch
K*
character string to be identified in the whole of the names
contents in the repertory of a class.
in
ipos
I
position in the name of the chain to be identified.
in
nbmax
I
dimension of the vector of K24 provided below
VAr
l_nom
V_K24
vector containing the list of the identifiers answering
search criterion
out
nbval
I
numbers maximum identifiers answering the criterion of
seek
nbval = - nbval if nbval > nbmax

4.17

impressions

Several routines make it possible to print the contents of an object JEVEUX or an object of collection, to print the attributes, to print out of the memory managed by JEVEUX (objects present, position, cut, etc), to print the catalogue of a class, or to consult the state of a data base.

Impression of the contents of an object JEVEUX or an object of collection

SUBROUTINE JEIMPO (cfic, nom_o, param, cmess)

in
cfic
K*
text defining the local name of the print file, with the centre
of Code_Aster one will use “MESSAGE” or “RESULT”
in
nom_o
K24
name of object JEVEUX
in
param
K*
not currently used (white ``obligatory)
in
cmess
K*
text appearing in comment with the impression

Impression of all the attributes of an object JEVEUX or an object of collection

SUBROUTINE JEIMPA (cfic, nom_o, cmess)

in
cfic
K*
text defining the local name of the print file, with the centre
of Code_Aster one will use “MESSAGE” or “RESULT”
in
nom_o
K24
name of object JEVEUX
in
cmess
K*
text appearing in comment with the impression

Impression of the catalogue

SUBROUTINE JEIMPR (cfic, nom_cl, cmess)

in
cfic
K*
text defining the local name of the print file, with the centre

of Code_Aster one will use “MESSAGE” or “RESULT”
in
nom_cl
K1
name of class or ``to treat all the classes
opened
in
cmess
K*
text appearing in comment with the impression

Impression of the state of the zone memory managed by JEVEUX
SUBROUTINE JEIMPM (cfic, cmess)
in
cfic
K*
text defining the local name of the print file, with the centre
of Code_Aster one will use “MESSAGE” or “RESULT”
in
cmess
K*
text appearing in comment with the impression
Routine JEIMPM makes it possible to publish the contents of zone memory managed at the time of the call. Here
in the order significance of the values appearing with the impression:
· name in summary of the class associated with the object,
-
G bases TOTAL,

-
V bases VOLATILE,
-
L bases LOCAL,
· identifying of collection, if the latter is worth 0, it is a simple object,
· identifying of simple object or number of object of collection,
· addresses memory of the segment of value,
· use of the segment of value, is worth U or X,
· length in unit of addressing (whole 8 bytes on CRAY) of the segment of values,
· statute of the segment of value, is worth D, A or X,
· name of the object, follow-up possibly of the number of object of collection.

Possible combinations of the statute and the use of a segment of values, like their significance are as follows:

U D:
segment of values used, in access in writing and possibly in reading. It will have to be discharged on disc.

U A:
segment of values used, in access in reading. It will not be discharged on disc.

X D:
unutilised segment of values, déchargeable: it will have to be discharged on disc. One request in writing or reading on the associated object returns its position directly without movement report and access disc. Its position can be recovered with all moment to place a new segment of values at the price of an access disc (origin: E U or L D).

X A:
segment of values unutilised, removable. A request in writing or reading on the object associated its position without movement report and access returns directly disc. Its position can be recovered constantly to place new segment of values without access disc (origin: E U, L U, L D or X D).

X X:
unutilised segment of values, free zone (origin: all others).

Example of impression obtained:

CL --NUM-- --KADM-- - US - LON UA - - ST ------------- NAME --------------
G
0
1
8 U
7
D
________GLOBALE ________$$CARA
G
0
17

22 U
3002
D
________GLOBALE ________$$PEPL
G
0
18
3031 U
2048
D
________GLOBALE ________$$INDX
....
L
23
1 548208 U
U
D
&&SYS CD.NUMERO
1
L
28
1 548217 X
48
D
&&SYS CD.NOM_EXTERNE
1
L
36
1 548272 X
48
D
&&SYS CD.DESCRIPTEURS
1
L
44
1 548327 X
115
D
&&SYS CD.POINTEURS
1
L
23
2 548449 X

2
D
&&SYS CD.NUMERO
2
L
28
2 548458 X
44
D
&&SYS CD.NOM_EXTERNE
2
L
36
2 548509 X
44
D
&&SYS CD.DESCRIPTEURS
2
L
44
2 548560 X
105
D
&&SYS CD.POINTEURS
2
G
0
29 548672 U
5010
D
&&SYS
RESULT.USER
V
310
1 553689 X
112896
With
&&MERIGI.ME001
.RESL
1
0
0 666592 X
1592

X
<<<<
FREE
>>>>
...
Impression of the list of objects JEVEUX present in a data base:
SUBROUTINE JEIMPD (cfic, nom_cl, cmess)
in
cfic
K*
text defining the local name of the print file, with the centre
of Code_Aster one will use “MESSAGE” or “RESULT”
in
nom_cl
K1
name of class or ``to treat all the classes
opened

---

in
cmess
K*
text appearing in comment with the impression
This routine publishes the catalogue of objects JEVEUX present on disc.
Impression of the attributes objects of collection and the objects managed by the system.
SUBROUTINE JEPRAT (cfic, name, nom_at, param, cmess)
in
cfic
K*

text defining the local name of the print file, with the centre
of Code_Aster one will use “MESSAGE” or “RESULT”
in
name
K24
name of object JEVEUX or name of class preceded by
character
$
in
nom_at
K8
name of the attribute or suffix of the system object to be printed.
nom_at takes its values among the following list:
for a collection
$$DESO, $$IADD, $$IADM, $$PEPL, $$NOM,
$$REEL, $$LONG, $$LONO, $$LUTI, $$NUM.
for a whole class
$$CARA, $$IADD, $$GENR, $$TYPE, $$ETAT,
$$DOCU, $$ORIG, $$RNOM, $$LTYP, $$LONG,
$$LONO, $$DATE, $$LUTI, $$HCOD, $$PEPL,
$$INDX, $$TLEC, $$TECR, $$IADM
in
param
K*
not currently used (white ``obligatory)
in
cmess
K*
text appearing in comment with the impression
Note:
This routine is a utility reserved for the assistance.
4.18
utilities
The following utilities were written to copy or for réinitialiser parts of objects simple or
objects of collection.
White or restoring following the type of object JEVEUX.
SUBROUTINE JERAZO (nom_o, N, i1)
in
nom_o
K24
name of object JEVEUX
in
N

I
a number of values with réinitialiser
in
I1
I
index in the vector of the first value with réinitialiser

4.19 A utility of debugging

A bad use of the address returned by routine JEVEUO can lead to a crushing
memory, and in particular with the destruction of the chaining of the segments of values. The programmer can
in this case to try to locate the routine provocative crushing by instrumenting its source code
with calls to JXVERI. This routine follows the chain of the entireties present in front of each segment
values and indicating the address of the following: at the time of a overflow this value is in general
crushed. Moreover, it is called upon the routine system HPCHECK checking the state of the allotted zones
during the initial dynamic allocation what makes it possible to detect an incursion out of the zone
licit memory.

SUBROUTINE JXVERI (cfic, cmess)
in
cfic
K*
text defining the local name of the print file, with the centre
of Code_Aster one will use "MESSAGE" or "RESULT"
in
cmess
K*
text appearing in comment with the impression

This routine transmits two messages, the first concerning the integrity or not of the chaining, the second
printing the code return of the routine system CRAY HPCHECK.

4.20 Routines of initialization used by the supervisor

The use of objects JEVEUX is not possible that after the initialization which requires to define:
· the maximum number of data bases, which one will be able to manage simultaneously,
· size of the zone memory managed by JEVEUX, which will be allocated dynamically.

This initialization is obligatorily carried out by the supervisor within the orders BEGINNING or CONTINUATION using the routine JEDEBU, which creates, automatically, all the system objects necessary:

SUBROUTINE JEDEBU (nbases, lzone, cmess, cvig, idebug)
in
nbases
I
maximum numbers of simultaneous data bases (5)
in
lzone
I
memory size allocated in unit of addressing
in
cmess
K*
local name of the print file of the error messages
in
cvig
K*
not used
in

idebug
I
used for an operation in debug mode
idebug = 0 normal operating mode
idebug = 1 engaged debug JEVEUX
When application JEVEUX is initialized, it is important to open the classes of objects on which one wish to work. To open a class, it is necessary to specify its characteristics: presence or not of an associated file, name of the class, name of the data base associated, characteristics file, etc… In mode DEBUG the operation of the manager of memory is modified, them unloadings on disc are not differed any more and memory occupied by a segment places it of values is positioned with an indefinite value (UNDEF on CRAY, NaN under Solaris). This mode of operation is used for déverminer code: the use of an address corresponding to one segment of values released causes a brutal stop.
SUBROUTINE JEINIF (stin, stout, nom_bas, nom_cl, nmax, nbloc, lbloc)
in
stin
K*
text defining the statute at the beginning of work:
· "DUMMY" not of associated file
· "BEGINNING" initialization or restoring of one
existing class
· "CONTINUES" recovery of the contents of one
existing class
in
stout
K*
text defining the statute at the end of the work:
· "SAVES"
safeguard on file in end
of application
· "DESTROYS" destruction of the class in end
of application
in
nom_bas
K*
local name of the data base (example: "TOTAL",
"VOLATILE")
in
nom_cl
K1
name of the associated class (example: "G", "V")
in
nmax

I
numbers maximum names of objects JEVEUX in the class
in
nbloc
I
numbers maximum recordings of the file of access
direct associated
in
lbloc
I
length of the recordings of the file of direct access
associated

Note:

A class can be opened or closed constantly, to amount of the number
maximum of classes gérables simultaneously.

To close a class, in the course of application:

SUBROUTINE JELIBF (cond, nom_cl)

in
cond
K*
text allowing to overload the definite value of stout
in JEINIF
· "SAVES" with immediate safeguard on file
· "DESTROYS" with immediate destruction
in
nom_cl
K*
name of the class to close (example: "G", "V")

Lastly, it is obligatory to call upon the routine of closure the application which carries out closing
of all the still open classes after safeguard of objects JEVEUX and the catalogues

---

present in memory and the application by instruction FORTRAN STOP stops. This routine is called with FINE centre of the order by the supervisor.

SUBROUTINE JEFINI (cond)

in
cond
K8
condition of fence:
· "NORMAL" safeguards according to the definite value of stout
in JEINIF
· "ERROR" safeguards opened classes, for
possible later analysis

Note:

· it is the only STOP usable in all the application to be able to re-use the bases of data.
· a user must call this routine to stop its application after detection of one error with cond = "ERROR",
· within Code_Aster a stop with the condition cond = "ERROR" does not authorize update of the TOTAL base in the repertory of the user, the concepts created not having been validated.

SUBROUTINE JETASS (nom_cl)

in
nom_cl
K1
name of the associated class (example: "G", "V")

This routine is intended to recover the recordings become unutilised following destruction of associated objects JEVEUX. That relates to only the large objects, i.e. those which require at least a recording of the file of direct access. The only effect is to reorder sequentially the recordings, the effective recovery of the place must then be carried out by recopying the beginning of the file of direct access, only the supervisor of Aster can carry out this action.

---

5 Examples of use

One will suppose, in this part, to have placed itself in an environment of application (for example that of ASTER) and to have opened the data bases associated with the classes G, V and L. One will not mention
that the calls to routines JEVEUX, the declarations and common FORTRAN are omitted.

5.1 Creation and re-use of a vector of reality

That is to say the following problem:
one wants to create (in routine SUBA) a vector of real containing co-ordinates X and Y of nno nodes. One will appelera this vector "COORDO_XY" and one will re-use it in routine SUBB.

SUBROUTINE
SUBA
(...)
...
COMMON/
RVARJE
/
ZR (1)
C
-
Beginning of the instructions:
(A)
CALL JEMARQ ()
C
-
Allowances of the vector on the "TOTAL" basis:
(b)

CALL WKVECT (“COORDO_XY”, “G V R”, 2*nno, JTAB)
C
-
“filling of the vector”
C 1 ino = 1, nno
ZR (JTAB - 1 + 2* (ino-1) +1) = X
ZR (JTAB - 1 + 2* (ino-1) +2) = Y
1
CONTINUOUS
(c)
CALL JEDEMA ()
END
SUBROUTINE
SUBB
(...)
...
COMMON/
RVARJE/
ZR (1)
C
-
Beginning of the instructions:
(D)
CALL JEMARQ ()
C
-
Recovery of the address of the vector (in reading):
(E)
CALL JEVEUD (“COORDO_XY”, “It, JTAB)
C
-
Recovery of the co-ordinates of the 27th node:
X27 =
ZR (JTAB - 1 + 2 X 26 + 1)
Y27 =
ZR (JTAB - 1 + 2 X 26 + 2)
...
(F)
CALL JEDEMA ()
END

Comments:

· lines (A), (c), (D), (E): routines JEMARQ/JEDEMA make it possible to release automatically objects at the end of the routines [§4.11],

· line (b):
-
“G V R”:
“G”: base “Total” (bases data of the user cf [§3.3])
“V”: vector,
“R”: reality,
-
2*nno: length of the vector
-
JTAB: address object in table ZR of common RVARJE
· line (E)
-
“It: access in “reading” of the object (cf [§4.5]).

5.2

Creation of a repertory of names and insertion of two names

```
CALL JECREO (“MES_NOMS”, “G NR K8”)
CALL JEECRA (“MES_NOMS”, “NOMMAX”, 25, K8BID)
C
CALL JECROC (JEXNOM (“MES_NOMS”, “NOM_1”))
CALL JECROC (JEXNOM (“MES_NOMS”, “NOM_5”))
C
CALL JELIRA (“MES_NOMS”, “NUTIOC”, IVAL, K8BID)
CALL JENONU (JEXNOM (“MES_NOMS”, “NOM_5”), NUM)
CALL JENUNO (JEXNUM (“MES_NOMS”, 1), NAME)
```

The call to JENONU returns value 2 in variable NUM, the call to JENUNO returns the value “NOM_1” in the variable NAME.

5.3

Dispersed collection of vectors of entireties

The objects of this collection are named in the repertory, managed by the user and created in the preceding example; they are variable length.

One adds a name with repertory "MES_NOMS" used by the collection and one defines the length of 2 vectors.

CALL JECREC ("MA_COLL", "G V I", "NO MES_NOMS", "DISPERSES", "VARIABLE", 25)
CALL JECROC (JEXNOM ("MA_COLL", "NOM_13"))
CALL JEECRA (JEXNOM ("MA_COLL", "NOM_13"), "LONMAX", 125, K8BID)
CALL JEECRA (JEXNOM ("MA_COLL", "NOM_1"), "LONMAX", 250, K8BID)

In another routine, one uses the object "NOM_13" which has been just created:

CALL JEVEUO (JEXNOM ("MA_COLL", "NOM_13"), "E", JTAB)
CALL JELIRA (JEXNOM ("MA_COLL", "NOM_13"), "LONMAX", LNOM13, CVAL)
C
C 10 K = 1, LNOM13
ZI (JTAB - 1 + K) = K
10
CONTINUOUS
..
CALL JEDETR (JEXNOM ("MA_COLL", "NOM_13"))

One will notice, in these two examples, the two manners of creating a name of object of collection in using the call to JECROC:

· on the repertory of names "MES_NOMS",
· on the name of collection, which uses the same repertory.

Attribute NUTIOC of the collection is updated at the same time as that of the simple object "MES_NOMS". The last instruction destroys the object of collection "NOM_13" and the name in repertory "MES_NOMS".

5.4

Contiguous collection of vectors of entireties

The objects of this collection are named in the repertory of names "MES_NOMS", already used for two preceding examples. They have a variable length defined by a vector lengths "LENGTHS" managed by the user.

In a first routine, one creates the collection and one adds an object "NOM_24"

CALL JECREC ("MA_COLL", "G V I", "NO MES_NOMS", "CONTIG", "LENGTHS", 25)
CALL JECROC (JEXNOM ("MA_COLL", "NOM_24"))

In another routine, one defines the length of objects 2 to 25, equal to the number of insertion of the object:

CALL JEVEUO ("LENGTHS", "E", JTAB)
C
C 10 I = 2, 25
ZI (JTAB - 1 + I) = I
10
CONTINUOUS

In another routine, one defines the length of object 1 (equalizes to 50), and one reaches all

ième
collection to define the 13
component of the first object:
CALL JEECRA (JEXNUM (“MA_COLL”, 1), “LONMAX”, 50, K8BID)
C
CALL JEVEUO (“MA_COLL”, “E”, JTABC)
K = 13
ZI (JTABC - 1 + K) = ...
ième
In another routine, one allocates the 2
object to define all its components:
CALL JEVEUO (JEXNUM (“MA_COLL”, 2), “E”, JTABOC)
CALL JELIRA (JEXNUM (“MA_COLL”, 2), “LONMAX”, L2, K8BID)

C 20 K = 1, L2
ZI (JTABOC - 1 + K) = ...
20 CONTINUOUS

---

ième
In another routine, one allocates the whole collection to define the first component of the 3
object, which one reaches by the vector cumulated lengths:
CALL JEVEUO (“MA_COLL”, “E”, JTABC)
C
CALL JEVEUO (JEXATR (“MA_COLL”, “LONCUM”), “E”, JTABCU)
C
IOBJ = 3
IAD = ZI (JTABCU - 1 + IOBJ)
ZI (JTABC - 1 + IAD) = ...
One will notice the two manners which make it possible to define the length of an object of collection:
· by using the call to JEECRA,
· by directly affecting the value in the vector lengths.
The first request with JEVEUO (within the last framework) carries out an access to the contiguous collection
broadly and thus the address of the first object returns, with load for the user to move by
report/ratio at this address to reach a particular object.
The second request gives access directly an object of collection and returns the address of this
object.
In the last example, one recovers the cumulated lengths of the objects of the collection, which
allows the user to thus reach any object, without multiplying requests JEVEUO.
Note:
In the three cases, all the objects of the contiguous collection are present in memory.

6

Appendix 1: list subroutines “user”

Functions of the type CHARACTER*32

JEXNOM (nom_co, nom_oc)

JEXNUM (nom_co, num_oc)

JEXATR (nom_co, “LONCUM”)

Creation of the descriptors

CALL JECREO

(nom_os, lis_at)

CALL JECREC

(nom_co, lis_at, access, stock, modlon, nmaxoc)

Assignment of an attribute

CALL JEECRA

(nom_o, nom_at, ival, cval)

Creation of a name in a repertory of names

CALL JECROC

(JEXNOM (nom_os, name))

Creation of an object of collection

CALL JECROC

(JEXNOM (nom_co, nom_oc))

Request of allowance

CALL JEVEUO

(nom_o, concealment, jtab)

CALL WKVECT

(nom_o, lis_at, length, jtab)

Enlarging of a vector

CALL JUVECA

(nom_os, length, jtab)

Recopy

CALL JEDUPO

(nom_cli, nom_in, nom_clo, nom_ou, dup_co)

CALL JEDUPC

(nom_cli, souchi, ipos, nom_clo, soucho, dup_co)

Requests of release and safeguard

CALL JEMARQ ()

CALL JEDEMA ()

CALL JELIBE (nom_o)

CALL JELIBZ ()

Request of existence

CALL JEEXIN (nom_o, iret)

To check the existence of a name and to obtain its sequence number

CALL JENONU (JEXNOM (nom_o, name), num)

To obtain the name associated with a sequence number
CALL JENUNO (JEXNUM (nom_o, num), name)
Destruction of the descriptors
CALL JEDETR (nom_o)
CALL JEDETC (nom_cl, souch, ipos)
Place available
CALL JEDISP (nbp, lplace)
Consultations
CALL JELIRA (nom_o, nom_at, ival, cval)
CALL JELSTC (nom_cl, souch, ipos, nbmax, l_nom, nbval)
CALL JEVEMA (mark)
Impressions
CALL JEIMPO (cfic, nom_o, param, cmess)
CALL JEIMPA (cfic, nom_o, cmess)
CALL JEIMPR (cfic, nom_cl, cmess)
CALL JEIMPM (cfic, cmess)
CALL JEIMPD (cfic, nom_cl, cmess)
CALL JEPRAT (cfic, name, nom_at, param, cmess)
Réinitialiser N values of a vector

CALL JERAZO (nomlu, nor, i1)

Debugging

CALL JXVERI (cfic, cmess)

Implementation and environment of exploitation (reserved to the Supervisor)

CALL JEDEBU (nbases, lzone, cmess, cvig, idebug)

CALL JEINIF (stin, stout, nom_base, nom_cl, nmax, nbloc, lbloc)

CALL JELIBF (cond, nom_cl)

CALL JEFINI (cond)

CALL JETASS (nom_cl)

---

7 Appendix 2: list accessible attributes

Following coding is used:

With
easily affected and nonmodifiable attribute (by
JECREO,
JECREC or JEECRA)
then
M
modifiable attribute

(by JEECRA)
C
consultable attribute only
(by JELIRA or JEIMPA)
bone
simple object
Co
collection
oc
object of collection
disper
dispersed collection
contig
contiguous collection
VAr
variable collection length
const
collection constant length
When the attribute is only accessible for a given kind, this last is indicated in first
column (cf [§ 2.1.2]). The white indicates that the attribute is not accessible.
In the case of the collections and of the objects of collection, one indicates, if it is necessary, the type of collection
concerned (contig & VAr for a contiguous collection variable length).
possible values
kind
bone
Co
oc
CLAS
With
With
C
GENR
E, V, or NR
With
With
C
TYPE *
I, R, C, L, K8,
K16, K24, K32,
K80
With
With

C
LTYP
C
C
C
LONMAX
V
With
With (const)
With (VAr)
NOMMAX
NR
With
LONUTI
V
M
M (const)
M (VAr)
NOMUTI
NR
C
C
DOCU
M
M
DATE
C
C
ORIG
C
C
IADM
C
C (contig)
C (disper)
IADD
C
C (contig)
C (disper)
LONO
C
C (contig)
C (disper)

USE
C
C (contig)
C (disper)
ACCESS
NO,
With
NO nom_uti,
NAKED **
STORAGE
CONTIG
DISPERSE
With
MODELONG
CONSTANT
VARIABLE
nom_uti ***
With
LONT
With (contig & VAr)
NMAXOC
With
NUTIOC
C
*
Routine JELIRA returns only K for the value of the STANDARD attribute for the type K8, K16, K24, K32 and K80. It then to consult attribute LTYP.
**
Routine JELIRA returns either “NO” for an internal pointer, or “NO name” where name is an external name of pointer of names.
***
Routine JELIRA returns either “CONSTANT”, or “VARIABLE”, or “VARIABLE name” where name is a name of external pointer length.

Author (S):
J.P. LEFEBVRE, J.R. LEVESQUE
Key:
D6.02.01-A
Page:
40/40
Intentionally white left page.
Data-processing handbook of Description
D6.02 booklet: Management of structures of data
HI-75/97/011 Ind A

Code_Aster ®
Version
4
Titrate:
Utilities of impression of messages
Date:
28/01/1999
Author (S):
J.P. LEFEBVRE, P. MIALON
Key:
D6.04.01
Page: 1/8
Mechanical department and Digital Models
Index:
With
Diffusion:
Developers

Data-processing handbook of Description
D6.04 booklet:
D6.04.01 document
Utilities of impression of messages

Summary

This document supplements the rules concerning the Inputs/outputs [D2.07.01], it describes the routines (package "UTMESS") allowing to carry out "short" writings of message.

EDF
Direction of the Studies and Research
Electricity of France
Project Codes of Mechanics

Contents

Data-processing handbook of Description
D6.04 booklet:
Index A

---

1 Types of messages

The logical units of impression are established at the beginning of completion of the work, by the supervisor, them
impressions emitted by the routines described in the following paragraphs will be carried out on these logical units without possibility of redirection.

The transmitted messages will be directed only according to their type:

Code
Type of message
Output files
F

fatal error message, the execution stops after various
ERROR
impressions. The concepts created during the execution are
MESSAGE
lost.
RESULT
E
error message, the execution continues (a little cf [§1.2]).
ERROR
MESSAGE
RESULT
S
error message, concepts created during the execution
ERROR
are validated by the supervisor, the execution stops.
MESSAGE
RESULT
With
message of alarm.
MESSAGE
RESULT
I
message of information of the supervisor (cf [§1.5]).
MESSAGE

1.1
Messages of the type F

This type of message is followed of a dead halt of the application, it is used within the framework of serious detection of error not being able to allow the normal continuation of an order Aster.
The emission of an error message F will cause the stop by CALL JEFINI (“ERROR”), this call start the memory print-out and repertories associated with the various bases,
moreover, one call to JXVERI is carried out to control the chaining of the segments of values.
The Supervisor adds in front of the transmitted message an impression containing the date and the number of
version.

Note:

·
the concepts created during the execution are not validated by the Supervisor, them objects JEVEUX y being attached being able to be partially filled or crushed, the base TOTAL is not safeguarded (in this case, the interface asterix does not recopy it towards the repertory user),
·
He is called upon a routine system (CALL ABORT ()) provocative increase of error:
of the names of the appealing routines and, according to the contents, impression type compilation

variables of these routines.
Data-processing handbook of Description
D6.04 booklet:
Index A

1.2
Messages of the type E

This type of message makes it possible to analyze a series of errors before the program stop. For example,
syntactic analysis of the command file by the Supervisor or the analysis of the file of grid
by order LIRE_MAILLAGE.
The transmitter of a message of the type E **will have to transmit a message of the type** F at the end of its analysis, with
less than one of the routines appealing emits it in its place (parameter of error of OPxxxx returned not no one with the supervisor)

1.3
Messages of the type S

This type of message causes a dead halt of the application with validation of the concepts created
during the execution and "clean" closing of the TOTAL base. Its employment is justified in
portions of programming instrumented to measure the past machine time (cf [D6.08.01]
Measure time CPU). It makes it possible to be secured against a stop system during an iterative process.
The developer must take some precautions all the same in order not to store fields
nonvalid, for example a not converged solution which could then imprudently be used
as initial solution.

1.4
Messages of the type A

The messages of alarm of the type A are to be prevented as long as that is possible. Indeed, it is not
recommended to worry the user of the code unnecessarily, and it is preferable not to fill it
file MESSAGE by superfluous information. The number of messages of alarm is limited
automatically with 5 identical **successive** messages.
It is recommended to the users who have messages of the type A "to repair" their file of

orders to do them disparaitre.

1.5

Messages of the type I

The messages of information are defined in [D2.07.01 §6].

Let us recall that:

·

they are ordered by **key word "IMPR"** of the orders, except those emitted by supervisor (who is "with the top" of the orders),

·

they are the **results** of the order, neither the **echo of the data** of the user, nor of weakened **alarms**,…

·

they belong to the **contractual** supply of Aster. One does not change them without into speaking in EDA and with the users.

These messages **should not be printed** by routines UTMESS,… presented in it document. **Only the supervisor** can use UTMESS <I>.

The orders wanting to transmit messages of information will do it with of WRITE and them utilities described in [D6.04.02] "utility of impression of information by IMPR of the orders".

Note:

We chose to print these messages with of "WRITE" because this method is more flexible that UTMESS: it is easier to manage the writings in columns. Moreover, each message is not systematically any more preceded by a white line and name of order (and routine) émétrice.

2

Operation general

Two methods are proposed in order to transmit a message:

·

if the message comprises only one simple text, one uses UTMESS [§3.5],
·
if the message comprises at the same time texts and values, the emission comprises three stages:
-
the initialization of message UTDEBM [§3.1],
-
the body of the message (répétable) [§3.3],
-
end of the message [§3.2].
The print routines described hereafter must incoming call the call to under
program initialization UTDEBM and a subroutine of termination UTFINM.
UTDEBM, by which one specifies the type of the message, prints a heading standardized indicating this type
and identifying the transmitter of the message.
Being given the variety of the situations this transmitter is not yet defined in a standardized way.
programmer will seek to define the transmitter so that one can find easily where the message has
summer emitted and by expecting that classification by name of transmitter can be used to establish one
documentation on the error messages. One proposes to name this transmitter by the chain of
characters formed by the name of the subroutine calling and a sequence number.
Example of call:
CALL UTDEBM ("F", "NMELNL_02", "VALUE OF NAKED NOT TROUVEE")
This call initiates the emission of a message of the type F, the transmitter is identified as being the routine
NMELNL, it is the second message appearing in the body of the routine.
The impressions are carried out via a plug of accumulation. Size of it
plug represents ten print lines. It is printed each time it is
filled.
The subroutine of termination UTFINM indicates to the software that the message is finished. In it
case, the current contents of the plug is emptied.
The plug has tabulations making it possible to align the impressions, these tabulations as well as
length in a number of characters per line are defined in the supervisor.
Four types of subroutines are available:
initializing routines
UTDEBM
and of end of use
UTFINM
print routines of values:
·
of integer type
UTIMPI
·
of real type
UTIMPR

.
of type
UTIMPC
complex
UTIMPK
.
of character type
UTIMPB
.
of logical type
subroutines of positioning
UTPOSI
message
UTSAUT
particular subroutine
equivalent with the call of UTDEBM and UTFINM
UTMESS
Note:
All the routines except UTMESS must be called between a UTDEBM and a UTFINM.

3
Subroutines with the use of the developers
3.1
Initializing routine
SUBROUTINE UTDEBM (ch1, the GPS, text)
in
ch1
K1

type of message (F, E, S, A or I)
in
the GPS
K*
name of the transmitter
in
text
K*
text to be printed

3.2

Subroutine of end of impression

SUBROUTINE UTFINM ()

This routine without argument makes it possible to declare the end of the impression and to empty the current contents
plug.

Note:

The call to this routine is obligatory to empty the plug of writing.

3.2

Print routines of values

SUBROUTINE UTIMPx (will cara, kvar, nbval, valley)
in
will cara
K1
editing character, takes the following values:
·
S the text and the values are put after in
plug,
·
L a return to the line is carried out before placing it
text and values in the plug.
in
kvar
K*
text
in
nbval
I
a number of values to be printed
in
valley
I
vector of values length nbval to be printed of type
R

correspondent with X:
C
·
INTEGER for UTIMPI
K*
·
REAL for UTIMPR
L
·
COMPLEX for UTIMPC
·
CHARACTER for UTIMPK
·
logical for UTIMPB

3.3
Subroutines of positioning in the plug

SUBROUTINE UTPOSI (ipos)
in
ipos
I
position in the line
This routine makes it possible to position a text or a value with a given column.
SUBROUTINE UTSAUT ()
This routine without argument makes it possible to jump a line.

3.4
Subroutine of simple messages

SUBROUTINE UTMESS (ch1, the GPS, text)

in
ch1
K1
type of message
in
the GPS
K*
name of the transmitter
in
text
K*
text to be printed

This routine makes it possible to print the heading and a text. The heading is the same one as that written by UTDEBM.

This subroutine empties the plug, a call to UTFINM is thus useless.

4

Examples of uses

First example

The following call:

CALL UTMESS (“F”, “NMDOME_05”, “IT THERE SEVERAL LOADS”
&
// ' THERMICS ')

Print the following message:

<F> <NMDOME_02> “IT THERE SEVERAL THERMAL LOADS”

Second example

Following calls:

CALL UTDEBM (“F”, “FOLOCA_02”, “ONE OVERFLOW ON THE LEFT”)
CALL UTIMPR (“It, '
VALUE A INTERPOLEE: ', 1, X)
CALL UTIMPR (“It, '
LOWER LIMIT: ', 1, VALLEY (1))
CALL UTFINM ()

Print the following messages:

<F> <FOLOCA_02> ONE OVERFLOWS ON THE LEFT
VALUE A INTERPOLEE: 1.2E-3
LOWER LIMIT: 1.5E-3

Third example

Following calls:

CALL UTDEBM (“, “OP0070”, “STOP FOR LACK OF TIME CPU”)
CALL UTIMPI (“, “WITH the NUMBER D " ORDER: ”, 1, NUMORD)
CALL UTIMPR (“It, “AVERAGE TIME BY INCREMENT OF LOAD: ”, 1,
&
TPS1 (4))

CALL UTIMPR (“It, “TIME CPU REMAINING: ”, 1, TPS1 (1))
CALL UTFINM ()
Print the following messages:
<S> <ASTER 2. 3.31 10/03/93 >
<S> <OP0070> STOP FOR LACK OF TIME CPU TO the SEQUENCE NUMBER:
2
AVERAGE TIME BY INCREMENT OF LOAD:
3.1856E+03
TIME CPU REMAINING:
1.5562E+03
Data-processing handbook of Description
D6.04 booklet:
Index A

---

5 Description of the internal initializing routine

This subroutine is only called by the Supervisor, it allows the initialization of the tables interns of correspondence classifies message/logical number of unit and parameters of management plug.

SUBROUTINE UTINIT (nbfica, mcol, itb)
in
nbfica
I
a number of files of alarm:
-
nbfica = 1 MESSAGE
-
nbfica = 2 MESSAGE and RESULT
in
mcol
I

a number of columns used for the impression (<133)
in
itb
I
not tabulations (to take 1 per defect)
Note:
·
the supervisor publishes the messages on 80 columns,
·
this routine calls upon function IUNIFI.

Data-processing handbook of Description
D6.04 booklet:
D6.04.02 document
Impressions directed by the key word INFORMATION
orders (package INFXXX)
Summary:
This document indicates how the developers must carry out the impressions directed by the word
key INFORMATION of the orders.
EDF
Direction of the Studies and Research
Electricity of France
Project Codes of Mechanics

---

1 Introduction

The information directed by the **key word INFORMATION** of the orders is message of information (INFORMATION)
concerning the course of calculation. They **contractual** (for level 1) and are documented in documentation of use.

Notice historical:

Until 1995, many orders comprised a key word "IMPR" of which the argument
could be worth: 0, 1, 2, 3,…. The effect of this key word was varied: writing of information, results, data… on the file "MESSAGE" or "RESULT". To clarify the situation, we have
decided to reabsorb gradually these key word IMPR by the key word INFORMATION (for INFORMATION) and
possible key word of impressions for the impression of the results.

The more detailed description of than is (or what is not) INFORMATION is made in [D2.07.01 §6], in short:

- INFORMATION is neither a result, nor an alarm…,
- the key word INFORMATION can take 2 values: 1 or 2; the defect is 1,
- the impressions are made by "WRITE" and not UTMESS <I> [D6.04.01 §1.5],
- INFORMATION are written on the file "MESSAGE"

The impressions being made by of WRITE formatted, the only problems to be regulated are:

- on which logical unit?
- how to respect the choice of the user: INFORMATION = 1 or 2?

This been the subject of paragraph 2.

Another problem occurs when an order **does not want a** message printed systematically by a utility routine. This been the subject of paragraph 3.

2

How to transmit a message of information?

·

Routine INFMAJ is used to inform "package" INFXXX of the level as impressions required by the user,

·

routine INFNIV is used to recover the level of impression required by the user like the logical unit of the file "MESSAGE".

·

the programmer of an order will thus make:

has

to put the key word INFORMATION in the **catalogue** of the order,

(2 possible values 1 and 2, default value: 1),

B

in routine OP000I

CALL GETVIS (", "INFORMATION", 0, 1, 1, NIV, IBID)

CALL INFMAJ (NIV)

C

in a routine wanting to print INFORMATION:

-

CALL INFNIV (IFM, NIV),

-

for INFORMATION of level 1 (contractual):

IF (NIV.GE.1) WRITE (IFM,…)

-

for INFORMATION of level 2:

IF (NIV.EQ.2) WRITE (IFM,…).

Note:

One could think that the INFORMATION of level 1 can be written without protecting itself by (IF (NIV.GE.1)…) because these impressions start by defect. One will see with [§3] why it is necessary the progéger.

To protect itself from the orders which do not have the key word INFORMATION in their catalogue, it supervisor updates the COMMON **before** each order:

Data-processing handbook of Description

D6.04 booklet:

Index A

CALL INFMAJ (1).

3
To protect itself from an excess of messages

The person in charge for an order is responsible for the impressions of level 1 of this order. He must to document and arrange themselves so that the file message remains readable.

A problem arises when an order called upon a utility printing of the INFORMATION of level 1 that the person in charge for the order judges without interest (or too many… for example in a loop). It is necessary to give to the order the means "of making conceal" the utility. For that, one has the 2 routines:

INFMUE () to make the code "DUMB" until new order,
INFBAV () to give the code in "TALKATIVE" mode.

These 2 routines are effective only if the level of impression required is 1: one cannot prevent it code "speech" if INFORMATION = 2.

Example:

Order OPOOOI does not want to only hear known as the routine B

SUBROUTINE
OPOOOI
CALL
INFMAJ
...
CALL
With (...)
...
CALL
INFMUE ()
CALL
B (...)
CALL
INFBAV ()
...
END

Note:

·

This system is not perfect because it only makes it possible to stop without understanding it flood of the impressions. It does not allow for example, to prohibit B from speaking all in leaving C (called by B) speech!

·

so that INFMUE/INFBAV is effective, it is necessary that the routines that one wants to make conceal their impressions of level 1 programmed in the following way: IF (NIV.GE.1)
WRITE….

---

4 Operation of utilities INFXXX

The purpose of this paragraph is only to help to include/understand (if necessary) the use of these routines:

·

COMMON/INF001/NIVUTI, NIVPGM, UNIT
INTEGER
NIVUTI:
level requested by the user: 1 or 2
INTEGER
NIVPGM:
level accessible to the programmer: 0, 1, or 2
INTEGER
UNIT:
logical unit of the file “MESSAGE”

·

COMMON INF001 **is used only by** routines INFXXX

·

SUBROUTINE INFMAJ ()
COMMON/INF001/…
GETVIS (nomCMD, “I”, “INFORMATION”,…, NIV)
NIVUTI = NIV
NIVPGM = NIV
UNIT = IUNIFI (“MESSAGE”)
END

·

SUBROUTINE INFNIV (IFM, NIV)
COMMON/INF001/…
INF = UNIT
NIV = NIVPGM
END

·

SUBROUTINE INFMUE ()
COMMON/INF001/…
IF (NIVUTI.EQ.1) NIVPGM = 0
END

·

SUBROUTINE INFBAV ()
COMMON/INF001/…
NIVPGM = NIVUTI
END

Data-processing handbook of Description
D6.04 booklet:
Index A

Code_Aster ®
Version
8.1

Titrate:
Use of the SD_RESULTAT

Date:
01/12/05
Author (S):
***J. PELLET**, L. VIVAN Key*
:
D6.05.01-B Page
: 1/10

Organization (S): EDF-R & D /AMA, CS

Data-processing handbook of Description
D6.05 booklet: -
D6.05.01 document

Use of the SD_RESULTAT

Summary:

The SD_RESULTAT [D4.06.08] is accessible only through the routines described in this document.

Routine RSCRSD must be supplemented to create new types of SD_RESULTAT.

Count

matters

Titrate:
Use of the SD_RESULTAT

Date:
01/12/05
Author (S):
***J. PELLET**, L. VIVAN Key*
:
D6.05.01-B Page
: 3/10

1
Routines of creation, destruction, enlarging

Note:

What a structure of data SD_RESULTAT, what a sequence number, one parameter, a variable of access…? the answers to these questions are in [D3.04.01] and [D4.06.08].

CALL RSCRSD
(nomsd, typesd, nbordr)

(in) nomsd

K8
name of the structure of data RESULT
(in) typesd
k16 standard of the structure of donnéss RESULT
(in) nbordr
I
sequence number numbers.

To create a structure SD_RESULTAT (if this structure already exists, one destroys it).
The modification of this routine makes it possible to create new types of SD_RESULTAT or to enrich those existing.

CALL RSDLSD
(nomsd)

(in) nomsd

K8 name of the structure of data RESULT

To destroy a structure SD_RESULTAT. This routine does not destroy the fields indexed by SD_RESULTAT.

CALL RSRUSD
(nomsd, iordr)

(in) nomsd
K8 name of the structure of data RESULT
(in) iordr I
sequence number.

To destroy in structure SD_RESULTAT the fields starting from a given sequence number.

CALL RSAGSD
(nomsd, nbordr)

(in) nomsd
K8 name of the structure of data RESULT
(in) nbordr I
new size of the RESULT (the size is doubled if NBORDR = 0)

To increase a structure SD_RESULTAT.

: 4/10

2
Routines of access starting from the key words of the order Aster

The access via the process control language to the data indexed in the SD_RESULTAT is possible by the same key words whatever the order (IMPR_RESU, CALC_G_THETA, etc…).

The description of these key words (TOUT_ORDRE, NUME_ORDRE, etc…) is for example in [U4.71.01].

For certain orders (IMPR_RESU, for example) these key words are under a key word factor (RESU) for others not (CALC_G_THETA). In this last case the argument key word specified in routines below must be ''.

CALL RSUTNU
(nomsd, motcle, iocc, knum, nbordr, jordr, prec,
crit, all, ilinst, ilfreq, iret)

(in) nomsd K8 name of the structure of data RESULT
(in) motcle K key word control ratio*
(in) iocc
I
number of occurrence of the key word factor
(in) knum
K19 name of vector JEVEUX to write the list of the numbers
(out) nbordr I
a number of found sequence numbers
(out) jordr
I
address JEVEUX of vector ZI
(in) prec
R8 precision requested
(in) crit
K8 criterion requested
(in) itout I
access of the order by the key word "ALL"
(in) ilinst I
access of the order by key word "LIST_INST"
(in) ilfreq I

access of the order by key word “LIST_FREQ”
(out) iret
I
code return, = 0: ok

Recovery starting from the key words of the order, the sequence numbers in a structure SD_RESULTAT.

Note:

Routine RSUTNU above recovers the list of the possible sequence numbers of SD_RESULTAT, possibly filtered by the key words of the process control language (TOUT=' OUI', LIST_INST=…). It is not sure for as much, that for all these numbers of order, all fields (NOM_CHAM) or all the parameters really exist. When one wants to recover the list of the sequence numbers for which a field (or a parameter) exists really, routines RSUTN1 or RSUTN2 should be used.

CALL RSTRAN
(interp, nomsd, motcle, iocc, kinst, krang, nbinst,
jinst, jrang, iret)

(in) interp standard K4 of interpolation wished
(in) nomsd K8 name of the structure tran_gene
(in) motcle K16 key word control ratio
(in) iocc
I
number of occurrence of the key word factor
(in) kinst K19 name of vector JEVEUX to write the list of the moments
(in) krang K19 name of vector JEVEUX to write the list of the numbers
(out) nbinst I
a number of moments read
(out) jinst
I
address JEVEUX of the vector ZR of the moments
(out) jrang
I
address JEVEUX of vector ZI of the numbers
(out) iret
I
code return, = 0: ok

Recovery from the key words of the order for a structure tran_gene Result, following *the type of interpolation:*

.

moments (INTERP = “FLAX”, “LOG”), user datum,

.

moments and their associated sequence numbers (INTERP = “NOT”), moments of calculation of tran_gene.

3
Routines of access to structures SD_RESULTAT

CALL RSORAC
(nomsd, access, ival, rval, kval, cval, prec, crit,
nutrou, ndim, nbtrou)

(in) nomsd K8 name of the structure of data RESULT
(in) access K16 name of the variable of access.
(in) ival
I
value of the so whole variable of access

(in) rval
R8 value of the so real variable of access
(in) kval
K. value of the variable of access if character
(in) cval
C16 value of the so complex variable of access
(in) prec
R8 precision requested
(in) crit
K8 criterion requested
(in) ndim
I
dimension of the list nutrou.
(out) nutrou I
list found sequence numbers.
(out) nbtrou I
a number of found sequence numbers.
if the number found nbtrou is higher than ndim,
then one returns nbtrou = - nbtrou

Recovery starting from a variable of access, sequence numbers of a structure SD_RESULTAT.

Convention:

if access =
“LONUTI”
: recovery working length of the .ORDR
if access =
“LONMAX”
: recovery maximum length of the .ORDR
if access =
“LAST”
: recovery of the last sequence number
if access =
“FIRST”
: recovery of the first sequence number
if access =
“TOUT_ORDRE”: recovery of all the sequence numbers.

CALL RSEXCH
(nomsd, nomsy, iordr, chextr, iret)

(in) nomsd K8 name of the structure of give RESULT
(in) nomsy K16 reference symbol
(in) iordr I
sequence number
(out) chextr K19 name of the extracted field
(out) iret
I
code return
0: the field exists.
>0: the field does not exist.
1) IORDR lower than the max is authorized:
100: the reference symbol is licit.
101: the reference symbol is prohibited.
1) IORDR higher than the max is authorized:
110: the reference symbol is licit.
111: the reference symbol is prohibited.

Recovery starting from a reference symbol and of a sequence number, name of a field in one structure SD_RESULTAT.
Data-processing handbook of Description
D6.05 booklet: -
HT-66/05/003/A

This routine manages the fields in structure SD_RESULTAT and called upon routine RSUTCH.

CALL RSNOCH
(nomsd, nomsy, iordr, chextr)

(in) nomsd K8 name of the structure of data RESULT
(in) nomsy K16 reference symbol
(in) iordr I
sequence number
(in) chextr K19 name of the field "extracts" on which one wants to point.

To note starting from a reference symbol and of a sequence number, the name of a field in a structure SD_RESULTAT.

Convention:

chextr
= '': the name of the field is that built by the routine,
chextr
'': the name of the field will be that provides by the developer.

Note:

Routine RSNOCH manages the bonds (via routine SDLIEN) between the various structures SD_RESULTAT.

CALL
RSADPA
(nomsd, concealment, will npara, lpara, iordr, itype, ljeveu,
ctype)

(in) nomsd K8 name of the structure of data RESULT
(in) concealment
K1 condition of access to the parameters:
"It: reading, "E": writing.
(in) I will npara

a number of sought parameters
(in) K16 will lpara lists names of the parameters
(in) iordr I
sequence number
(in) itype I
code indicating that one wishes the type
= 0 step of the type
0 one provide the type
(out) ljeveu I
list addresses JEVEUX in ZI, ZR,…

(out) ctype
K4
list types of addresses JEVEUX
I whole, R real, C complex,
K8 K16 K24 K32 K80 character.

Recovery starting from and a list sequence number of name (S) of parameter (S) or variable (S) access, addresses JEVEUX of a structure SD_RESULTAT.

Data-processing handbook of Description
D6.05 booklet: -
HT-66/05/003/A

CALL RSNOPA
(nomsd, icode, nomjv, nbacc, will nbpara, jpara)

(in) nomsd K8 name of the structure of data RESULT
(in) nomjv K19 name of vector JEVEUX to write the list of the names
(in) icode I
= 0, variables of access alone
= 1, parameters alone
= 2, variables of access and parameters
(out) nbacc
I
a number of variables of access
(out) I will nbpara
a number of parameters
(out) will jpara
I
address JEVEUX of nomjv

Recovery of the names and the number of parameters in a structure SD_RESULTAT as well as names and the number of variables of access. These names are stored in a vector JEVEUX of name nomjv.

CALL RSUTNC
(nomsd, nomsy, ndim, nomch, nuordr, nbtrou)

(in) nomsd K8 name of the structure of data RESULT
(in) nomsy K16 reference symbol
(in) ndim
I
dimension of the tables
(out) nomch
K16 table of the field names
(out) nuordr I
table of the sequence numbers of the found fields
(out) nbtrou I
a number of found fields
if the number found nbtrou is higher than ndim,
then one returns nbtrou = - nbtrou

Recovery starting from a reference symbol, fields noted and their associated sequence numbers in a structure SD_RESULTAT.

CALL RSUTOR
(nomsd, nomsy, chextr, ndim, tnomsy, nuordr, nbtrou)

(in) nomsd K8 name of the structure of data RESULT
(in) nomsy K16 reference symbol
(in) chextr K19 name of the field to be sought
(in) ndim
I
dimension of the tables
(out) tnomsy K16 table of the reference symbols
(out) nuordr I
table of the sequence numbers
(out) nbtrou I
a number of found fields
if the number found nbtrou is higher than ndim,
then one returns nbtrou = - nbtrou

Recovery starting from a field, sequence number and possibly the reference symbol in one structure SD_RESULTAT.

Convention:

if nomsy = '': recovery of the reference symbols and the sequence numbers.

Data-processing handbook of Description
D6.05 booklet: -
HT-66/05/003/A

---

4 Routine interpolation

CALL
RSINCH
(nomsd, nomsy, access, rval, chextr, proldr,
prolga, istop, iret)

(in) nomsd K8 name of the structure of data RESULT
(in) nomsy K16 reference symbol
(in) access K16 name of the variable of access
(in) rval
R8 actual value of the variable of access
(in) chextr K19 name of the field has to create. (if there exists, it is destroyed)
(in) proldr standard K8 of prolongation wanted on the right
(in) prolga standard K8 of prolongation wanted on the left
(in) istop I
in the event of interpolation error:
= 0, do not write a message, do not make stop.
= 1, written message, does not make stop.
= 2, written message, made stop.
(out) iret
I

code return:
1) the field is calculated:
= 0, the field are interpolated between 2 values.
= 1, the field is prolonged on the left.
= 2, the field is prolonged on the right.
1) the field is not calculated:
= 10, there is not any field for the interpolation.
= 11, prolongation on the left prohibited.
= 12, prolongation on the right prohibited.
= 20, the variable of access is illicit.

Interpolation for a reference symbol and a variable of access, a field in a structure SD_RESULTAT.

5 Routines
of existence

CALL RSEXIS
(nomsd, iret)

(in) nomsd K8 name of the structure of data RESULT

(out) iret
I
code return
= 0, the structure exist; 0, the structure do not exist.

Checking of the existence of a structure SD_RESULTAT.

CALL RSEXPA
(nomsd, icode, nompar, iret)

(in) nomsd K8 name of the structure of data RESULT
(in) icode I
= 0, variable of access
= 1, parameter
= 2, variable of access or parameter
(in) nompar K16 name of the parameter or the variable of access
(out) iret
I
= 0, the name do not exist.
= 100, the name exists

Checking of the existence of a name of parameter or a name of variable of access in a structure SD_RESULTAT.

CALL RSVPAR
(nomsd, iordr, nompar, ipar, rpar, kpar, iret)

(in) nomsd K8 name of the structure of data RESULT
(in) iordr I
sequence number
(in) nompar K16 name of the parameter to be checked
(in) ipar

I
so whole value of the parameter
(in) rpar
R8 so real value of the parameter
(in) kpar
K. value of the parameter if character
(out) iret
I
= 0, it are not a parameter
= 100, the value of the parameter is correct
= 110, the value of the parameter is not correct

Checking of the existence of a name of parameter and its value for a sequence number gives in a structure SD_RESULTAT.

---

6 Example

Let us take the example of the storage of the modes of mechanical vibration (mode_meca):

one calculated 5 modes and one stores them in a structure SD_RESULTAT.

Stage 1: creation of structure SD_RESULTAT

NBMODE

=
5

CALL RSCRSD (MODES, “MODE_MECA”, NBMODE)

Stage 2: storage of the clean vectors

C 10 LM = 1, NBMODE

...

CALL RSEXCH (MODES, “DEPL”, IM, CHAMNO, IER)

...
CALL
JEVEUO (CHAMNO//“.VALE”, “E”, LVALE)

C 12 IEQ = 1, NEQ
ZR (LVALE+IEQ-1)
=
...
12
CONTINUOUS

CALL RSNOCH (MODES, “DEPL”, IM, '')

...

Stage 3: storage of the variables of access and parameters modal

CALL RSADPA (MODES, "E", 1, "FREQ", IM, 0, LFREQ, K8B)
ZR (LFREQ)
=
...

CALL RSADPA (MODES, "E", 1, "NUME_MODE", IM, 0, LNUME, K8B)
ZI (LNUME)
=
...

CALL RSADPA (MODES, "E", 1, "STANDARD", IM, 0, LNORM, K8B)
ZK24 (LNORM)
=
...

...

Author (S):
***E. SCREWS**, J. PELLET*

Organization (S): EDF/IMA/MMN

Use of the structures of data counts

Summary:
This document describes the routines making it possible to use the structures of data counts described in [D4.02.05].

Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

Contents

Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

***Code_Aster* ®**
Version
4.0
Titrate:
Use of the structures of data counts
Date:
19/03/98
Author (S):
***E. SCREWS**, J. PELLET*
Key:
D6.06.01-B
Page:
3/18

1 General information

1.1 What is this that one counts? :

A table is a structure of data of data-processing nature making it possible to store one together of whole values, real, complex or character strings.

*A table is comparable with a data base EXCEL (version 5), i.e. one can see it as a list of columns (the term of "column" is here to make the bringing together with a list EXCEL; that does not want to say that the impression of a table is always done in column (see [§7])) in opposite. Each column has a field name, that we call **parameter**, and contains of similar data of type: I, R, C, K8, K16, K24 or K32.*

Example 1: T1

NUME_ORDRE	***INST***	***NODE***	***G_LOCAL***
1	*10.*	*N1*	*5.*
1	*10.*	*N2*	*6.*
1	*10.*	*N3*	*7.*
1	*10.*	*N4*	*8.*
2	*20.*	*N1*	*9.*
2	*20.*	*N2*	*9.*
2	*20.*	*N3*	*8.*
2	*20.*	*N4*	*8.*
3			

30.
N1
7.
3
30.
N2
6.
3
30.
N3
5.
3
30.
N4
4.
Example 2: T2
ACTION
NUME_ORDRE
INST
NODE
DX
DY
NET
SIXX
ENTITLE 1
1
10.
N1
3.
5.
ENTITLE 1
1
10.
N2
6.
7.
ENTITLE 1
1
10.
N3
8.
9.
ENTITLE 1

2
20.
N1
11.
12.
ENTITLE 1
2
20.
N2
15.
13.
ENTITLE 1
2
20.
N3
19.
18.
ENTITLE 2
2
20.
MA1
-12.
ENTITLE 2
2
20.
MA2
-14.

1.2
Some properties of the tables

· A table has a limited number of columns (or parameters). These parameters are chosen by developers of the orders creating of the tables. The name of a parameter is a chain of with more 16 characters.

· On the other hand, the number of lines of a table is often "dynamic": it depends in general choices of the user: nodes of examination, moments of calculations,…

· The values contained in a column of a table are very in the same way standard FORTRAN: realities, complexes, entireties or texts. One can store in form "text" in a table of the names of SD Aster or of the names of objects JEVEUX; for example of the names of functions.

· A table is known as "full" when all its lines contain values for all them parameters of the table. Table T1 above is full. A table which is not full is known as "digs" (table t2 above).

· The lines of a table are naturally ordered by their order of insertion in the table (routine TBAJLI).

· The columns of a table are naturally ordered by the order of declaration of theirs parameters (routine TBAJPA).

· A table has at least a line. On a line, there can be blank cells: the parameter associated is not affected (table t2 above).

2 ***List utilities for the tables***

Function	***name***	***paragraph***
To create a new table	*TBCRSD*	*[§3.1]*
To declare the parameters	*TBAJPA*	*[§3.2]*
To recover the existence and the type of a parameter	*TBEXIP*	*[§3.3]*
To amalgamate several tables in only one	*TBFUTB*	*[§3.4]*
To destroy a table	*DETRSD*	*[§3.5]*
To duplicate a table	*COPISD*	*[§3.6]*
To build a table while adding to it of the lines to one		

TBAJLI
[§4.1]
To recover the number of a line in a table
TBNULI
[§4.2]
To print a table on listing
TBIMPR
[§7]
To filter the lines of a table by imposing criteria on one or more TBEXTB
[§6.2]
parameters to create a new table of smaller size
To sort the lines of a table according to certain parameters TBTRTB
[§6.3]
selected. The result of the sorting is a table whose lines were
reordered.
To collect in a vector the values corresponding to a parameter TBEXVE
[§5.1]
given
To create a function starting from 2 columns of a table
TBEXFO
[§5.2]
To see the value associated with a parameter given for a line with a table
TBLIVA
[§5.3]
To recover all the numerical values of a table in a SD TBEXLR
[§8]
LISTR8

3

Routines of management of a table

3.1

*Routine TBCRSD**: To create a new table***

TBCRSD
(nomtab, bases)
nomtab
in
K19
Name of the new table to be created
If it already exists, it is destroyed + emission <A>
base
in
K1
Base creation of the table counts (“G”, “V”,…)

3.2

*Routine TBAJPA**: To add parameters to the table***
TBAJPA
(nomtab, nbpar, nompar, typpar)
nomtab
in
K19
Name of the table where one wants to add parameters
nbpar
in
I
A number of parameters to be added
nompar
in
V (K16)
List names of the parameters to be added
typpar
in
V (K8)
List types of the parameters:
"R", "I", "It, "K8", "K16", "K24", "K32"

3.3
*Routine TBEXIP**: To test the existence and the type of a parameter***
TBEXIP
(nomtab, para, exist, typpar)
nomtab
in

K19
Name of the table to be examined
para
in
K16
parameter to be tested
exist
out
L
.TRUE. : the parameter already exists in the table nomtab
typpar
out
K8
type of the parameter if there exists déjàdans the table:
“R”, “I”, “It, “K8”, “K16”, “K24”, “K32”
3.4
*Routine TBFUTB**: To amalgamate several tables in only one table***
TBFUTB
(tabout, basout, ntab, ltabin, para, typpar, VI, vr, vc, vk)
tabout
in
K19
Name of the table which one wants to create
basout
in
K1
Base creation of the table tabout (“G”, “V”,…)
ntab
in
I
A number of tables which one wants to amalgamate
ltabin
in
K19
Names of the tables which one wants to amalgamate
para
in
K16
New parameter (optional) which will make it possible to distinguish the origin of each line of the new table
if para=' 'the following arguments are not used.
typpar
in

K8
Type of the new parameter (optional)
VI
in
V (I)
List values for the new parameter “I” (optional)
vr
in
V (R)
List values for the new parameter “R” (optional)
vc
in
V (C)
List values for the new parameter “It (optional)
vk
in
V (K)*
List values for the new parameter “K” (optional)
This routine can be practical to create a hollow table, the developer can realize each under-table separately and to use routine TBFUTB to amalgamate them in only one table.
Example:
One wants to amalgamate the 2 tables: T1 and T2
With
B
C
D
x1
x2
x3
x4
x5
x6
x7
x8
With
B
E
y1
y2
y3
y4
y5
If one writes:

ltabin (1) =T1
ltabin (2) =T2
CALL TBFUTB (T3, “V”, 2, ltabin, '', kbid, ibid, rbid, cbid, kbid)
The table is obtained: T3
With
B
C
D
E
x1
x2
x3
x4
x5
x6
x7
x8
y1
y2
y3
y4
y5

If one writes:
ltabin (1) =T1
ltabin (2) =T2
VK (1) = ' ACTION1'

VK (2) = ' ACTION2'
CALL TBFUTB (T3, “V”, 2, ltabin, “, “K8”, ibid, rbid, cbid, VK)
The table is obtained: T3
NR
With
B
C
D
E
ACTION1
x1
x2
x3
ACTION1
x4
x5
ACTION1
x6
x7
x8
ACTION2
y1
y2
y3
ACTION2
y4
y5

Remarks on the order of the lines and the columns of the new table:

The lines of the new table are ordered by putting the lines of the tables end to end that one amalgamates.

To order the parameters the following rules are adopted:

· the new parameter (optional) is numbered in first,

· the parameters of the 1st table of ltabin are then added in the order that they have in ltabin (1)

· the parameters of the 2nd table of ltabin are then added (except those already present in ltabin (1))in the order which they have in ltabin (2)

· ...

3.5

To destroy a table

CALL DETRSD (“TABLE”, nomtab)

3.6

To duplicate a table

CALL COPISD (“TABLE”, “V”, tabin, tabout)

4

Routines of writing of values in a table

4.1

*Routine TBAJLI**: To add a line to the table***

TBAJLI
(nomtab, nbpar, nompar, VI, vr, vc, vk, nume)
nomtab
in
K19
Name of the table where one wants to add a line
jxvar
nbpar
in
I
A number of parameters for the line
nompar
in
V (K16)
List names of the parameters of the line
VI
in
V (I)
List values for the parameters "I"
vr
in
V (R)
List values for the parameters "R"
vc
in
V (C)
List values for the parameters "It
vk
in
V (K)*
List values for the parameters "K"
nume
in
I
/0: one adds the line at the end of the table
/I: the iième line of the table is replaced

Example:
That is to say a table containing the "whole" parameters of the type "I1", "I2", "I3", parameters "real" "R1" and "R2" and parameters "character strings" "K1" and "K2"
Let us suppose that one wants to add a line to it containing: I2=i2, R1=r1, K2=k2, K1=k1.
One will be able to call routine TBAJLI with the arguments:
nbpar = 4
nompar = ("I2", "R1", "K2", "K1")
VI = (i2)
vr = (r1)
vk = (k2, k1)
nume = 0

4.2
*Routine TBNULI***: Allows to recover a number of line in a table**

TBNULI
(tabin, npacri, lipacr, VI, vr, vc, vk, lprec, lcrit, nume)
tabin
in
K19
Name of the table which one wants to recover a line
npacri
in
I
A number of parameters implied in the criteria of selection line (dimension of lipacr)
lipacr
in
V (K16)
List names of the parameters criteria
VI

in
V (I)
Values of the criteria for the parameters "I"
vr
in
V (R)
Values of the criteria for the parameters "R"
vc
in
V (C)
Values of the criteria for the parameters "It
vk
in
V (K)
Values of the criteria for the parameters "K"
lcrit
in
V (K8)
List secondary criteria of equality for the parameters
floating:
"EQUAL", "RELA", "ABSO" [§6.1.1].
lprec
in
V (R)
List precise details (for the criteria of equality of the parameters
floating [§6.1.1].)
nume
out
I
= 0: There is no line corresponding to the criteria.
= I: The line I is the only one which corresponds to the criteria of
selection
< 0: There are several lines corresponding to the criteria
One seeks a line in the table tabin by imposing conditions on his parameters.
The mechanism of selection of a line in a table (arguments lipacr, lcrit, lprec, VI,
vr, vc, vk) are explained to [§6.1].
When this line is found (and single), one returns his number what makes it possible to modify this line using routine TBAJLI.
4.3 Examples
That is to say the table: T3
With
B
C

x1
x2
x6
x7
x8
One adds a line to T3:
CALL TBAJLI (T3,1, “B”, ibid, z1, cbid, kbid, 0)
The table then is obtained:
With
B
C
x1
x2
x6
x7
x8
Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

---

z1
One recovers the number of the line such as A=x1:
CALL TBNULI (T3,1, "A", ibid, x1, cbid, kbid, 1.d-6, "RELA", ilig)
The line ilig is modified:
CALL TBAJLI (T3,1, "It, ibid, z2, cbid, kbid, ilig)
The table then is obtained:
With
B
C
z2
x6
x7
x8
z1

5 Routines of reading of values in a table

5.1 Routine TBEXVE: Reading of the values of a column of a table

TBEXVE
(nomtab, para, nomobj, basobj, nbval, typval)
nomtab
in
K19
Name of the table in which one wants to extract one
jxin
column
para
in
K16
Parameter indicating the column to be extracted

nomobj
in
K24
Name of object JEVEUX containing the values read in
jxout
the table
basobj
in
K1
Base “G”, “V” on which one created the vector nomobj
nbval
out
I
Numbers of extracted values
typval
out
K4
Type JEVEUX of the extracted values:
I/R/C/K8, K16,…

5.2

Routine TBEXFO**: Create a function starting from a table**

TBEXFO
(nomtab, parax, paray, nomfo, basfon)
nomtab
in
K19
Name of the table in which one wants to extract one
jxin
function.
parax
in
K16
Parameter X-coordinate indicating the column to be extracted
paray
in
K16
Parameter ordered indicating the column to be extracted
nomfo
in
K24
Name of the function to be created
jxout
basfon

in
K1
Base “G”, “V” on which one creates the function
5.3
Routine TBLIVA**: Reading of the value of a cell**
TBLIVA
(nomtab, npacri, lipacr, VI, vr, vc, vk, lcrit, lprec, para, ctype,
vali, valr, valc, valk, ier)
nomtab
in
K19
Name of the table in which one wants to extract the value
jxin
of a cell
npacri
in
I
A number of parameters implied in the criteria of
choice of the line (dimension of lipacr)
lipacr
in
V (K16)
List parameters criteria
VI
in
V (I)
Values of the criteria for the parameters “I”
vr
in
V (R)
Values of the criteria for the parameters “R”
vc
in
V (C)
Values of the criteria for the parameters “It

---

vk
in
V (K*)
Values of the criteria for the parameters “K”
lcrit
in
V (K8)
List secondary criteria of equality for
floating parameters:
“EQUAL”, “RELA”, “ABSO” [§6.1.1]
lprec
in
V (R)
List precise details (for the criteria of equality of
floating parameters [§6.1.1])
para
in
K16
Parameter associated with the column with the sought value
ctype
out
K8
Type of the found value
vali
out
I
Found value if parameter “I”
valr
out
R
Found value if parameter “R”
valc
out
C

Found value if parameter “It
valk
out
K*
Found value if parameter “K”
ier
out
I
Code return:
0: OK
1: para does not exist in the table for the found line
2: no the line found corresponding to the criteria
3: several lines found corresponding to the criteria
This routine makes it possible to read the value associated with a parameter given for a selected line in a table. One selects the line by imposing values on certain parameters.
mechanism of selection of a line in a table (arguments lipacr, lcrit, lprec, VI, vr, vc, vk) are explained to [§6.1.1].
Example:
That is to say the table: T3
With
B
C
7.
4
“Z”
12.
0
“A1”
4
“A2”
lipacr (1) = ' A', vr (1) =6.999, lcrit (1) = ' RELA', lprec (1) =0.01
lipacr (2) = ' B', VI (1) = 4
CALL TBLIVA (T3,2, lipacr, VI, vr, vc, vk, lcrit, lprec, “It, ctype,
vali, valr, valc, valk, ier)
at exit one a:
valk=' Z'
ier=0
ctype=' K8'

6 Routines of filtering and sorting of a table

6.1 Mechanism of filtering of lines in a table

In routines TBLIVA, TBNULI and TBEXTB, it is necessary “to filter” an existing table for

to retain only one of them (or several) lines. It is this mechanism which we explain here.
To filter a table, the user forces criteria on certain parameters. He will say for example
to retain that lines of the table for which NUME_ORDRE=1.Il can use several criteria of
selection of the lines and the same parameter can appear several times in the list of the criteria.

The “types” of possible selection criterion are:
EQ
“equality” for the entireties, the texts, realities or the complexes.
For the floating numbers (real or complex), this criterion is supplemented by one
“secondary” criterion explained below (cf 6.1.1).

“not-equality” (cf EQ)
LT
“smaller than”
Relations of order:
- natural for the entireties and realities
- alphabetical for the texts
- **invalid** for the **complexes**
WP
“larger than” (cf LT)

“smaller or equal to” (cf LT)
GE
“larger or equal to” (cf LT)
VACUUM
blank cell
NON_VIDE
Nonempty cell

Note:
In routines TBLIVA and TBNULI, the type of criterion of selection is (for the moment) **always** equality ("EQ"). The list of the criteria given in these 2 routines (lcrit) is thus the list of "secondary" criteria (cf 6.1.1).
Arguments: VI, vr, vc, vk of these routines contain the values associated with these criteria (according to the type associated with each parameter to which the criteria relate).
We will explain the use of all these arguments on an example (without floating numbers):
That is to say a table containing of the I1 parameters, I2, I3 of the "whole" type and K1, K2 of the type "character".
One wants to extract from them the lines which satisfy the following criteria:
1) the value of K1 is different from "ACTION1"
2) the value of I2 lies between 12 and 21
3) the value of I3 is worth 999
4) the value of I1 is "nonempty"
5) the value of K2 is "higher" (within the meaning of the alphabetical order) than "III"
One will give like arguments of TBEXTB:
npacri=6 There are 6 selection criteria because the 2nd criterion is double: (x>12) and (x<21)
lipacr = ("K1", "I2", "I2", "I3", "I1", "K2")
lcrpa = ("", "WP", "LT", "EQ", "NON_VIDE", "WP")
vk (1) = ' ACTION1', VI (1) =12, VI (2) =21, VI (3) =999, vk (2) = ' III'
Note:
The types of criterion "EMPTIES" and "NON_VIDE" do not require arguments in the tables VI, vr,…

6.1.1 Particular case of the criteria of equality (or nonequality) for numbers "floating"

The equality of the floating numbers is a dangerous concept in data processing because it can depend on certain truncations: errors rounding for example. For these criteria of selection, one thus uses a list of secondary criteria which specify which equality is desired. There is three possible type for secondary criteria:
"EQUAL"
"exact" equality of the 2 floating numbers
"ABSO"
equality of the 2 floating numbers except for an epsilon (eps) in absolute comparison:
truth if $|x1-x2| < eps$
"RELA"
equality of the 2 floating numbers except for an epsilon (eps) in relative comparison:
truth if $|x1-x2| < eps * |x1|$
Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

Code_Aster ®
Version

4.0
Titrate:
Use of the structures of data counts
Date:
19/03/98
Author (S):
E. SCREWS, J. PELLET
Key:
D6.06.01-B
Page:
11/18
Example:
That is to say a table containing the whole parameters: I1 and I2, the real parameters R1 and R2 and complex parameters C1 and C2. One wants to extract the lines from them corresponding to the following criteria:
1) I1 > 12
2) C1 = c1 with 0.01 near in “absolute”
3) R1 = r1 with 0.1 near in “relative”
4) C2/= c2 with 0.02 near in “relative” (the numbers which **are not** in the disc of center
c2 and of ray 0.002*c2)
5) R2/= r2 to 0.2 near in “absolute” (the numbers which **are not** in the interval of center
r2 and of ray 0.2)
One will give like arguments of TBEXTB:
npacri=5
lipacr = (“I1”, “C1”, “R1”, “C2”, “R2”)
licrpa = (“WP”, “EQ”, “EQ”, “”, “”)
VI (1) = 12, vr (1) =r1, vr (2) =r2, vc (1) =c1, vc (2) =c2
lcrit = (“ABSO”, “RELA”, “RELA”, “ABSO”)
lprec = (0.01, 0.1, 0.02, 0.2)
6.2
Routine TBEXTB**: Filtering and extraction of a new table**
TBEXTB
(tabin, basout, tabout, npacri, lipacr, lcrpa, VI, vr, vc, vk,
lprec, lcrit)
tabin
in
K19
Name of the table which one wants to extract from the lines
jxin
basout
in
K1
“G”, “V”: base creation of tabout

tabout
in
K19
Name of the table which will contain the lines extracted from
jxout
tabin
npacri
in
I
A number of parameters implied in the criteria
of extraction (dimension of lipacr and lcrpa)
lipacr
in
V (K16)
List parameters criteria [§6.1]
lcrpa
in
V (K10)
List criteria of selection:
EQ, LT, WP, IT, GE, EMPTY, NON_VIDE
The significance of these criteria is given to [§6.1].
lcrit
in
V (K8)
List secondary criteria of equality for
floating parameters:
“EQUAL”, “RELA”, “ABSO” [§6.1.1].
lprec
in
V (R)
List precise details (for the criteria of equality of
floating parameters [§6.1.1].)
VI
in
V (I)
Values of the criteria for the parameters “I”
vr
in
V (R)
Values of the criteria for the parameters “R”
vc
in
V (C)

Values of the criteria for the parameters “It
vk
in
V (K)
Values of the criteria for the parameters “K”
Note:
When one recovered in tabin the lines satisfying all the criteria given, one
preserve in the table tabout that the columns for which there is at least a value.
The same parameter can appear several times in the list of the criteria (lcrpa).
Example:
That is to say the table: T3
With
B
C
4
“Z”
11.
0
“A1”
24.
9
“A2”
240.
9
“A2”

12.

12.
lipacr (1) = ' A'
lcrpa (1) = ' NON_VIDE'
lipacr (2) = ' B'
lcrpa (2) = ' GT'
VI (1) = 1
lipacr (3) = ' A'
lcrpa (3) = ' LE'
vr (1) = 1.d2
CALL TBEXTB (T3, “G”, T3B, 3, lipacr, lcrpa, VI, vr, vc, vk, lprec, lcrit)
at exit T3B contains:
With
B
C
24.
9
“A2”
12.
12.
6.3
Routine TBTRTB**: Sorting of the table**
TBTRTB
(tabin, basout, tabout, will npara, lipara, lcrit)
tabin
in
K19
Name of the table which one wants to classify the lines
jxin
basout
in
K1
Base creation of tabout: G/V/L
tabout
in
K19
Name of the table containing all the lines of tabin
jxout
classified according to following criteria's
will npara
in
I
A number of parameters implied in the sort criteria
(dimension of will lipara and lcrit)

will lipara
in
V (K16)
List parameters criteria
lcrit
in
V (K2)
List types of criteria: “CR” or “DR.”
“CR”: ascending order
“DR.”: descending order
This routine is used to create a new table (tabout) by permuting the order of the lines of a table existing (tabin) according to certain sort criteria.

Sorting

Values R, I: By ascending values or decreasing
Values K8, K16, K24, K32: Alphabetically growing or decreasing
Example:
will npara = 2
will lipara = (“NODE”, “INST”)
lcrit = (“CR”, “DR.”)
The new table (tabout) will be ordered:
· firstly alphabetically crescent of the names of nodes,
· secondly by order descending of the moments of calculation.

Note:

The 2nd criterion does not apply that if there is equality within the meaning of the first criterion.
The blank cells are classified in “head” (they are smallest for the relation of order).

7

Routine TBIMPR: **Impression of a table**
TBIMPR
(table, format, file, nparim, lipaim, nparpg, lipapg, formar,
formac)
count
in
K19
Name of the table which one wants to print
format
in
K8
Format of impression of the table:
“EXCEL”, “TABLE”, “MOT_CLE”
file
in
K*
Name of the file listing: “RESULT”,…
nparim
in
I
A number of parameters to be printed
if nparim=0, one prints ALL the parameters.
lipaim
in
V (K16)
List parameters to be printed.
The order of impression of the parameters is that of the list
lipaim
nparpg
in
I
A number of parameters of “pagination”
lipapg
in
V (K16)
List parameters of pagination
formar
in
K8
/Format of writing of the actual values
/"=> default value: 1PE12.5
formac
in

K2
Convention of writing of the complex numbers:
/"=> default value: “IH”
/“IH” (left real, imaginary part)
/“MP” (module, phase)
The user with the possibility of printing his results under the following formats:

7.1 FORMAT
:
“EXCEL”

NUME_ORDRE
INST
NODE
DX
DY
1
4.
N7
3.4
3.8
1
4.
N4
2.4
2.8
1
4.
N2
1.4
1.8
4
8.
N7
3.4
3.8
4
8.
N4
2.4
2.8
4
8.
N2
1.4

1.8
7
20.
N7
3.4
3.8
7
20.
N4
2.4
2.8
7
20.
N2
1.4
1.8

7.2 FORMAT
:
“TABLE”

DX
INST
4.
8.
20.
N7 NODE
3.4
3.4
3.4
N4
2.4
2.4
2.4
N2
1.4
1.4
1.4

7.3 FORMAT
:
“MOT_CLE”

NUME_ORDRE: 1 INST: 4.
NODE: N7 DX: 3.4
DY: 3.8
NUME_ORDRE: 1 INST: 4.

NODE: N4 DX: 2.4
DY: 2.8
NUME_ORDRE: 1 INST: 4.
NODE: N2 DX: 1.4
DY: 1.8
NUME_ORDRE: 4 INST: 8.
NODE: N7 DX: 3.4
DY: 3.8
...

7.4 FORMAT
"EXCEL" with pagination (here definite by the parameter
"NODE")

NODE: N7
NUME_ORDRE
INST
DX
DY
1
4.
3.4
3.8
4
8.
3.4
3.8
7

20.
3.4
3.8
NODE: N4
NUME_ORDRE
INST
DX
DY
1
4.
2.4
2.8
4
8.
2.4
2.8
7
20.
2.4
2.8
NODE: N2
NUME_ORDRE
INST
DX
DY
1
4.
1.4
1.8
4
8
1.4
1.8
7
20.
1.4
1.8
By defect the format of impression is the format “EXCEL”, i.e. presentation in columns of various selected parameters.

8

Routine TBEXLR**: Transformation of a table into a SD**

LISTR8

8.1

Routine drank

To reach quickly (on the level of the routines te00ij for example) information contained in tables, one can transform these tables into SD of the type LISTR8. It is the object of the routine TBEXLR.

Note:

The reading of this paragraph 8 can be jumped by all those which are not interested in the routine very particular that is TBEXLR.

8.2 Interface of TBEXLR

TBEXLR
(table, lr8, base)
count
in
K19
Name of the table which one wants to transform into SD LISTR8
lr8
in
K8
Name of SD LISTR8 result
jxout
base
in
K1
Base creation of lr8: “G”, “V”,…

8.3 Restrictions of use of TBEXLR

Routine TBEXLR can transform a table into LISTR8 only if this table is “diagonal by blocks”. If it is not the case, the routine will stop in fatal error.

D6.06.01-B
Page:
15/18

8.3.1 Examples of diagonal tables per blocks

With
B
C
D
E
F
1.
2.
2.1
3.
8.
9.
10.
11.
19.
With
B
C
D
E
1.
3.
10.
11.

8.3.2 Examples of nondiagonal tables per blocks

With
B
C
D
E
F
1.
2.
2.1
3.
5.
7.
8.
9.

10.
11.
19.
With
B
C
D
E
F
1.
2.
2.1
3.
8.
9.
10.
11.
4.
19.
8.4 How
one
is table transformed into LISTR8?
That is to say the table
parameter
B
C
D
E
F
G
H
I
K
type K1 values
R
I
K1
R
R
I
I
R
W

1.
2
S
3.
2.
2
S
3.
W
3.
2
S
3.
S
1
2
S
1
3
S
1
4
S
1
5
S
12.
13.
14.
S
15.
Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Use of the structures of data counts
Date:
19/03/98
Author (S):

E. SCREWS, J. PELLET
Key:
D6.06.01-B
Page:
16/18
S
16.
8.4.1 stage 1: one retains only the nonempty columns of type “I” or “R”
parameter
C
D
F
H
I
K
type values R
I
R
I
I
R
1.
2
3.
2.
2
3.
3.
2
3.
1
2
1
3
1
4
1
5
12.
13.
14.
15.
16.

8.4.2 stage 2: recognition of the "blocks" of the "diagonal" table
parameter
C
D
F
H
I
K
type values R
I
R
I
I
R
1.
2
3.
2.
2
3.
3.
2
3.
1
2
1
3
1
4
1
5
12.
13.
14.
15.
16.

8.4.3 stage 3: setting in vector of the values selected, conversion of the entireties into realities:

One puts end to end in the vector of realities the numerical values found in the blocks of count:

nb_blocs, nb_col (bloc1), nb_lig (bloc1), values (bloc1), nb_col (bloc2), nb_lig (bloc2), values (bloc2),…

The values of a block are written line by line. The table above becomes the vector of realities then

below.
3.
3.
3.
1.
2.
3.
2.
2.
3.
3.
2.
3.
2.
4.
1.
2.
1.
3.
1.
4.
1.
5.
1.
5.
12.
13.
14.
15.
16.

Note:

We presented the list of realities on 4 lines to facilitate the reading of the result of transformation, but the produced LISTR8 contains (3*3+2*4+1*5) + 3*2 + 1 realities simply put end to end.

Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Use of the structures of data counts

Date:
19/03/98
Author (S):
E. SCREWS, J. PELLET
Key:
D6.06.01-B
Page:
17/18

9
Simple example of creation and exploitation of a table

That is to say the example of the T2 table:

ACTION
NUME_ORDRE
INST
NODE
DX
DY
NET
SIXX
ENTITLE 1 1
10.
N1
3.
5.
ENTITLE 1 1
10.
N2
6.
7.
ENTITLE 1 1
10.
N3
8.
9.
ENTITLE 1 2
20.
N1
11.
12.
ENTITLE 1 2
20.
N2
15.

13.
ENTITLE 1 2
20.
N3
19.
18.
ENTITLE 2 5
20.
MA1
-12.
ENTITLE 2 5
20.
MA2
-14.

Notice preliminary:

The detailed interfaces of the utility routines are given to the paragraph [§2]. But them some lines of FORTRAN below can be included/understood without their reading.

```
% Declaration of the table on the TOTAL basis:
CALL TBCRSD (“T2”, “G”)
% Declarations of the parameters of the table and the types of their data:
CALL TBAJPA (“T2”, 1, “ACTION”, “K8”)
CALL TBAJPA (“T2”, 1, “NUME_ORDRE”, “I”)
CALL TBAJPA (“T2”, 1, “INST”, “R”)
CALL TBAJPA (“T2”, 1, “NODE”, “K8”)
CALL TBAJPA (“T2”, 1, “DX”, “R”)
CALL TBAJPA (“T2”, 1, “DY”, “R”)
CALL TBAJPA (“T2”, 1, “MESH”, “K8”)
CALL TBAJPA (“T2”, 1, “SIXX”, “R”)
% Addition of the lines in the table:
REAL*8 VR (3)
CHARACTER*8 VK (2)
INTEGER VI (1)
CHARACTER*16 LPARA1 (6), LPARA2 (5)
DATED LPARA1/“ACTION”, “NUME_ORDRE”, “INST”, “NODE”, “DX”, “DY”/
DATED LPARA2/“ACTION”, “NUME_ORDRE”, “INST”, “MESH”, “SIXX”/
%
VK (1) = action
IF action = intitule1
C nume_ordre = 1,2
VI (1) = nume_ordre
VR (1) = T = urgent (nume_ordre)
C node = N1, N2, N3
VK (2) = node
```

VR (2) = DX (node, T)
VR (3) = DY (node, T)
CALL TBAJLI (“T2”, 6, LPARA1, VI, VR, CBID, VK, 0)
CONTINUOUS
CONTINUOUS
ELSE IF action = intitule2
VI (1) = 5 = nume_ordre
VR (1) = T =instant (5)
C nets = MA1, MA2
VK (2) = mesh
VR (2) = SIXX (mesh, T)
CALL TBAJLI (“T2”, 5, LPARA2, VI, VR, CBID, VK, 0)
CONTINUOUS
ENDIF
Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Use of the structures of data counts
Date:
19/03/98
Author (S):
E. SCREWS, J. PELLET
Key:
D6.06.01-B
Page:
18/18
Intentionally white left page.
Data-processing handbook of Description
D6.06 booklet:
HI-75/97/032 - Ind A

Code_Aster ®
Version
8.1

Titrate:
Utilities for the Structures of data

Date:
01/12/05
Author (S):
J.** Key **PELLET
:
D6.07.05-B Page
: 1/10

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D6.07 booklet: -
Document: D6.07.05

Utilities for the Structures of data

Summary:

One presents in this document some utilities of interest general operating on structures of data: copy, destruction, existence, impression.

Utility DISMOI is used to extract "scalar" information (1 entirety or 1 text) in a Structure of Data.

1 Routine
COPISD

SUBROUTINE COPISD
(TYPESD, BASES, SD1, SD2)
drank:
to duplicate a structure of data (SD1) under another name (SD2).
SD2 will have the same contents as SD1.

IN TYPESD
K*
type of the 2 structures of data SD1 and SD2.
types allowed today:
"CHAMP_GD", "FUNCTION", "CHAM_NO_S", "CHAM_ELEM_S", "TABLE",
"RESULT", "VARI_COM", "CORRESP_2_MAILLA", "GRID"
IN K1 BASE
"G"/"V"/"It: name of the base where SD2 will be created
IN
SD1 K*
name of SD SD1
JXIN
IN
SD2 K*
name of SD SD2
JXOUT

2 Routine
IDENSD

LOGICAL FUNCTION IDENSD
(TYPESD, SD1, SD2)
drank:
to test the identity of the contents of 2 structures of data SD1 and SD2

IN TYPESD
K*
type of the 2 structures of data to be compared.
type allowed today: "PROF_CHNO"
IN
SD1 K*
name of SD SD1
JXIN
IN
SD2 K*

name of SD SD2
JXIN

OUT IDENSD L .TRUE. : 2 SD SD1 and SD2 are identical
.FALSE. : 2 SD SD1 and SD2 are different

3 Routine
EXISD

SUBROUTINE EXISD
(TYPESD, NOMSD, IRET)
drank:
to answer the question: “there exists a structure of data of the type TYPSD and of
name NOMSD? “.

IN TYPESD
K*
type of the structure of data to be tested.
types allowed today:
“FIELD”, “CHAM_NO_S”, “CHAM_ELEM_S”, “TABLE”, “RESULT”,
“CHART”, “RESUELEM”, “CHAM_NO”, “CHAM_ELEM”, “GRID”,
“MODEL”, “LIGREL”, “FUNCTION”, “MATR_ASSE”, “NUME_DDL”
IN
NOMSD K*
name of the SD to be tested
JXIN

OUT IRET
I 0: the structure of data does not exist
1: the structure of data exists
Data-processing handbook of Description
D6.07 booklet: - HT-66/05/003/A

---

4 Routine
DETRSD

SUBROUTINE DETRSD
(TYPESD, NOMSD)
drank:
to destroy a structure of data of the type TYPSD and name NOMSD.
I.e. to destroy all the objects JEVEUX which make it up.

IN TYPESD
K*
type of the structure of data to be tested.
types allowed today:
"NUME_DDL" "PROF_CHNO"
"MATR_ASSE" "VECT_ASSE"
"MATR_ELEM" "VECT_ELEM"
"VARI_COM" "FUNCTION"
"TABLE" "DEFI_CONT" "RESO_CONT"
"SOLVEUR" "CORRESP_2_MAILLA" "CLOUD"
"CHAM_NO_S" "CHAM_ELEM_S" "LIGREL"
"CHAM_NO" "CHAM_ELEM" "CHART"
"CHAMP_GD" "RESULT" "GRID"

IN
NOMSD K*
name of the SD to be destroyed
JXIN

5 Routine
IMPRSD

SUBROUTINE IMPRSD
(TYPESD, NOMSD, FICH, TITLE)
drank:
to print "lisiblement" a structure of data

IN TYPESD
*K**
type of the structure of data to be tested.
types allowed today:
“FIELD” “CHAMP_S”
IN
*NOMSD K**
name of the SD to be printed
JXIN

*IN FICH K**
name of the file for the impression: “MESSAGE”, “RESULT”,…
IN K TITLE*
titrate given to the impression.

6 Routine
UTIMSD

SUBROUTINE UTIMSD
(FICH, LEVEL, LATTR, LCONT, SCH1, IPOS, BASES)
drank:
to print “dirtily” a structure of data.
i.e. to print the “rough” contents of the objects JEVEUX which compose it.

Actually, one does not treat really structures of data: all are sought
the objects JEVEUX whose name contains a certain character string.
But it is that in general, all the objects of a SD have names starting
consequently chain.

*IN FICH K**
name of the file for the impression: “MESSAGE”, “RESULT”,…
IN LEVEL
I
desired level of impression:
0: impression of the only name of the objects.
1: for the collections, one will print only the 10 1st objects.
2: all the objects of collection are printed
-1: one prints a “summary” of the objects: only one line per object.
IN LATTR L
.TRUE. : one prints the attributes of objects JEVEUX
.FALSE. : one does not print the attributes of objects JEVEUX
IN LCONT L
.TRUE. : one prints the values of objects JEVEUX

.FALSE. : one does not print the values of objects JEVEUX
*IN SCH1 K**
character string allowing to select the objects to be printed.
Data-processing handbook of Description
D6.07 booklet: - HT-66/05/003/A

The declaration of this chain is very important (its length), because it
condition the number of found objects.
if SCH1=' TOTO' but that SCH1 is declared K19, only the objects will be printed
starting with "LOUSE" followed by 15 white.
IN IPOS I
position to which one will seek the 1st character of SCH1.
IN K1 BASE
name of the base JEVEUX on which one seeks the objects.
"G", "V",…
if '': one seeks on all the open bases.

Exemple1:

CALL UTIMSD ("RESULT", 2, .FALSE., .TRUE., FIELD (1: 19), 1, "V")

fact the "dump" of the field named FIELD and which is on the VOLATILE basis. The attributes are not printed
objects JEVEUX.

Exemple2:

CALL UTIMSD ("RESULT", 0, .FALSE., .FALSE. , ".DESC", 20, '')

writing the name of all the objects whose name contains chain ".DESC" in position 20.

7 Routine
DISMOI

7.1 Principle

This routine must avoid multiplying the sequences of programming necessary to recover information (entirety or text) in a Structure of Data (SD).

Example:

.
the name of the grid associated with a field,
.
the number of equations of a nume_ddl,
.
...

It is to some extent a form of "JELIRA" on the SD.

To recover name (MA) grid associated with the field (CH), one will make:

CALL
DISMOI ("F", "NOM_MAILLA", "CH", "FIELD", IBID, MA, IER)

Note:

.
One can extend this routine to "objects" which are not really SD. It is enough that one can name the object and associate a type to him. It is for example the case of sizes, type_elem and phenomenon,
.
certain SD are not really named because they are ***single****. It is for example the case of the catalogue of finite elements ("&CATA" cf [D4.04.01]) in this case the name of the object is unutilised.*

Titrate:
Utilities for the Structures of data

Date:
01/12/05
Author (S):
J.** Key **PELLET
:
D6.07.05-B Page
: 5/10

7.2 Syntax
of call

call DISMOI (C_M, question, nom_SD, type_SD, rep_i, rep_c, ier)

CM
K1
character of identification of the type of message which is transmitted to
UTMESS ("F" makes it possible to stop the execution in the event of error),
K question*
key word specifying the request,
nom_SD
*K**
name of a SD,
*type_SD K**
key word specifying the type of the structure of data: nom_SD,
rep_i
I
answer (when the answer is whole),
rep_c
*K**
answer (when the answer is a character string),
ier
I
code return of error,
ier = 0 whole is well
impossible ier request.

In a "normal" use (out of the routines of "title"): one puts a question which must have one *answer. One makes then:*

call
DISMOI ("F", question, nom_SD, type_SD, rep_i, rep_c, ibid)

and ibid is not tested

If the request fails, the stop brutal ("F") but that is translated a programming error.

7.3
List types recognized by DISMOI

Name of the type
Length
Routine
"CHART" K19
DISMCA
"CATALOGUE" K0
DISMCT
"CHAM_ELEM" or "RESUELEM"
K19
DISMCE
"CHAM_MATER" K8
DISMCM
"CHAM_NO" K19
DISMCN
"FIELD" K19
DISMCP
"LOAD" K8
DISMCH
"SIZE" K8
DISMGD
"UNKNOWN" K19
DISMIC
"INTERF_DYNA" K14
DISMLI
"LIGREL" K19
DISMLG
"MACR_ELEM_STAT" K8
DISMML
"GRID" K8
DISMMA
"MATR_ASSE" K19

DISMMS
"MATR_ELEM" or "VECT_ELEM"
K8
DISMME
"MODEL" K8
DISMMO
"NUME_EQUA" K19
DISMNE
"NUME_DDL" K14
DISMNU
"PHENOMENON" K16
DISMPH
"RESULT" K8
DISMRS
"TYPE_ELEM" K16
DISMTE
"TYPE_MAILLE" K8
DISMTM

Note:

.
the length of the names of typified objects kN is indicative: routine DISMOI supplements (or truncate) the name given by the user according to the associated type,
.

the name of routines DISMXX associated with the various types is given (in 3rd column) to allow programmers to add new possibilities,
.

***Regulate:** Losque one uses DISMOI, it is always necessary to call DISMOI and never them specific routines DISMXX.*

7.4
List possible questions

In the table below, one gives for each question:

.
the heading of the question (text in capital letters between quotes),
.
the type of the result of the question: I, K3, K8,…,
.
an explanation of the answer possible.

"BESOIN_MATER"
K3
"YES"/"NOT"
if the MODEL requires a CHAM_MATER
"CALC_RIGI"
K3
"YES"/"NOT"
Allows to know if one type_element can calculate "rigidity" (and thus if it is a "principal" element of modeling and not an element of "edge")
"CARA_ELEM" K8
name of the subjacent CARA_ELEM.
``: there is no subjacent CARA_ELEM.
"CHAM_MATER" K8
name of the subjacent CHAM_MATER.
``: there is no subjacent CHAM_MATER.
"COEF_MULT" I
value of the "multiplying" coefficient of the number of values of the CHAM_ELEM (for the variables intern)
"DIM_GEOM" I
2/3: dimension of the problem: 2D or 3D.

on the types: type_elem, ligrel, model

the answer can be:

1: all the subjacent type_elem are 1D (X)
2: all the subjacent type_elem are 2D (X, Y)
3: all the subjacent type_elem are 3D (X, Y, Z)

if it coexists several type_elem of different size:
120: mix 1D and 2D
023: mix 2D and 3D
103: mix 1D and 3D
123: mix 1D, 2D and 3D

if there are static substructures,
one adds 1000 while waiting
to know their real dimensions (1,2, or 3)
who cannot be found today.

on the grid type:
the answer is:
2: the grid aster read is of type "COOR_2D"
3: the grid aster read is of type "COOR_3D"

This value is that read in the file of grid: COOR_2D or COOR_3D.
Caution: it is not because COOR_3D which the grid is really 3D.
"ELAS_F_HYDR"
K3
"YES"/'NOT
if the CHAM_MATER uses for its behavior ELAS_XXX at least a function
hydration
Data-processing handbook of Description
D6.07 booklet: - HT-66/05/003/A

: 7/10

"ELAS_F_SECH"
K3
"YES"/'NOT
if the CHAM_MATER uses for its behavior ELAS_XXX at least a function drying
"ELAS_F_TEMP"
K3
"YES"/'NOT
if the CHAM_MATER uses for its behavior ELAS_XXX at least a function temperature
"ELEM_VOLU_QUAD"
K3
"YES"/"NOT"/"MEL":
"YES": All the elements of the MODEL are mechanical, voluminal and quadratic.
"NOT": All the elements of the MODEL are mechanical, voluminal and linear.
"MEL": There are elements of the MODEL mechanical, voluminal linear and the other quadratic ones.
"EXI_AMOR_ALPHA" K3 "YES"/'NOT
if the cham_mater refers at least a material has which has the CMP "AMOR_ALPHA"
"EXI_AMOR_HYST"
K3
"YES"/"NOT"
if the cham_mater refers at least a material has which has the "CMP" "AMOR_HYST"
"EXI_COQ1D" K3
"YES"/"NOT" if the MODEL contains finite elements of modelings COQU_C_PLAN or COQU_D_PLAN or COQU_AXIS
"EXI_COQ3D"
K3
"YES"/"NOT" if the MODEL contains finite elements of modelings COQU_3D
"EXI_ELEM" K3
"YES"/"NOT" if the MODEL contains finite elements (it can only contain static substructures)
"EXI_ELTVOL" K3
"YES"/"NOT" if the MODEL contains "voluminal" elements
"EXI_HYDRAT" K4
"NOT": the mechanical load does not contain a hydration
"EVOL": the mechanical load contains a evol_ther hydration

"CHGD": the mechanical load contains a field of hydration
"EXI_PLAQUE" K3
"YES"/"NOT" if the MODEL contains elements of plate: modelings DST/DKT or Q4G
"EXI_POUX" K3
"YES"/"NOT" if the MODEL contains elements of beam "to the LICE".
"EXI_RDM" K3
"YES"/"NOT" if the MODEL contains elements of r.d.m. (beam, plate or hull)
"EXI_SECHAG" K4
"NOT": the mechanical load does not contain drying
"EVOL": the mechanical load contains a evol_ther drying
"CHGD": the mechanical load contains a field of drying
"EXI_TEMPER" K4
"NOT": the mechanical load does not contain a temperature
"EVOL": the mechanical load contains a evol_ther temperature
"CHGD": the mechanical load contains a field of temperature
"EXI_THM_CT" K3
"YES"/"NOT" if the MODEL contains elements of modelings XXX_THM_CT
"EXI_THM_VR" K3
"YES"/"NOT" if the MODEL contains elements of modelings XXX_THM_VR
"EXI_TUYAU" K3
"YES"/"NOT" if the MODEL contains elements "pipe"
"MODELING" K16
name of MODELING associated with a MODEL.
If there are several MODELINGS in the MODEL, the answer is ''
"NB_CHAMP_MAX" I raising number of the sequence numbers of a SD RESULT.
"NB_CHAMP_UTI" I numbers sequence numbers used of a SD RESULT.
"NB_CMP_MAX" I
raising component count of a SIZE.
"NB_DDLACT" I
a number of active DDLS = a number of physical DDLS minus the number of constraints kinematics.
"NB_EC" I
a number of entireties necessary to code a size: nb_ec = nb_cmp_max/30
"NB_EQUA" I
a number of equations of a linear system.
"NB_GREL" I
"GRELS" in the LIGREL numbers.
"NB_MA_MAILLA" I numbers MESHS of the GRID.

"NB_MA_SUP" I
a number of additional MESHS of the LIGREL.
Handbook of Data-processing Description
D6.07 booklet: - HT-66/05/003/A

"NB_NL_MAILLA" I numbers NODES of LAGRANGE of the GRID. this number can be nonnull if the grid contains SUPER_MAILLES.
"NB_NO_MAILLA" I numbers NODES of the GRID.
"NB_NO_MAX" I
raising number of the NODES of the TYPE_MAILLES.
"NB_NO_SS_MAX" I numbers maximum nodes for a SUPER_MAILLE of the GRID.
"NB_NO_SUP" I
a number of additional NODES of the LIGREL.
"NB_SM_MAILLA" I numbers SUPER_MAILLES of the GRID.
"NB_SS_ACTI" I
a number of active substructures in a MODEL.
"NB_TYPE_MA" I
a number of TYPE_MAILLES in the catalogue.
"NBNO_TYPMAIL" I numbers nodes of a type_maille
"NOM_GD" K8
name of the SIZE.
"NOM_GD_SI" K8
name of the SIZE simple partner.
"NOM_LIGREL" K19
name of the LIGREL.
"NOM_MAILLA" K8
name of the GRID.
"NOM_MODE_CYCL" K8 name of the MODE_CYCL.

"NOM_MODELE" K8
name of the MODEL.
"NOM_NUME_DDL" K14
name of the NUME_DDL.
"NOM_OPTION" K16
name of the OPTION (catalogues) calculation.
"NOM_TYPMAIL" K8
name of a type_maille
"NU_CMP_LAGR" I
number of component "LAGR" in a SIZE.
"NUM_GD" I
number of the SIZE.
"NUM_GD_SI" I
number of the SIZE simple partner.
"NUM_TYPMAIL" I
number of a type_maille
"NUME_EQUA" K19
name of associated SD NUME_EQUA.
"PARA_INST" K3
"YES": if the CHART is a chart of FUNCTIONS depending on time
'': if not
"PHENOMENON" K16
name of the PHENOMENON associates a MODEL.
the PHENOMENON is single in a MODEL)
"PROF_CHNO" K19
name of the subjacent PROF_CHNO.
"SUR_OPTION" K16
name of the option "user" who "chapeaute" possibly the real option (i.e of catalogue) associated A the object. ex: "CHAR_MECA" for "CHAR_MECA_PESA_R"
"THER_F_INST" K3
"YES"/"NOT" if the cham_mater uses for its behavior THER_XXX with less one function of time
"STANDARD" K16
type of a concept of which one knows nothing ("UNKNOWN"):
'FUNCTION, "CHAM_ELEM", 'TABLE '', "EVOL_ELAS",…
"TYPE_CHAMP" K4
type of the field
"CART": CHART
"RESL": RESUELEM
"NOEU": CHAM_NO
"ELGA": CHAM_ELEM at the points of GAUSS
"ELNO": CHAM_ELEM with the nodes
"TYPE_CHARGE" K7

type of a LOAD
“MECA_RE”: real mechanics (AFFE_CHAR_MECA)
“MECA_FO”: mechanics function (AFFE_CHAR_MECA_F)
“THER_RE”: thermics real (AFFE_CHAR_THER)
“THER_FO”: thermics function (AFFE_CHAR_THER_F)
“ACOU_RE”: real accoustics (AFFE_CHAR_ACOU)
“ACOU_FO”: accoustics function (AFFE_CHAR_ACOU_F)
“TYPE_MATRICE” K7
type of the matrices
“SYMETRI”: all the matrices are symmetrical
“NON_SYM”: there is at least a nonsymmetrical matrix.
'': the subjacent size is not of type “stamps”
“TYPE_RESU” K16
type of a RESULT: “EVOL_THER”, “EVOL_ELAS”, EVOL_NOLI',…. or
“FIELD”

“TYPE_SCA” K3
scalar type
*“R”: real*8*
“I”: integer
*“It: complex*16*
*“K8”: character*8*
*“K16”: character*16*
“TYPE_SUPERVIS”
Standard K16 which the supervisor would give to a SD:
“CHAM_NO_DEPL_R”, “CHAM_ELEM_EPSI_R”,…

"TYPE_TYPMAIL" K4
type of a type_maille:
/"POIN"/"LIGN"/"SURFING" or "VOLU"
"Z_CST"
K3
"YES"/"NOT"
"YES": if all the nodes of the GRID have same "Z exactly" (3rd co-ordinate)
"NOT": if not

7.5
Count cross of the possibilities

In the table below, one gives for each type of Structure of Data:

- *the theoretical length of the names of the objects of this type,*
- *the list of the questions which one can put on this type.*

"CHART"
K19
"NOM_GD"
"NOM_MAILLA"
"PARA_INST"
"TYPE_CHAMP"
"CATALOGUE" K0
"NB_NO_MAX"

"NB_TYPE_MA"
"FIELD"
K19
"NOM_GD"
"NOM_LIGREL"
"NOM_MAILLA"
"NOM_MODELE"
"NOM_OPTION"
"NUM_GD"
"TYPE_CHAMP"
"TYPE_SUPERVIS"
"CHAM_ELEM"
K19
"COEF_MULT"

"NOM_GD"
"NOM_LIGREL"

or "RESUELEM"
"NOM_MODELE"
"NOM_OPTION"
"NOM_MAILLA"
"TYPE_MATRICE" "TYPE_SCA"
"TYPE_CHAMP"
"TYPE_SUPERVIS"
"CHAM_MATER" K8
"ELAS_F_TEMP"

"ELAS_F_HYDR"
"ELAS_F_SECH"

"EXI_AMOR_ALPHA" "EXI_AMOR_HYST"
"THER_F_INST"
"CHAM_NO" K19
"NB_EQUA"
"NOM_MAILLA"
"NOM_NUME_DDL"
"NUM_GD"
"PROF_CHNO"
"TYPE_CHAMP"
"TYPE_SUPERVIS" "NOM_GD"
"LOAD" K8
"EXI_TEMPER"

"EXI_HYDRAT"

"EXI_SECHAG"
"NOM_LIGREL"
"NOM_MAILLA"
"NOM_MODELE"
"PHENOMENON"
"TYPE_CHARGE"
"SIZE" K8
"NB_CMP_MAX"

"NB_EC"
"NOM_GD_SI"
"NUM_GD"
"NUM_GD_SI"

“NU_CMP_LAGR”
“TYPE_MATRICE”
“TYPE_SCA”
“UNKNOWN” K19
“STANDARD”
“INTERF_DYNA”
K14
“NB_CMP_MAX”
“NB_EC” “NOM_MAILLA”
“NOM_MODE_CYCL” “NOM_NUME_DDL”
“NUM_GD”
“LIGREL”
K19
“DIM_GEOM”
“EXI_ELEM” “NB_GREL”

“NB_MA_SUP”
“NB_NO_MAILLA”
“NB_NO_SUP”
“NB_SS_ACTI”
“NOM_MAILLA”
“NOM_MODELE”
“PHENOMENON”
“NB_MA_MAILLA”
“MACR_ELEM_STAT” K8
“NOM_MAILLA”
“NOM_MODELE” “NOM_NUME_DDL”
“GRID” K8
“DIM_GEOM”

“NB_MA_MAILLA”
“NB_NL_MAILLA”
“NB_NO_MAILLA” “NB_NO_SS_MAX” “NB_SM_MAILLA”
“Z_CST”
“MATR_ASSE” K19
“CARA_ELEM”
“CHAM_MATER”

“NB_EQUA”
“NOM_GD_SI”
“NOM_MAILLA” “NOM_MODELE”
“NOM_NUME_DDL”
“NUM_GD_SI” “PHENOMENON”
“SUR_OPTION”

"MATR_ELEM"
K8 "CARA_ELEM"
"CHAM_MATER"
"NB_SS_ACTI"
or "VECT_ELEM"
"NOM_MAILLA"
"NOM_MODELE" "PHENOMENON"
"SUR_OPTION"
"TYPE_MATRICE"
"MODEL" K8
"BESOIN_MATER"
"DIM_GEOM"

"ELEM_VOLU_QUAD"
"EXI_COQ1D"
"EXI_COQ3D"
"EXI_ELEM"
"EXI_PLAQUE"
"EXI_POUX" "EXI_RDM"
"EXI_THM_CT"
"EXI_TUYAU"
"EXI_ELTVOL"
"EXI_THM_VR"
"MODELING"
"NB_MA_MAILLA"
"NB_NL_MAILLA" "NB_NO_MAILLA" "NB_NO_SS_MAX"
"NB_SM_MAILLA" "NB_SS_ACTI"

"NOM_LIGREL"
"NOM_MAILLA"
"PHENOMENON"
"Z_CST"
"NUME_DDL" K14
"NB_EQUA"
"NOM_GD"
"NOM_MODELE"
"NUM_GD_SI" "PHENOMENON" "NOM_MAILLA"
"PROF_CHNO"
"NUME_EQUA" K19
"NOM_MAILLA"
"PHENOMENON" K8
"NOM_GD"
"NUM_GD"
"RESULT" K8
"NB_CHAMP_MAX"
"NOM_MAILLA"
"TYPE_RESU"
"TYPE_ELEM"
K16
"DIM_GEOM"
"MODELING"
"PHENOMENON"
"TYPE_TYPMAIL" "NBNO_TYPMAIL"

"NOM_TYPMAIL"
"NUM_TYPMAIL" "CALC_RIGI"
"TYPE_MAILLE" K8
"NBNO_TYPMAIL"
"NUM_TYPMAIL"
"TYPE_TYPMAIL"

Data-processing handbook of Description
D6.07 booklet: - HT-66/05/003/A

Code_Aster ®
Version
4
Titrate:
Utilities of management of the CHARTS
Date:
28/01/1999
Author (S):

J. PELLET
Key:
D6.10.01
Page:
1/6
Mechanical department and Digital Models
Index:
With
Diffusion:
Users - Developers
Data-processing handbook of Description
D6.10 booklet:
D6.10.01 document

Utilities of management of the CHARTS

1
Introduction

The charts are fields [D4.06.05] "constant by element". Formally a chart is one structure of data which puts in correspondence meshs and events of same size (for example MATER for the characteristics of material).

The charts are often used to store the data affected by the user on "zones" of its grid (orders AFFE_CHAR_MECA, AFFE_MATERIAU, AFFE_CARA_ELEM,…).

These fields can then be used like "entries" of elementary calculations [D3.02.01].

Four routines are at the disposal of the programmers to build CHARTS:

MECACT
:
to create a constant chart.
ALCART
:
"to allocate" a chart.
NOCART
:
to write a couple (size, zone_affectée) in a chart.
TECART
:
"to finish" a chart (optional operation).

2
To create a constant CHART: routine MECACT
CALL MECACT (chart, mocle, nommoa, nomgd, ncmp, licmp, ICMP, rcmp, K8cmp)
CHART
K19
in
name of the chart to be created.
jxout
MOCLE
K6
in
/“NETTED”:
one affects all the meshs of the specified GRID
in NOMMOA (1: 8)
/“MODEL”:
one affects all the late meshs of the LIGREL
specified in NOMMOA (1: 19)
NOMMOA
*K**
in
name of the GRID (or the LIGREL) on which is based the chart.
NOMGD
I
in
name of the size associated with the chart.
NCMP
I
in
a number of CMPS of the affected size
LICMP
L_K8
in

list names of the CMP of the affected size.
ICMP
L_I
in
list values of the CMPS of the affected size if this one is of "whole" type
RCMP
L_R
in
list values of the CMPS of the affected size if this one is of "real" type
K8CMP
L_K8
in
list values of the CMPS of the affected size if this one is of type "K8"

***Example**:*
licmp (1) = "DX"
licmp (2) = "DY"
rcmp (1)
= 1.0
rcmp (2)
= 2.0
CALL MECACT ("ma_carte", "netted", NETTED, "depl_r", 2, licmp, ibid, rcmp, '')
will cause to create a constant chart of the size "DEPL_R" for which, all them meshs of the grid will be affected by the values: (DX = 1.0, DY = 2.0).

3
To create a CHART by "zones"
3.1
Principle of creation by zones

A chart is an ordered list of couples (size, zone_affectée). When for example, one user written in his command file:
AFFE_CHAR_MECA (….
PRES_REP: (
ALL: "YES"
NEAR: 0.)
PRES_REP: (
GROUP_MA: (GM1, GM3)
NEAR: 2.)
PRES_REP: (
NET: (M7, M8)
NEAR: 7.)
PRES_REP: (

NET: (M9)
NEAR: 9.)
The chart of size "PRES_R" contains 5 zone_affectée and the sizes which are attached there.
It will be said that the chart was created "by zone". In the "large" made program:
CALL ALCART
allowance of the chart
CALL NOCART
ALL: "YES"
NEAR: 0.
"writing" of 1st zone_affectée in
CHART
CALL NOCART
GROUP_MA: GM1
NEAR: 2.
"writing" of 2nd zone_affectée in
CHART
CALL NOCART
GROUP_MA: GM3
NEAR: 2.
"writing" of 3rd zone_affectée in
CHART

CALL NOCART
NET: (M7, M8)
NEAR: 7.
"writing" of 4th zone_affectée in
CHART

CALL NOCART
NET: (M9)
NEAR: 7.
“writing” of 5th zone_affectée in
CHART

3.2
To allocate a CHART: routine ALCART

Call ALCART (chart, netted, nomgd, nasmax, nmamax)
CHART
K19
in
name of the chart to be created.
jxout
NETTED
K8
in
name of the grid on which the chart is based.
jxin
NOMGD
K8
in
name of the size associated with the chart.
NASMAX
I
in
raising number of zone_affectée of the chart.
NMAMAX
I
in
raising total number of meshs forming the lists of meshs of
zone_affectée
For the example of [§3.1] one will make:
Call ALCART (“ma_carte”, “mailla_1”, “depl_r”, 5, 3)

3.3
To write in a CHART: routine NOCART

Call NOCART (Chart, code, group, mode, nma, limano, limanu, nomlig, ncmp)
CHART
K19
in
name of the chart where one wants “to write”
jxvar
CODE
I

in
code zone_affectée:
+1: the whole of the meshs of the grid (ALL: “YES”)
-1: the whole of the late meshs of a LIGREL
+2: a GROUP_MA of the grid
+3: a list of meshs of the grid
-3: a list of late meshs of a LIGREL
GROUP
K8
in
used only if Code = 2
it is the name of a group of meshs of the grid.
MODE
K3
in
used only if code = ± 3.
mode = “NUM” if one is useful oneself of numbered meshs: LIMANU
mode = 'NOM' if one is useful oneself of named meshs: LIMANO
NMA
I
in
used only if code = ± 3.
it is the number of meshs in list LIMANU (or LIMANO)
LIMANO
L_K8
in
used only if code = + 3
it is the list of the names of the meshs of the grid which are affected by size.
LIMANU
L_I
in
used only if code = ± 3
it is the list of the numbers of the meshs which one affects.
NOMLIG
K19
in
used only if code = - 1 or = 3.
it is the name of the LIGREL where are defined the LATE meshs.
NCMP
I
in
it is the number of CMP which one wants to note in the chart.

Information not transmitted by arguments.
The description of the size which one wants to note in the chart makes via 2
objects of work which are allocated by ALCART:
Data-processing handbook of Description
D6.10 booklet:
Index A

CHART (1: 19)/".NCMP"
V (K8)
CHART (1: 19)/".VALV"
V (?)
(? = I, R, C, K8.)

·
in CHART (1: 19)/".NCMP", the programmer writes the name of the components of size which he wants to note.

·
in CHART (1: 19)/".VALV", the programmer writes the values of the components (in the same order as .NCMP).

3.4
Treatment of the example of [§3.1]

CALL ALCART (CHART, netted, "PRES_r", 5, 4)
CALL JEVEUO (CHART (1: 19) // ' .NCMP', "E", IANCMP)
ZK8 (IANCMP) = ' PRES'
CALL JEVEUO (CHART (1: 19) // ' .VALV', "E", IAVALV)
ZR (IAVALV) =0.
CALL NOCART (CHART, 1, "",""", 0, "", IBID,""", 1)
ZR (IAVALV) =2.
CALL NOCART (CHART, 2, "GM1", "", 0,""", IBID, '', 1)
ZR (IAVALV) =2.
CALL NOCART (CHART, 2, "GM3", "", 0,""", IBID, '', 1)

ZR (IAVALV) =7.
LIMANO (1) = ' M7'
LIMANO (2) = ' M8'
CALL NOCART (CHART, 3, "", "NAME", 2, LIMANO, IBID,"", 1)
ZR (IAVALV) =9.
LIMANO (1) = ' M9'
CALL NOCART (CHART, 3, "", "NAME", 1, LIMANO, IBID,"", 1)

3.5
Principle of overload

The principle of overload is applied for a CHART: in the list of the affected zones, one element of the list overloads the elements which precedes it in the list. That wants to say that if one mesh belongs to several zone_affectée different, the size which is associated is for him that associated the zone_affectée last which contains this mesh.
If in the preceding example, the M8 mesh belongs to 2 GROUP_MA GM1 and GM3, the pressure which him
is associated is 7.

Important remark:

*The overload is total for **all the** CMPS of the size.*
If one makes for example (chart of "DEPL_R"):
GM1 - >
DX: 1.
DY: 2.
GM3 - >
DX: 3.
DZ: 4.
The meshs of GM3 are affected by the size:
(DX: 3. DZ: 4.)
The meshs of GM1 (except those of GM3) are affected by the size
:
(DX: 1. DY: 2.)
One can want that the principle of overload is more "fine" and that meshs of the intersection GM1 and GM3 "profit" from the 2 assignments and that their associated size is:
DX: 3. DY: 2. DZ: 4.
This is possible thanks to routine TECART [§3.6]

3.6
Overload "fine": routine TECART

CALL TECART (chart)
CHART
K19
in
name of the chart "to be finished"
jxvar

This routine has appeller after the last call to NOCART.
This operation modifies the contents of the chart to take account of the rule of "fine" overload defined in the preceding paragraph.
Practically, one "extends" the chart on all the meshs, one determines on each one of them the size who is associated to him by a "fine" overload of the type:

M1: DX = 1.
DY =
2.0
order of the calls
+ M1: DX = 3.0
DZ = 4.0
with NOCART
M1: DX = 3.0 DY =
2.0
DZ = 4.0

In the second time, one recompacte the chart, by gathering the meshs which are affected by even size.

Intentionally white left page.

Author (S):
Mr. BOIN, *F.VOLDOIRE, P. MIALON*

Organization (S): EDF/IMA/MMN
Data-processing handbook of Description
D8.01 booklet: Presentation of documentation
D8.01.03 document
Graphic charter for the realization of the formulas
mathematics in documentation
of Code_Aster
Summary
After having identified the minimal general mathematical objects most commonly employed by community of the mechanics developing in Aster,
(
K
- J 3M - 2M + J ***C*** *+* ***K)*** ***X*** *=*
I
J I

K

() E
.g (P
I
N
)
i=1
one exposes the instructions of striking of the mathematical formulas which allow on the one hand one returned paper and
acceptable screen
! (T) - di (
v (T) grad T) = F (T)
and which, in addition, answers the criteria required in the international publications dealing with mechanics of the solid.
In documentation Aster, the mathematical formulas are developed under the Equation editor of Microsoft Word5 (version of "MathType Editor Equation" of Design Science Inc).

1
Spirit and carried
1.1
Constraints imposed by the projection of the numerical documents Aster on a media

Part of the instructions for the drafting of the formulas in the documents with the formalism Aster, has summer controlled by the concern to keep an acceptable esthetics and a legibility
independently of the media and the basic police force of the surrounding text.
In the current state of the art as regards physical representation of the formulas in the documents

electronic, in the absence of DTD (Description of the Type of Document to formalism SGML), those are comparable with drawings. They thus do not undergo reformating according to the media of consultation (paper, cathode ray tubes).
The electronic book comprises as many external files of formulas (drawings). Contents of these files comes to be posted with the consultation of the book to the site which it must have in the text. book comprises a table connecting the name of the file (the formula) and the position in the book.

1.2
Standards and recommendations Aster

*They indicate the manner of representing the types of the mathematical objects typographically them more frequently handled by the mechanics of the solid. The principle is the use of typographical enrichments Italic and **Fat** to typify these objects.*
The Aster writer will use of these recommendations which constitute a minimal representation acceptable by the community of the mechanics of the solid developing in Aster. They:

· *approaches returned the TeX trainer,*
· *takes as a starting point the the necessary rules to publish in the following reviews:*

-
Comp. Meth. Appl. Mech. Eng.
-
Int. J. Num. Meth. Eng.
-
ASME J. Appl. Mech.
-
Europ. J. Mech. A/Solids.

· *takes account of the possibilities and the limitations of the Equation editor of Microsoft Word5.*

What gives for example:
(
K
*- J 3M - 2M + J **C** + **K**) **X** =*
I
J I

K

() E
.g (P
I
N
)
i=1
(calculation carried out by operator DYNA_LINE_HARM [U4.54.02 §1])
! (T) - di (

v (T) grad T) = F (T)
(calculation carried out by operator THER_NON_LINE [U4.33.02 §1])

2

2
or 3

2
1
VM =
ij - tr (). ij
3

3

I, J 1
=
(calculation carried out by the operand INVARIANT of procedure POST_RELEVE [U4.74.03]).
Data-processing handbook of Description
D8.01 booklet: Presentation of documentation
HI-75/94/032/A

---

2 Typographical realization of the formulas in Aster

After having identified the mathematical objects selected, one enumerates enrichments which y apply, the police forces to be used, the bodies, the relative positions of the elements which compose them formulas (indices, exponents, symbols of relations, etc…).

2.1 Enrichments and mathematical types of objects

The table hereafter summarizes on the objects selected, the basic typographical achievements that it aster writer will employ as far as possible.
Type of object
Ital
Romanian
Fat
Maig
Police forces
Numbers
X
X
Times
Scalar variable
X
X
Times or Symbol (1)
Usual function
X
X
Times (2)
Function with value
X
X
Times or Symbol
scalar
Function with values
vectorial or
X
X
Times or Symbol (3)
tensorial
Tensor, Matrix,
vector (dimension 2
X
X
Times or Symbol (3)
and more)
Space scalars
X
X
DESCARTES (4)
or of vectors
Space functions

X
X
Monotype
Corsiva (5)
Text
X
X
Geneva (6)
1) If a Greek capital letter is employed for a scalar variable then to always strike it in Romain.
2) The Equation editor of Word5 can recognize the name of forty usual functions like: det, lim, cos, Im etc…
*3) For the Symbol police force, the **Fat** appears on the screen but not clearly with the impression. Example:*
(fat), (not fat).
4) Body of realities ¤, the complexes ç, the entireties ·. One can test difficulties of printing DESCARTES organizes when it is employed in the Equation editor. The printer replaces them characters DESCARTES by a white. Unknown remedy for the date of publication of this document. To address itself to the Person in charge for Documentation Aster.
5) For example: (F), (here Body 18) to note a space of functions, (P) a problem, (S) one system.
6) According to MacOS and the versions of Word5 and the Equation editor which one lays out it is possible that Geneva in a "text" of formula left on the printer in Courier. To prefer then Helvetica which does not present this disadvantage.

Caution

It results from 4 and 5 that the operating systems MacOS of the Aster writers will have to be gréés by these police forces.

Data-processing handbook of Description
D8.01 booklet: Presentation of documentation
HI-75/94/032/A

2.2
Examples for the functions
Dim.
Writing of the application
Physical examples
spaces

¤ ¤
F (*X*) = *B F*
E (*T*)
YOUNG modulus function of the temperature
N
F (**T**) = *B F*
G (**S**) = *y*

¤ ¤
N
m
F (**T**) = **V** = **F**
K (**S**) geometrical Rigidity

¤ ¤
m
F (*has*) = **T** = **F**
With (*T*) Elasticity function of the temperature

¤ ¤
2.3
Body of the components of the formulas
Elements of the formula
Body

Examples
Normal terms (*)
12 Pt
Exponents and indices
9 Pt
Symbols
18 Pt
Under symbols
12 Pt
(*) If one uses Monotype Corsiva for a normal term, to prefer the body 14 Pt.
That is to say the adjustment following in
the heading

menu
of the Editor
mathematical formulas
Data-processing handbook of Description
D8.01 booklet: Presentation of documentation
HI-75/94/032/A

2.4
Relative positions of the elements of a formula
It is necessary to understand by there, the relative position of the indices and exhibitors compared to the term which they affect
and the relative position of the lines of equations or the lines and columns of matrices. They are taken default values of the equation editor of Microsoft Word5 expressed hereafter in % of the body symbols.
That is to say the adjustment following in
the heading

menu

of
the Editor of formulas
mathematics
2.5
Style sheet for the formulas
Heading

menu
of the Editor
mathematical formulas
2.6
Spaces on both sides of the sign =
One recommends to isolate the sign well = while laying out
sufficient white on both sides of the sign. **Drank**:
to make quite readable the two members of the equations. One
recommend to add to spacing by affected defect
automatically by the Equation editor after the sign
of relation = a white of a quadratin.

2.7
Texts in the formulas
If the author wishes to accompany his formula by a text (what is disadvised) for, for example,
to clarify certain terms, this text will be in Geneva 10 Romain nonfatty Style "Text" of the sheet of
style of the Equation editor (with the reserves expressed in [§2.1]). In this case, the unit formulates
+ text forms only one graphic block.
(
K

- *J* 3M - 2M + *J* **C** + **K**) **X** =
I
J *I*

K

() E
.g (*P*
I
N
)
i=1
where **C** = Matrix of Damping

2.8
Formulas except text and in text

The typography of the terms of formulas integrated in a paragraph is the same one as in the formula it even. An example is given in [§3.6].

3
Recommendations and advice

3.1
Notations author --> reader

At the head of document the writer will expose his notations, mainly in what they differ or supplement the *Aster* recommendations. It will take care to choose a symbolism present in the Editor equations of Word.

3.2
Notations author --> typist

The writer will indicate on his manuscript, by a code with him the instructions of enrichment of terms of its mathematical formulas.

3.3
The “transposed” sign

Transposed of a matrix or a vector (and opposite of matrix) as follows:

MT M-1 M - *T* **xT**
,
,
,
. Modal mass for mode *I*: **driven C**
I
I

3.4 Tiny
Greek

In the Symbol police force one will préfèrera the tiny phi with to avoid confusions

Data-processing handbook of Description

3.5
Functions and variables

Not to confuse the function and its realization for a given value of its variable.

To always indicate what depend the functions the first time that the function appears. Example:

1
G (,) =
(- (tr)) 2
Id - (
y) (Criterion of plasticity)
3

3.6 Derived

To indicate **where** are taken the derivative, at least during their first appearance. It is recommended following formalism:

that is to say the function G (,), its partial derivative compared to for = and = is written:

G
(,)

or this one

ij, *J* + fi = 0

for an equilibrium equation.

3.7
Convention of the repeated indices

In a indicielle notation, one will use the convention of EINSTEIN known as **“of the repeated indices”**. This
convention, makes it possible to reduce the writing and to be freed from employment from the symbol from summation.

Principle: an index repeated twice, once in top, once in bottom, or more simply
twice in bottom, a summation (1,…, N) indicates automatically.

N
Example: **v**
= *life*
I
I
= **v I.E.(internal excitation)**
i=1
v, vector
VI, component
I.E.(internal excitation), basic vector
tr = *K*
1
2
3
K = 1 + 2 + 3
tr = trace of the tensor

= Id. = *ij*
K
ij
= *K*
3
3
. = *ij*.
ij
ij

= .ij or more simply *ij*.*ij*.
i=1 j=1

3.8

Greek indices and Latin indices

One advises the use the index Greek (, etc…) for a course in the interval {1, 2} and them Latin indices (*I*, *J*, *K*, etc…) in the interval {1, 2, 3}.

3.9

Alignment and balance of the equations

To adopt a provision such as the similar terms are on the same balance.

0
Z
µ
0
2
µ

= **A** (**E** (**U**)

- *Z* **K**
3
(**U3**) + **Aij** (**E** (**U**)
µ
ij () + **Kµ** (**U3**) *ij* ()

+ *Z*
2
0
réf
ij (**Udil**) - *kl* (*T* - *T*
) *ikjl*) + (*O*)

0
Z
µ
0
Z
µ
33
= **A**
33
(**E** (**U**)

- *z3* **K** (**U3**) + **A33ij** (**E** (**U**)

μ
ij () + **K** μ (**U3**) *ij* ()

+ *Z*
réf
ij (2
Udil) - *kl* (0
T - *T*
) *ikjl*) + (*O*)
Data-processing handbook of Description
D8.01 booklet: Presentation of documentation
HI-75/94/032/A

---

4 Examples

These examples are extracted from the isotropic form of thermoelasticity.

1
D
ij
= *ij* - *kkij*
3
2
éq
3D *D*

3**.** - (tr)
1

2
VM
=
. =

=
(*I* - *J*)
2
2
2 *I*, *J*
2
2
D *D*
3 (*éq*
.
=
VM)

4.1

Thermodynamic potential, density of free energy 3D

1
1 *C*
2
F (, *T*)
(tr) 2 µ**.** - 3K (*T* - *Tréf*) tr -
(*T* - *Tréf*
=
+ *ij* *ij*
)
2
2 *T*
K
2
1 *C*
2
F (, *T*)
(tr) µ *D.D*
réf
réf
=
+
ij - 3
K (*T* - *T*) tr -
(*T* - *T*
ij
)
2
2 *T*

Stability: positive definite potential:

μ > 0; 3K =

3 + 2μ > 0 *E* > 0; - 1 > > 0 5
,
4.2
Complementary potential, density of enthalpy free 3D
-
1
2
+

1 *C*
2
F (
*, *T*)
(tr) +
. + (*T* - *Tréf*) tr +
(*T* - *Tréf*
=
ij
ij
)
2nd
2nd
2
2 *T*
1
2
1
D
D

1 *C*
2
F (
*, *T*)
(tr)
.
réf
réf
=
+
ij +
(*T* - *T*) tr + (*T* - *T*

ij
)
18K
4µ
2
2 *T*
Data-processing handbook of Description
D8.01 booklet: Presentation of documentation
HI-75/94/032/A

4.3
Coefficients of elastic rigidity 3D

F
= *ij* = *ijkl*
réf
ij
ij kl
ik
jl
réf
ij
kl + (*T* - *T*
) *D* = (+2µ) *kl* -3K (*T* - *T*)
ij (*T*,)

4.4
Relations stress-strains 3D

réf
ij
= *kkij* + 2µij - 3K (*T* - *T*) *ij*

E

E

réf
ij
=
ij +
tr *ij*
(*T* - *T*)
1 +

1-
2

- 1-
ij
2
11
+ 2µ

0
0
0 11
1

22

+ 2µ

0
0
0
22
1

33

+ 2µ 0
0
0 33
1

=
•
-
3 *K* (*T* - *Tréf*)

12
0
0
0
2µ
0
0
12
0

23
0
0
0
0
2µ

0
23
0

31
0
0
0
0
0
2

µ 31
0

4.5
Relations deformation-constraints 3D

-
1 +

réf
ij
=
kkij +
ij + (*T* - *T*) *ij*
E
E
11
1
- -
0
0
0 11
1

22
-
1
-
0
0
0
22
1

33
1 - -
1
0
0
0
33
1

=

+ (*T* - *Tréf*
•
)

12
E
0
0
0
1 +
0
0
12
0

23
0
0
0
0
1+
0
23
0

31
0
0

0
0
0
1

+

31
0

4.6
Elastic plane constraints 2D
11
1
0 11
1

E

E

22
=
1
0

-
T - *T réf*
•
1

2
22

(
)
1 -
1-

12
0 0 1

- 12
0

=
réf
COPL

+ (*T* - *T*
) *COP*
D
L
E
1-

E
=
+
2
2 (
+)
réf

-
(*T* - *T*)
1-

1-
4.7
Complementary potential 2D
1 - 2
2
1 +

F *
()
2
DEPL
=
(2trD) +
(12 -11.22)
2nd
E

Intentionally white page.

Organization (S): EDF-R & D /SINETICS

Data-processing handbook of Description
D9.02 booklet: -
Document: D9.02.01

Code_Aster: the maintenance of Eficas and of supervisor

Summary:

This document is with the use of the developers of the supervisory core of Aster. It clarifies the interpretation of the file
data of the user (construction of the command set) and the sequence of the executions.
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Code_Aster ®
Version
6.4

Titrate:
Code_Aster: the maintenance of Eficas and the supervisor

Date:
02/04/04
Author (S):
***C. CAREMOLI**, A. YESSAYAN Key*
:
D9.02.01-A Page
: 2/34

Contents

---

1 Introduction

Numerical study

To make a numerical study with Code_Aster, the end-user starts the functionalities of code and provides information necessary to the execution of these functionalities.

A functionality is selected via an order. An order is initially a name i.e. a character string or an external identifier (known of the user) of functionality. It also incorporates attributes and initially the identifier interns functionality (exploitable by code FORTRAN).

Information necessary to the launching and the execution of the order otherwise-known as them parameters of the digital processing, are data introduced by means of key words them identifier. For a given order, a certain number of parameters must be defined.

The description of the orders and the key words is carried out by the developers of Code_Aster. in a file called “catalogues orders”.

The end-user uses the code via a file called “command file”. He provides to it orders whose composition must be compatible with their description in the catalogue of orders. The orders are numerous but the number of associated data is much more important with possible combinations themselves very many.

Finally the computer code stores information (orders and key words) in a structure of data interns called “command set”.

The role of the supervisor

The supervisor is the part of Code_Aster which manages the command set. In particular:

· it imports the catalogue of the orders in the Python memory,
· it charges the orders and their key words in the Python memory,
· it carries out the treatment orders by order,
· it provides - at the request of FORTRAN the value of the parameters to the functionalities of Code_Aster. stored in the memory Python. For that the supervisor proposes API C.

The graphic interface Eficas

It is possible to define “manually” its command file, for example by means of one text editor. However, the user nonfamiliar with syntax of the orders which it handles but also the language python will be able to use Eficas.

The graphic interface Eficas is intended for the end-user of Code_Aster. It allows him to build valid orders with statically such validated associated key words them, then to generate a command file bound for the code. The user has the insurance that the file product by Eficas has a correct syntax.

The core

The sources common to the Eficas interface-graph and the supervisor of Code_Aster are organized (identified and gathered) in a specific space of called development *the “core”. The comprehension of these sources is obviously a condition necessary for to undertake the maintenance of the two tools which use them.*

Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Code_Aster ®
Version
6.4

Principal actors

The principal types of actors evolving/moving in the environment of Eficas and the supervisor are listed below.

· The end-user: he rather knows the physical part of the problem to be solved and he has as a task to introduce valid data into the code, of launching calculations and stripping them results; it can use the graphic interface Eficas for that.
· The developer of functionalities in Code_Aster : it practises primarily programming FORTRAN of the numerical algorithms and, in the case of an macro-order, the drafting of script corresponding Python; it enriches and/or modifies the catalogue of order and it uses API C of the supervisor to recover in his code FORTRAN, them values provided by the end-user.
· The developer of the supervisor in Code_Aster: it writes scripts of reading and of interpretation of the catalogue of orders and command set; it is also occupied module python aster written out of C (astermodule.c) which serves as API to the supervisor.
· The person in charge for the maintenance of the code, the environment of use and management of configuration: he centralizes the sources. In addition to code FORTRAN, these sources include/understand those of the catalogue (Python) of the orders, scripts (Python) of the supervisor and those of modulate aster.
· The developer of the graphic interface Eficas develops a man-machine interface. Its role is to conceive and program dialog boxs as well as sequences of event allowing the taking into account of the requests of the end-user and the checking at every moment - validity of the command set in the course of construction.

In this document, one rather stresses the principles of the design of the core that on the detail scripts which one cannot save the examination.

Covered subjects

The first chapter proposes a summary definition of the terms used abundantly in the continuation document.

Chapter 2 presents few conventions and the organization of the source files in repertories: packages. It also describes the environment necessary to the development and operation of the supervisor of Code_Aster and graphic interface Eficas.

In chapter 3, some recalls are carried out on the language of Python script. They must direct the future person in charge for maintenance towards certain techniques that it will be necessary for him to control for to carry out its task.

The important subject of the factory of order is evoked in chapter 4. It is a base necessary to comprehension of all the structure of the command set.

Lastly, an answer is brought to chapter 5, some questions which one can provide that all future person in charge for maintenance will be able to be posed.

1.1 Definitions

Mode "by batch"

The treatment of the orders user by Code_Aster can be carried out according to two modes.

In the first mode, the file order is charged in memory to create the command set.

This creation of the jdc makes it possible to validate syntax python (brackets, commas), Aster syntax (coherence with the catalogue) and to validate the last concepts in argument. Finally after this checking, the command set is traversed to carry out the digital processings correspondents.
This first mode is called "mode by batch". The end-user selects this mode while specifying value "YES" for key word PAR_LOT under the order BEGINNING.

In the second mode, the command file is charged in memory to create the play of orders. Then the stages (equivalent to the orders) are built and carried out sequentially.
The end-user selects this mode by specifying the value "NOT" for key word PAR_LOT under order BEGINNING.

By defect, the mode used is PAR_LOT=' OUI'.

Operator of Code_Aster

An operator is a unit of fascinating Code_Aster in load a functionality of the code. Concretely it is a subroutine FORTRAN whose name is numbered, for example the subroutine OP0001 which charges a grid in the memory of the application. The classification of the operators facilitate association between their internal representation (subroutine FORTRAN) and their representation external for the end-user (order).

Order

An order is a character string identifying a numerical operator. It thus allows the end-user to start the execution of this operator starting from a data file called "command file".

There are 4 types of orders: OPER, PROC, MACRO and FORMULATE.

The developer of the numerical operator defines - in the catalogue of the order - control characteristics and those of the key words corresponding to the parameters of the operator numerical:

· its name (the character string usable by the end-user),
· its rules of composition in key words,
· an explanatory French and/or English comment,
· the key defining the handbook and the chapter devoted to this key word in the documentation of Code_Aster.

Order OPER

In addition to the attributes enumerated above, an ordering of type OPER has the following attribute:

· The type of the structure of data Aster produced by the operator and turned over by order;

Order PROC

An ordering of type PROC to the characteristic not to turn over a value. This characteristic put aside, it has the same attributes as an ordering of type OPER.

MACRO order

Macro is a function written in Python by the Aster developer - which calls orders i.e. operators - of Code_Aster. It stores results which could be recovered via the supervisor.

The text the macro one can be public; in this case it is stored in a specific file of macro sub-directory. If it is deprived, it is placed or imported in the command file.

An ordering of MACRO type makes it possible the end-user to use the macro one. For example order MACRO_MATR_ASSE allows to launch the macro public macro_matr_asse_ops.

Catalogue of an order

The catalogue of an order is the whole of the Python instructions describing the definition of order i.e. the values assigned to the attributes of the order.

The catalogue of an order is written by the developer of the numerical operator associated *order.*

Catalogue general

The catalogue general is a Python file containing the description of all the orders otherwise-known as containing the catalogues of all the orders.

Command set

The command set is the structure of data - organized in a Python object containing the whole of the information provided by the end-user, in the command file.

Command file

The command file makes it possible the end-user to start the numerical operators carrying functionalities of Code_Aster via the orders.

Structure of data Aster

A structure of Aster data is an organization of data produced by an operator numerical of Code_Aster. It is identified by a type itself declared at the beginning of the catalogue (cata.py); what makes it possible symbolically to use it in the command file although it that is to say produced by FORTRAN.
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Simple key word

A simple key word is a character string identifying a data used in entry by one *operator (a numerical functionality of Code_Aster). A simple key word is thus defined inside of an ordering of Code_Aster.*

*The end-user will be able to provide a **value** to the parameter of an order via the name simple key word corresponding in the command file.*

*The developer of functionality of Code_Aster will define as for him, the **characteristics** of the key word simple in the catalogue of the order containing the simple key word:*

· its name (the character string usable by the end-user and the numerical operator),
· the type of the parameter (whole, real, text, concept,…),
· the statute of the simple key word (optional or obligatory),
· the default value to be assigned to the parameter,
· the minimum number of data which the end-user will have to provide behind the simple key word,
· the maximum number of data which the end-user will have to provide behind the simple key word,
· an explanatory French and/or English comment.

The supervisor of Code_Aster charges in the memory of the application, the characteristics of the key word
simple, starting from the catalogue of the orders. Then it charges (and checks) possibly the value of parameter of the order starting from the command file provided to the application by the end-user.

The numerical operator of Code_Aster questions the supervisor via the API getvxx to recover value of the parameter starting from the name of the key word. The supervisor turns over the value provided by
the user or the default value of the parameter.

Key word factor

A key word factor is a character string identifying a group of simple key words semantically associated. A key word factor is defined inside an order. An order can contain several key words possibly optional factors., each key word factor containing itself of the simple key words of the same name.

*The end-user will be able to define in his command file, a key word factor by specifying sound name then its **value** i.e. the value of the definite numerical parameters behind the key words simple of the key word factor.*

*The developer of the functionality of Code_Aster defines the **characteristics** of the key word factor in the catalogue of the order containing the key word factor:*

· its name (the character string usable by the end-user and the numerical operator),
· its rules of composition in simple key words,
· the statute of the key word factor (optional or obligatory),
· the minimum number of repetition of the key word factor,

· the maximum number of repetition of the key word factor,
· an explanatory French and/or English comment,
· the key defining the handbook and the chapter devoted to this key word in the documentation of Code_Aster.
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Conditional block

A conditional block (a block), associates:

· of the simple key words,
· of the key words factors,
· and of the conditional blocks.

The occurrence of the block in its order, depends on a condition expressed at the time of the definition of order by the developer of the numerical functionality.

The developer of the operator corresponding to the order containing the block, specifies them characteristics of the block:

· its name,
· its condition,
· its rules of composition in simple key words,
· an explanatory French and/or English comment.

The end-user will be able to give a value to the parameters of the treatment by using the key words factors and key words simple associates in the conditional block but without specifying the name of block.

Regulate composition

*The composite entities of the catalogue of orders such as "command set", orders,
key words factor and conditional block, structure of other entities while possibly following one or several rules of composition among the following ones:*

AU_MOINS_UN
Rule AU_MOINS_UN expresses that one at least entities whose names passed in arguments must be present in the composite entity in which figure the rule.
UN_PARMI
Rule UN_PARMI expresses that one and only one of the entities of the entities whose names are passed in arguments must be present in the composite entity in which figure the rule.
EXCLUDED
The rule EXCLUDED expresses which if one of the entities whose name passed in argument, is present, the entities corresponding to the other arguments must miss in the entity composed in which figure the rule. Otherwise-known as if several entities of the group are present the rule is violated.
TOGETHER
The rule TOGETHER expresses that if one of the entities whose name passed in argument is present in the composed entity, then all those corresponding to the other names will have to be it too. The order of the occurrences does not have importance. And if none the entities represented in the rule is not present in the composite entity, the composite entity is valid.
PRESENT_PRESENT
Rule PRESENT_PRESENT expresses that if the entity corresponding to the ***first*** *name is present, then all those corresponding to the other names will have to be it also in the composite entity current. The order of occurrence of the other entities does not have importance. If none the entities represented in the rule is not present, the composite entity is valid.*

PRESENT_ABSENT
*Rule PRESENT_ABSENT expresses that if the entity corresponding to the **first** name is present, then all those corresponding to the other names will have to miss in the composite entity current. The order of occurrence of the other entities does not have importance. If none the entities represented in the rule is not present, the composite entity is valid.*

Each rule of composition (called also simply "rule") is a class (see the module regle.py).

1.2
Assumption of responsibility of maintenance

The following step is proposed to the candidate with the maintenance of the graphic interface Eficas and/ or of the supervisor of Code_Aster.

· Etudier "Accas" i.e. what is common to the graphic interface Eficas and to supervisor of Code_Aster;
· Etudier the structure of the catalogue general of the orders: in the file catalogues and in memory of the application. For that:
- to familiarize itself with the techniques of programming in Python, used in Accas;
- to develop a small model of factory (cf [§4.1]) for integrating the mechanism well of base loading of the key words.
· Etudier the structure of the command set (in its file) and memory; in particular the question of the loading of the command set (mechanism and zones of codes concerned) must be considered;
· Examiner scripts Python or the sources C brought into play at the time of requests of modification or of treatment of the errors detected by the users.

2
The organization Python sources

2.1 Conventions

Following conventions the purpose of which are to facilitate the reading of scripts, are imperfectly applied

· a name of class starts with a capital letter;
· the identifier of an object of the type Python list starts with the prefix l_ (this rule is used in Eficas);

*· in the packages used by the supervisor (Core, Execution, Validation, Build and Accas) only one class is defined by module i.e. by file *.py;*
· in the packages used by the supervisor (Core, Execution, Validation, Build and Accas) the name of each module starts with a prefix indicating the name of the package.
- N_ for Core
- V_ for Validation
- E_ for Execution
- B_ for Build
- A_ for Accas

2.2
Typology of the modules Python

Each class is defined in a module: for example, class MCSIMP is defined in the module N_MCSIMP.py where the N_ prefix indicates the name of the package (Core) containing the module.

The technique of the packages makes it possible to cut out the modules according to the sphere of activity in
which it acts. With each field corresponds a Python package

For example, class MCSIMP exists in each of the five packages,

· · Noyau: N_MCIMP.SIMP;
· · Validation: V_MCSIMP.SIMP;
· · Ihm: I_MCIMP.SIMP;
· · Accas: A_MCIMP.SIMP;
· · Build: B_MCIMP.SIMP;

***Core** This package contains primarily the system of class of the factory of the command set.*

Validation
This package contains the modules carrying out the checks of validity of the objects (orders, conditional blocks, key words,…).

Build This package is present only in the supervisor contains the modules treating the orders of macro type and methods of request to the command set since the API-C: interfaces GETVxxx.

Accas
This package is most important. It contains in particular - the classes more including used as well by the graphic interface Eficas as by the supervisor of Code_Aster.
It is in this package that it is necessary to seek the objects and the methods specialized in nongraphic treatment of the orders:
· loading of the catalogue;
· loading of a command set;
· execution of the command set.
The classes girls defined in this package are it by heritage of classes relationships having it even name that the classes girls but being located in a different package.

Ihm
The classes of this package enrich the classes by the Core of methods nonrelated to the graphic aspect used by the graphic interface Eficas.

Editor
This package contains the modules of graphic treatment of the command set.
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

2.3
Hierarchy of the repertories of the sources

2.3.1 The python sources of the supervisor

The supervisor of Code_Aster is composed of written modules python:

· out of C for the applicative interface of the supervisor: the module aster (astermodule.c);
· in Python for the management of the command set (loading, organization,…).

The diagram below presents the repertories containing the Python sources of the supervisor.

Eficas

Core
Accas
Validation
Build
Execution
Macro

2.3.2 Sources of the Eficas interface-graph

The sources are organized in repertories under the repertory of Eficas installation:

Eficas

Core
Accas
Validation

Editor
Ihm

Extensio

Macro
Misc
Tools

2.4 Environment

This paragraph describes the pre-necessary conditions with the operation of the Supervisor and the interface
Eficas graph.

2.4.1 Installation of the interpreter Python

To inform

2.4.2 Shell parameters of configuration

To inform

2.4.3 Installation of the Eficas interface-graph

To inform

2.4.4 Update of the supervisor

To inform

3
Use of the language Python in Eficas and in
supervisor of Code_Aster

One presents here the techniques of programming in Python language, whose control is necessary to any candidate with the maintenance of the graphic interface Eficas and the supervisor of Code_Aster.

3.1
Call of a function with a variable number of arguments

The end-user of Code_Aster uses a Python script to start the functionalities and to provide values with the parameters of the functionality. Supply of these values being sometimes optional, the Accas core uses the mechanism envisaged in the Python language to pass a variable number arguments with a function.

To maintain Accas, it is, consequently, necessary to control this technique of which us let us present a small example below.

```
# script main.py
def fonc (a number, *tup_args, ** d_args):
print a number
print repr (tup_args)
print repr (d_args)

fonc (11111, “arg 2”, “arg 3”, 4, n=5, j=6)
```

Under Unix, the interpretation of script main.py is done by:

```
$ python main.py
```

It gives the following result on the standard exit:

```
11111
(“arg 2”, “arg 3”, 4)
{“: 5, “I: 6}
```

Only the argument a number is obligatory. The possible positional arguments according to are stored in a tuple (which can be empty) and the possible arguments passed by key word are stored in one dictionary.

This technique is used in particular, by the objects which build the command set in memory then which initialize it.

3.2

Use of spaces of name

3.2.1 Concept of space of names

A space of names (see [bib1], page 97) is a Python dictionary containing a whole of couples name/value. The name is in general a character string and the value can be a value numerical, a function or an object.

In a module Python, each instruction is carried out in a called specific space of names ***space names room*** *whose contents can be posted by the function locals (). Instructions also access to the* ***total space of names*** *has whose contents can be posted with the function globals ().*

Accas uses in particular, a space of names to store the dictionary of the definitions which will be used to interpret the key words and to charge their value in the memory.
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

3.2.2
modulate
__builtin__

A remarkable module is the standard module __builtin__ which is except particular case ([bib2], page 100) referred in each module user by the attribute __builtins__ ***in mode*** ***ReadOnly only****.*

The interpretation of the following sequence:

 # script main.py

print globals ()

post

{"__doc__": None, "__name__": "__main__", "__builtins__": <module'__builtin__ '(built-in) >}

The module __builtin__ is imported by defect; all data and the functions which it contains are thus accessible by defect, in all the modules of the application. Data and functions defined in the module __builtin__ can thus be regarded as most total with the application.

Inter alia, one finds in this module, the following tools:

· the functions locals (). and globals ();
· the variable __debug__ which conditions interpretation according to its value 1 (defect) or 0;
· the function "builtin" __import__.

```
# script main.py
# One installs the module context in the total space of the interpreter
# (__builtin__)
# under name CONTEXT in order to have access to the functions
# get_current_step, set_current_step and unset_current_step of anywhere
importation context
importation __builtin__
__builtin__.CONTEXT=context
```

It is possible to import and enrich the module __builtin__; this technique is used in Eficas in Noyau/__init__.py script.

3.2.3 Execution of an order Python in a space of names

In the following example, variable GB defined at the beginning of script is accessible in all the units module: it is defined in the total space of names.

```
# script main.py
gb=2
def fonc (has):
b=gb*a
print "locals () =", locals ()
print "globals () =", globals ()
return B
x=123
z=-1
```

u=fonc (X)
print z+u

The interpretation of script main.py posts:

locals () = {"B": 246, "has": 123}
globals () = {"__doc__": None, "fonc": <function fonc At 0x810aee4>, "Z": -1, "X": 123, "__builtins__": <module "__builtin__" (built-in) >, "__name__": "__main__", "GB": 2}

A specific space of names can be created and used to carry out instructions stored in a character string.

```
# script main.py
d_contexte= {"has": 1, "B": 2}
print d_contexte
s_commande=' x=a+b'
exec s_commande in d_contexte
print d_contexte
```

When it interprets script main.py, the interpreter enriches space by names d_contexte with "X": 3 the result of the instruction s_commande. But the local space of names does not contain the result the instruction s_commande.

During the use of the instruction exec, one can specify that the data created must be it in the local space of names as the example shows it below.

 # script main.py
 # script main.py
d_contexte= {“has”: 1, “B”: 2}
print d_contexte
s_commande=' x=a+b'
exec s_commande in d_contexte, locals ()
print “x=”, X

what posts:

{“B”: 2, “has”: 1}
x= 3

The instruction Python exec makes it possible to create a space of names by the interpretation of a Python script -
stored in a character string - by specifying a space of names behind the key word in, by
example: exec s_script in space

3.3
A second manner of importing a module

The function “builtin” __import__ ([bib2], p. 100), makes it possible to import a module starting from its name
stored in a character string. In fact, this function is called by the Python interpreter at the time
importation of a module by the instruction importation ([bib3], p629). This technique is used with several
recoveries in Eficas (packages Editor and Extensions).

nom_module=' string'
print to dir ()
module_string =__import__ (nom_module)
print to dir ()
print to dir (module_string)
print module_string.lower (“ABCD”)

One notices that the function __import__ turns over in module_string, a reference on
modulate string from which it is possible to reach the tools contained in the module.

The function "builtin" __import__ can also be used to import a module specific to leave of a parcelling (package). The following example imports the catalogue general of the orders of Code_Aster then posts on the standard exit the name of each order.

```
importation sys
TOP='/local/yessayan/Eficas/EficasPourSalome' # repertory of installation
sys.path.append (SIGNAL)
sys.path.append (TOP+'/Aster')
from Cata importation cata
for order in cata.JdC.com mandes:
print commande.nom
```

It can be written:

```
importation sys
TOP='/local/yessayan/Eficas/EficasPourSalome'
sys.path.append (SIGNAL)
sys.path.append (TOP+'/Aster')
package=__import__ ("Cata.cata")
module_cata= getattr (package, "cata")
for order in module_cata.JdC.com mandes:
print commande.nom
```

As it is indicated in the documentation of python (http://www.python.org/doc/current/lib/built-in-funcs.html), the instruction

__import__ ("Cata.cata")

import the Cata package.

It thus remains to recover the module of the catalogue starting from the package for example: module=getattr (package, "cata"))

The order __import__ makes it possible to import a module or a package whose name is not known that at the moment of the interpretation of current script.

3.4
Modulate Python written in language C

4
The catalogue general of the orders of Code_Aster

4.1
Example 1: a factory to build the simple key words

4.1.1 Principle of the factory

The end-user of Code_Aster provides values to the functionalities via key words.
A key word makes it possible to introduce a value (key word simple) or another key word (key word factor).

The mechanism of loading and especially of use of the value of the key word is complicated by the fact
that the type of this value is not single. This type can be for example: float, int, string,… In the following example:

MASSE_VOLUMIQUE=7800.0

FICHIER_MAILLAGE=' maillage.unv'

Key words simple MASSE_VOLUMIQUE and FICHIER_MAILLAGE receive the value respectively 7800.0 and "maillage.unv"

It is thus necessary to describe some share the characteristics of the key word simple (name, type, default value,
unit, optional or obligatory statute,…). The answer to this question is founded on separation characteristics/following value:

· the principal role of an object of the type MCSIMP is to wrap the value of a simple key word otherwise-known as the principal attribute of an object of the type MCSIMP is a value;
· an object of the type SIMP has two principal roles:
- to wrap the definition of a simple key word: an object SIMP contains the whole of characteristics of a simple key word: its name, the type of its value,…;
- but this object also has a function __call__ which makes it possible to generate a key word simple starting from its characteristics; an object SIMP can be regarded as one machine to manufacture an object of MCSIMP from where the factory term

Each object MCSIMP contains its value and a reference on the object SIMP which describes its characteristics.

The technique of the factory is presented Ci below by reducing classes MCSIMP and SIMP to their minimum.

· Class MCSIMP has two attributes:
- definition: its definition (a reference on object SIMP which created object MCSIMP) to leave it is possible to recover the text part of the key word: definition.nom or sound definition.type type;
- valley: its value
· class SIMP has two attributes:
- name: the part text (character string which can contain a white space) key word MCSIMP which object SIMP will build;
- type: the type of the value introduced by the key word MCSIMP which object SIMP will build.

The function __call__ of class SIMP creates an object of the type MCSIMP. That implies that the class
MCSIMP is defined before the definition of this function.

In the model of design factory applied to the command set of Code_Aster, the class manufactured (example MCSIMP) must be defined before the class fabricante (SIMP).
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Code_Aster ®

Version
6.4

Titrate:
Code_Aster: the maintenance of Eficas and the supervisor

Date:
02/04/04
Author (S):
C. CAREMOLI, A. YESSAYAN Key
:
D9.02.01-A Page
: 19/34

```
# script main.py
importation sys
# string in python 2.1 becomes str starting from python 2.2
s_version=str (sys.version_info [0]) + '. '+str (sys.version_info [1])
assert (float (s_version) <=2.1), 'the version "+s_version+" of python is
INVALIDE'

class MCSIMP:
def __init__ (coil, definition, valley, parent=None):
self.definition = definition
self.val = valley
self.parent = None

class SIMP:
def __init__ (coil, name, type):
self.nom = name
self.type =type
def __call__ (coil, valley, parent=None):
assert (str (standard (valley))== " <type '"+self.type+" '>")
return MCSIMP (definition=self, val=val, parent=parent)

d_context= {
"MASSE_VOLUMIQUE": SIMP (nom=' MASS VOLUMIQUE', type=' float'),
"FICHIER_MAILLAGE": SIMP (nom=' FILE MAILLAGE', type=' string')
}

s_commande = "rho=MASSE_VOLUMIQUE (7800.0)"
exec s_commande in d_context, locals () # rho is added to the space of
names locals ()
```

print rho
print rho.definition.nom
print rho.val

s_commande = "mail=FICHIER_MAILLAGE ("maillage.d")"
exec s_commande in d_context, locals () # email is added to the space of names locals ()
print email
print mail.definition.nom
print mail.val

sys.stderr.write ("FINE NORMAL of main.py " + ' \)

The interpretation of script main.py above gives following posting:

<__main__.MCSIMP authority At 0x810c4ac>
DENSITY
7800.0
<__main__.MCSIMP authority At 0x810c4d4>
FILE GRID
maillage.d

Let us consider the lines

s_commande = "rho=MASSE_VOLUMIQUE (7800.0)"
exec s_commande in d_context, locals ()

The mechanism of construction of the rho object, in the local space of names, is a mechanism of factory including/understanding the following stages:

· In the space of names d_context, the order becomes:
rho=SIMP (nom=' MASS VOLUMIQUE', type=' float') (7800.0)
then it is carried out;
· An object of the type SIMP is built with the name "DENSITY";
· the method __call__ is called with val=7800.0 in argument;
· starting from the name (self.nom) and of the value (valley), the method __call__ creates and turns over
an object of the type MCSIMP;
· the turned over object is affected with the variable rho in the local space of names.

Important:

In the order s_commande, the key words should not contain of white space:
"rho=MASSE_VOLUMIQUE (7800.0) "is a valid Python instruction.

A white space in "rho=MASSE VOLUMINAL (7800.0)" would generate an error with interpretation.

In the dictionary of the key words the character strings used for the keys, must obey writing rules of an identifier Python: no white space.

4.1.2 The organization of the factory in files

The organization in files, presented below, described that which is used for the graphic interface Eficas and for the supervisor. It also describes the procedure of loading of the command set in the memory starting from the information provided in the command file - by the user final.

The files all of scripts Python - must be defined in the following order:

1) MCSIMP. py
2) SIMP.py
3) dictio.py: the dictionary of key words MASSE_VOLUMIQUE and FICHIER_MAILLAGE
4) valeurs.py: the file of the values provided by the end-user (for Aster, the file of orders user)
5) main.py: the code charging the values provided by the end-user while reading and interpreting the file valeurs.py (for Aster the éxecutable, also interpreter python: aster.exe)

Let us recall that the definition of class MCSIMP must be carried out before that of class SIMP. It who leads to an organization starting with the definition of class MCSIMP.

 # MCSIMP.py script

class MCSIMP:
def __init__ (coil, definition, valley, parent=None):

assert (definition. __class__. __name__==' SIMP')
self.definition = definition

assert (standard (valley). __name__==definition.type)
self.val = valley
self.parent =None

return
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Code_Aster ®
Version
6.4

Titrate:
Code_Aster: the maintenance of Eficas and the supervisor

Date:
02/04/04
Author (S):
C. CAREMOLI, A. YESSAYAN Key
:
D9.02.01-A Page
: 21/34

The attribute self.parent useless here, will be used when the key word is defined inside one order. It will then contain a reference on this order.

The developer of Accas can then define class SIMP.

 # SIMP.py script

importation MCSIMP
importation standards

class SIMP:
def __init__ (coil, name, type):

```
self.nom = name
self.type =type
def __call__ (coil, valley, parent=None):
assert (str (standard (valley))== " <type '“+self.type+” '>”)
return MCSIMP.MCSIMP (definition=self, val=val, parent=parent)
```

Once two modules SIMP and MCSIMP placed at its disposal, the developer of functionalities numerical of the code can now define a “applicatif catalogue of key words 'in the file dictio.py. This script defines in a dictionary python, description (standard, possible values, field of definition,…) values associated with the key word with kind to allow the reading of these values.

For example:

```
# script dictio.py

from SIMP importation SIMP
dict= {“MASSE_VOLUMIQUE”: SIMP (nom=' MASS VOLUMIQUE', type=' float',
“FICHIER_MAILLAGE”: SIMP (nom= `FILE MAILLAGE', type=' string')}
```

And the end-user can finally use the functionalities by providing a script, for example valeurs.py.

```
# script valeurs.py

rho=MASSE_VOLUMIQUE (7800.0)
mail=FICHIER_MAILLAGE (“maillage.d')
```

Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

To charge in memory the data provided by the user, the code will read the command file as follows:

```
# script main.py

from SIMP importation *


d_context= {"MASSE_VOLUMIQUE": SIMP (nom=' MASS VOLUMIQUE', type=' float'),
"FICHIER_MAILLAGE": SIMP (nom=' FILE MAILLAGE', type=' string')}


nom_script_valeurs = "valeurs.py"
f=open (nom_script_valeurs, "R")
string_parametres = f.read ()
f.close ()

exec string_parametres in d_context, locals ()

print rho, rho.definition.nom, rho.val
```

To charge in memory, the value associated with a simple key word, script python should be interpreted
text of the key words simple and value (S) associated (S) provided by the end-user - in space with names (it
dictionary d_context) of the key word.

4.2
Example 2: a factory to build an order

In practice, the key words are not separately but obligatorily defined inside one order. What complicates the process of construction of the key words in the memory. Us now let us present an example always simplified intended to facilitate the comprehension of it process.

For that we will consider that a command set is a list of orders of the type "procedure" and that each order is parameterized by simple key words and only by simple key words (not of conditional block, not of put-key factor). This simplification increases legibility while preserving all the categories of difficulties of facing to charge the play of order in memory.

One thus starts by adding a factory orders of the type PROC_ETAPE.

4.2.1 Addition of classes PROC and PROC_ETAPE with the factory

The class PROC_ETAPE which models an ordering of type "procedure", is very close to classify MCSIMP. So much so that both could inherit a class common mother (it is moreover the case in Accas).

 # PROC_ETAPE.py script

print "\ tImport of "+__name__

class PROC_ETAPE:

```
def __init__ (coil, definition, args= {}):
print 2*' \ t'+ " PROC_ETAPE __init__: creation of an object “+ \
coil. __class__. __name__
print 3*' \ t'+' PROC_ETAPE __init__: definition.nom=', definition.nom
print 3*' \ t'+' PROC_ETAPE __init__: args=', args

assert (definition. __class__. __name__==' PROC')
self.definition = definition
self.valor = args

return
```

An object PROC_ETAPE is manufactured by an object of the type PROC. Its attribute self.definition is
a reference on the object PROC which created it.

```
# PROC.py script
print “\ tImport of “+__name__
importation PROC_ETAPE
from SIMP importation *

class PROC:
def __init__ (coil, name, COp, ** args):
print 1*' \ t'+ " PROC __init__: creation of an object “+ \
coil. __class__. __name__
print 2*' \ t'+' PROC __init__: nom=', name
print 2*' \ t'+' PROC __init__: args=', args
self.nom = name # text of the order
self.entites = args # dictionary of the manufacturers
self.op = COp # number of operator FORTRAN
return

def __call__ (coil, ** args):
#
args
contains

definition

values

MCSIMP
(MASSE_VOLUMIQUE,
# FICHIER_MAILLAGE)
print 1*' \ t'+' PROC __call__: args=', args
```

print 1' \ t'+' PROC __call__: self.entites=', self.entites*
 # construction of the simple key words of the order and addition in
 # the dictionary of the key words of order PROC in the course of
 # construction
dict = {}
for K, v in args.items ():
dict [K] = self.entites [K] (val=v, parent=self)
print 1' \ t'+' PROC __call__: dict=', dict*
return PROC_ETAPE.PROC_ETAPE (coil, dict)
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

During its creation carried out starting from the catalogue, object PROC memorizes in the dictionary self.entites, the composition of the order in simple key words; this information will be used in the second time to build the simple key words of the order when object PROC will be called upon via its method __call__.

It is also the method __call__ which will initialize the key words located inside the order with the values provided in the command file (module SIMP is exactly that presented in the first example).

4.2.2 Concept of catalogue of orders

In the first example ([§4.1]), the description of the key words was made in the dictionary (space names) d_context. But to facilitate the task of the developers of Code_Aster, it is preferable to describe the orders and their contents via a script Python then to convert it script in a dictionary which will be used as space of names for the loading of the orders. For our second example, the catalogue can be written thus;

 # script cata.py

print 1' \ t'+ " Importation of "+__name__*

*from SIMP importation **
*from PROC importation **

AFFE_MATERIAU=PROC (nom=' AFFE_MATERIAU',
op=10,
MASSE_VOLUMIQUE=SIMP (nom= " DENSITY ", type=' float'),
FICHIER_MAILLAGE=SIMP (nom= " FILE GRID ", type=' string'))

This catalogue contains only one order: AFFE_MATERIAU of which the use will start the call to routine FORTRAN op0010. This routine will use two parameters MASSE_VOLUMIQUE and FICHIER_MAILLAGE

The conversion of the catalogue into a dictionary is done into important simply the catalogue in one space names. What does the following sequence:

d_context= {}
*string_cata= " from cata importation * "*
exec string_cata in d_context

4.2.3 Concept of command file

The command file is him also a script Python, very simple intended to be interpreted in the space of names of the catalogue of orders. For example:

 # script commandes.py

AFFE_MATERIAU (MASSE_VOLUMIQUE=7800.0, FICHIER_MAILLAGE= " maillage.unv ")

Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Code_Aster ®
Version
6.4

Titrate:
Code_Aster: the maintenance of Eficas and the supervisor

Date:
02/04/04

In script commandes.py above, the end-user asks for the execution of routine FORTRAN op0010 with MASSE_VOLUMIQUE=7800.0 and FICHIER_MAILLAGE= " maillage.unv ". He is interpreted by the following sequence:

```
f_commandes=open ("commandes.py", "R")
string_commandes = f_commandes.read ()
f_commandes.close ()
exec string_commandes in d_context
```

At the end of which the command set (here reduced to only order AFFE_MATERIAU) is defined in the space of names d_context.

4.2.4 Use

catalogue
and of the command file

Following Python script carries out the loading of the catalogue, the loading of the command file and the examination of the command set in the memory.

```
# script main.py
importation traceback
trace=traceback.extract_stack ()
script_file=trace [0] [0]
prefixe=script_file+': '
print prefixe+ " BEGINNING of ", script_file

d_context= {}
# 1. Loading of the catalogue
# Creation - into important the catalogue cata - of a space of being useful name
# for the interpretation of the command set

print 3*' \ n'+prefixe+ " importation of the catalogue "
string_cata= " from cata importation * "
exec string_cata in d_context

# 2. Loading of the text of the orders
# Lecture of the file of commmandes (the text of the orders east stores in
```

one
 # chains of caratères)

print 3' \ n'+prefixe+ " reading of the text of the orders "*
f_commandes=open (“commandes.py”, “R”)
string_commandes = f_commandes.read ()
f_commandes.close ()

 # 3. Creation of the command set
 # Interpretation of the text of the order in the d_contexte of the catalogue.
structure
 # command set, produced, is stored in the space of name d_context
print 3' \ n'+prefixe+ *
“Conversion of the text of the orders (string) into a command set
(d_context) “

exec string_commandes in d_context

 # 4. Course of the structure command set in the d_contexte

print 3' \ n'+prefixe+ " Posting of the command set "*
importation standards
for K, v in d_context.items ():
yew
type (v) ==types.InstanceType
and
v. __class__. __name__==
“PROC_ETAPE”:
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

: 26/34

 # if the attribute is an order, one examines his value
 # i.e. its key words
print 1*' \ t'+v.definition.nom+' \ t'+str (v. __class__)
for kk, vv in v.valeur.items ():
print 2*' \ t'+kk, ": ", vv, "\ you, vv.val

***print 2*' ***
print prefixe+ " FINE NORMAL of ", script_file

4.3
The file catalogues

4.3.1 Catalogue of an order

The catalogue of an order contains the description of the order. Each order is one authority of class OPER, PROC or MACRO

Attribute Description
name
name of the order (character string without white space)
COp
number of operator FORTRAN: entirety ranging between 1 and 199
sd_prod
type of the result, for the orders of the type OPER
regulate
list rules of composition of the order
Fr
French comment
Doc.
reference of documentation Aster
reentrant

repetable

entities
Composition of the order: arguments containing the description of the key words used to provide values to the key words of the order

Example

ASSE_MAILLAGE=OPER (nom=' ASSE_MAILLAGE', op= 105, sd_prod=maillage,
fr=' To assemble two grids under only one nom',
docu=' U4.23.03-e', reentrant=',
GRID =SIMP (statut=' o', typ=maillage, min=2, max=2),
) ;

In this example, drawn from the true catalogue of orders of Code_Aster:

· the described order is called ASSE_MAILLAGE;
· it makes it possible to start operator FORTRAN op0105;
· it turns over a data of the grid type; this type is defined in the beginning of catalogue cata.py;
· the use of operator FORTRAN op0105 requires obligatorily, the supply of 2 data of grid type
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Code_Aster ®
Version
6.4

Titrate:
Code_Aster: the maintenance of Eficas and the supervisor

Date:
02/04/04
Author (S):
C. CAREMOLI*, A. YESSAYAN Key*
:
D9.02.01-A Page
: 27/34

4.3.2 General structure of the file catalogues

The file catalogues orders of Code_Aster the module $TOP/Aster/Cata/cata.py - contains following information:
· importation of all information of the module Accas, in particular Accas.A_ASSD.ASSD
· declaration of the types deriving from the generic type Accas.A_ASSD.ASSD, used to typify them values of the key words or the values turned over by the orders; for example types:
- entirety, reality, complex, list, chains;
- the geometrical ones, No (node), groupno, my (mesh), groupma;

- grid, model, MATER
- etc
· the list of the catalogues of the orders i.e. the description of all the orders,
with for each order

4.4
Installation of the catalogue in the memory

It is important to remember that a reference on the catalogue running is stored in the module whose reference is stored in CONTEXT. Reference CONTEXT is itself defined in the total space of names __builtin__

A reference on the catalogue running is obtained by:

CONTEXT.get_current_cata ()

4.4.1 The structure of data in the package Core

The following classes have as a role to store the command set in the memory and to restore value of the key words to the request.
OBJECT
get_val (): type_de_base
get_etape (): STAGE
MCSIMP
JDC
get_val (): type_de_base

get_etape (): STAGE
MCCOMPO
get_mocle (key): OBJECT
MCFACT
MCBLOC
get_val (): OBJECT
get_val (): OBJECT
STAGE
get_etape (): STAGE
PROC_ETAPE
MACRO_ETAPE

FORM_ETAPE

The whole of the classes presented below take part in the loading of the command set in memory. They inherit all the abstract class ENTITY. And their principal function is method __call__ which ensures the role of factory (factory) of object.

ENTITY

__call__ (valley, name, relative): OBJECT

SIMP
PROC

__call__ (valley, name, relative): MCSIMP
__call__ (** args): PROC_ETAPE

FACT
OPER

__call__ (valley, name, relative): MCFACT
__call__ (reuse, ** args): STAGE

BLOCK
JDC_CATA

__call__ (procedure, cata, cata_ord_dico,
__call__ (valley, name, relative): MCBLOC

name, relative, ** args): JDC

MACRO

__call__ (reuse, ** args): MACRO_ETAPE

FORM

__call__ (reuse, ** args): FORM_ETAPE

4.4.2 The Accas package

The Accas package is the principal package used by the supervisor of Code_Aster and the interface Eficas graph. It contains the modules corresponding to the classes obtained by assembly (use of the multiple heritage) of the classes of the other packages.

Class MCSIMP

N_OBJECT.OBJECT
V_OBJECT.OBJECT
I_OBJECT.OBJECT

N_MCSIMP.MCSIMP

V_MCSIMP.MCSIMP
I_MCSIMP.MCSIMP
__init__ (valley, definition, name, relative)
get_val (): type_de_base

isvalid (): Boolean
GetText (): string

A_MCSIMP.MCSIMP

__init__ (valley, definition, name, relative)

The objects of class MCSIMP are built with the manufacturer of class A_MCSIMP.MCSIMP.
The diagram above represents the principal contribution of each package to the functionalities of class MCSIMP
The method get_val () is used by the supervisor to recover the value of the simple key word and to turn over to the operator of Code_Aster which requires it.
Data-processing handbook of Description
D9.02 booklet: - HT-66/04/003/A

Code_Aster ®
Version
6.4

Titrate:
Code_Aster: the maintenance of Eficas and the supervisor

Date:
02/04/04
Author (S):
C. CAREMOLI, A. YESSAYAN Key
:
D9.02.01-A Page
: 31/34

Class MCCOMPO

The class MCCOMPO which models a composite OBJECT - does not exist in the Accas package but it is important because it is used as a basis for classes MCFACT, MCBLOC.

N_OBJECT.OBJECT
V_OBJECT.OBJECT
I_OBJECT.OBJECT

N_MCCOMPO.MCCOMPO
V_MCCOMPO.MCCOMPO
I_MCCOMPO.MCSIMP

build_mc (): type_de_base
carryforward (): string
getlabeltext (): string

The diagram above highlights behaviors justifying the existence of class MCCOMPO.
· the method build_mc () in the package Core: it builds the objects located inside
OBJECT;
· the method carryforward () which turns over the report/ratio of validation by applying the method isvalid () to
all OBJECT located in the MCCOMPO.
Data-processing handbook of Description

D9.02 booklet: - HT-66/04/003/A

5 *The macro-orders Python*

Summary description of the nature of the macro-orders:

· A Python macro-order can produce one or more results (called concepts) then that the simple orders produce zero (order of the type PROC) or a result (ordering of type OPER);
· An macro-order has parameters like an ordinary order; it is key words factors and of the simple key words;
· The principal concept produced by macro is turned over by the macro one while the concepts secondary products are arguments modified by the macro one;
· The secondary produced concepts must be typified: that is done via one function provided by the developer of macro via the argument sd_prod of the macro one;
· The body of the macro-order is fascinating a Python function charges of it the treatment which the call to other orders includes (or even with other macro-orders).

To define an macro-order, its developer must thus define:

· key words of the order;
· the type of the produced concepts;
· the body of the macro-order.

6 Some questions

6.1

How to know which file catalogue is used by a calculation?

The situation is as follows:

· a calculation was carried out with Code_Aster;
· several files of catalogue of the orders are present in the environment;
· the user or the developer wants to know that which is actually used.

A solution can be as follows:

· to import the catalogue in the command file, for example in ahlv100a.com m;
· to insert in the command file the following sequence which:
- the catalogue imports,
- writing the reference of the catalogue on the standard exit,
- the treatment stops.

importation cata, sys
print “ahlv100a.comm: cata=”, cata
sys.exit (0)

With this sequence, one obtains a result of the following style:

cata=<module “cata” from '/home/salome/yessayan/Devel/Asterv7/bibpyt/Cata/cata.pyc'>

From where it is deduced that the catalogue used can be in the file:

/home/salome/yessayan/Devel/Asterv7/bibpyt/Cata/cata.py

6.2
How mode DEBUG is managed?

*In Eficas and the supervisor - in fact, in any script Python - mode DEBUG is managed via one standard variable defined in the total space of names __builtin__: __debug__. In mode normal of interpretation (python main.py), __debug__ is put at **1** (in main.pyc) but in mode optimized (python O main.py) __debug__ is put at **0** (in main.pyo)*

At any moment, in all the modules, the variable __debug__ can be used to condition it treatment.

6.3
Where the catalogue of order is it charged in the memory

That it is in the supervisor or the graphic interface Eficas, JdC, the Python object containing it catalogue is created in module cata Cata package. More exactly, JdC is created at the time where the module cata is imported: the importation is carried out in the method imports class SUPERV modulate Execution/E_SUPERV. After the importation, the JdC object contains in its attribute orders, of list type - the definition of all the orders available like that of all the key words associated each order.

6.3.1 Creation of the JdC object

At the beginning of script cata.py, JdC is declared by the instruction:

JdC = JDC_CATA (code=' ASTER',
execmodul=None,
rules = (AU_MOINS_UN (“BEGINNING”, “CONTINUATION”),
AU_MOINS_UN (“FINE”),
A_CLASSER ((“BEGINNING”, “CONTINUATION”), “END”)))

This instruction calls mainly upon the method __init__ class N_JDC_CATA.JDC_CATA (package Core). In this method, the JdC object created is recorded in total space __builtins__, via the variable _cata in module CONTEXT: __builtins__ [“CONTEXT”]. _cata.

A reference on the catalogue running is always available in L `total space of names __builtins__.

6.3.2 Loading of the entities of the catalogue in the JdC object

After creation the loading is always carried out at the time of the importation of the catalogue in method imports, by creating objects of the types

· OPER:
· PROC:
MACRO ·:

6.4
Where the command set is carried out by the supervisor?

The command set J (object of the Accas.A_JDC.JDC type) is carried out in the Execute method class SUPERV in module E_SUPERV of the package Execution.

Two cases are possible:

· In mode PAR_LOT=' OUI' (in script the attribute j.par_lot of the command set is positioned with “YES”), the treatment is carried out by the call

j.exec_compile ();

· In mode PAR_LOT=' NON' (in script the attribute j.par_lot of the command set is positioned with “NOT”), the treatment is carried out by the call

ier= self.ParLotMixte (J).

6.5
With what is used the key word _F used in the command file?

Into the command file, a key word factor is introduced by the character string _F. In fact this character string is a name of class which deals with creation in memory of

dictionary corresponding to the key word factor starting from a description using the equal sign "=" and
brackets rather than the two points ": " and the accodances which it would be necessary to use with one dictionary standard Python.

For example:
ELAS=_F ***(****E = 2.1E11, NAKED = 0.3, ALPHA = 1.E-5, RHO = 8000.****)***
is equivalent to:
ELAS= {E: 2.1E11, NAKED: 0.3, ALPHA: 1.E-5, RHO: 8000.}

This presentation is more adapted to the wishes of the end-users and the tradition of process control language of Code_Aster.

6.6 Where the interface getvxx of the command set find-such?

The methods getvxx belong to the class STAGE defined in module B_ETAPE of the package Build.

7 Bibliography

[1]
Introduction to Python, Lutz Mark & David Ascher, O' REILLY, Paris, 2001
[2]
Python, Essential Reference, David Mr. Beazley, New Riders, 2001
[3]
Python 2.1 Bible, Dave Brueck & Stephen To tan, 2001

Page:
1/50
Organization:
EDF/IMA/MMN
Data-processing handbook of Description
D9.02 booklet:
D9.02.02 document
Documentation of development and maintenance
of the manager of memory JEVEUX for Code_Aster

1 Introduction

Code_Aster was developed in FORTRAN 77, this language does not have dynamic management memory and does not allow a very strong structuring of the types. The manager of memory JEVEUX made it possible to mitigate part of these disadvantages by offering the following possibilities:

· ***dynamic management*** *of the zone memory allocated at the beginning of work,*
· ***management of the overflows report*** *on file, with filing of the results in end of work,*
· ***structuring of the data*** *of Code_Aster, with access by name to the handled objects and standardization of types FORTRAN used.*

This document is intended for the documentation and the maintenance of the routines of the manager of memory JEVEUX prefixed by JJ, I or JX, routines JX call upon functions in general not portables. Thereafter, we will indicate this whole of routines under the term "the software". One precise description of the operation and the internal organization of the software is detailed there. The reader will be able to refer to documentation [D6.02.01] Management Memory JEVEUX which describes the interface of routines "user" I…. .

2/50

2

Organization of the segmentation memory

JEVEUX manages dynamically a zone memory allocated at the beginning of work according to parameters of execution provided to the main program of Code_Aster by the interface. This allowance initial is carried out by routine JXALLM in routine JEDEBU.

*On CRAY, the allowance of the zone memory is realized dynamically using the routine system HPALLOC. This zone is seen in the software through the table of the whole type (INTEGER*8 ISZON) of length LISZON and an address of beginning JISZON. It is stored in the /IZONJE/ commun run. One will use in the continuation of the document the term "word" to indicate the unit of addressing. On CRAY the word has*

*for length 8 bytes and corresponds to length of type INTEGER*8.*

The zone is managed word by word in unit of the type INTEGER (unit of addressing), the segments of values associated with objects JEVEUX are chained ahead and behind. Each segment of values is framed by 8 words containing, in this order, following information:

1) the address measured compared to JISZON of the following segment;

*2) the value of a shift used to align the segments of the type length higher than the unit of addressing. On CRAY, this value is always null for the segments of values associated objects of the type INTEGER and REAL*8. It is worth 8 (bytes) sometimes for segments of the values associated with objects of the type COMPLEX*16: indeed it happens that it beginning of the segment of value (position 5) cannot coincide with a position in the table of reference ZK16, it is then necessary to move of a word (8 bytes). It is even more frequent with type CHARACTER when it is worth 16, 32 or 80!*

3) the whole identifier associated the simple objects or the collections;

4) the statute of the segment of values which can take value X or U;

5) the state of the segment of values which can take value X, A or D;

6) the whole identifier associated the objects of collection;

7) the code of the class associated with object JEVEUX;

8) the address measured compared to JISZON of the preceding segment.

The management of the zone memory with type INTEGER does not make it possible to be aligned correctly with

types of length higher than this unit of addressing. Although the order EQUIVALENCE present in the software allows to align the initial address of the various variables (tables) of reference ZI, ZR, ZC, ZL, ZK8, ZK16, ZK24, ZK32 and ZK80, the positioning of an associated segment of values with an object of the type ZK32 has little chance to be aligned with a "multiple" of 4 of table ISZON on CRAY, from where need for managing a décallage among the descriptors.

The values making it possible to code the statute and the state of the segment of values are obtained in order to it

that the representation does not correspond to any type used within the segment of values and this in order to

to detect possible crushings of these descriptors. This role deals with the routine JJLIRS which is called mainly at the time of the requests of setting in memory of the segment of values. The emission of an error message of the type "POSSIBLE CRUSHING UPSTREAM…" indicate that the identifier (3)

or
the statute (4) were crushed, the emission of an error message of the type "CRUSHING DOWNSTREAM POSSIBLE… "indicates that the state (5) or classifies it (7) were crushed.
On CRAY, the values used have the following octal representation:

0100000000000000000000
for X,
0300000000000000000000
for A,
0200000000000000000000
for U,
0400000000000000000000
for D.

4/50

At the time of the call to routine JEDEBU, one creates a segment of values “X X " occupying the whole of zone with 8 words framing it, 4 words on both sides are added to ensure the compatibility of chainings before and back. Value 0 is deposited in end of the managed zone. The memory is released at the end of the work at the time of the call to routine JXLIBM, controls of integrity are carried out with this occasion (on CRAY, exploitation of the codes return of utilities HPCHECK and HPDEALLC).

Use of the segments of values

The couple state/statute makes it possible to know the use of a segment of values in memory:

· XX indicates that the segment of values is free, this position is directly usable,

· UA indicates that the segment of values is used in reading (its image disc will not be brought up to date after release),

· UD indicates that the segment of values is used in writing (its image disc will be brought up to date after release),

· XD indicates that the segment of values was released, but that its contents will have to be discharged on disc,

· TESTSTEMXÀ indicates that the segment of values was released, this position is directly usable.

3

Management of the segmentation memory

A request of access in reading or writing on the segment of values associated with an object JEVEUX cause, if it does not appear there already, a loading in memory of the contents of the segment of values associated. The address memory of an object JEVEUX corresponds to its relative position in the table ISZON. Au préalable, it is necessary to carry out a research, using routine JJALLS in traversing the segmentation memory to find a position to insert the segment of values. One use for this purpose the chaining before coded among the descriptors preceding each segment by values. The algorithm used is held in the following way:

1) one examines whether a free zone is available at the end of the zone memory,

2) one considers then spaces made up by agglomerating the free zones (XX) and them removable segments of values (TESTSTEMXÀ),

3) finally one carries out research among spaces made up by agglomerating the free zones (XX), removable segments of values (TESTSTEMXÀ) and segments of values déchargeables (XD).

Research begins starting from the position chosen at the time of the last request thanks to the value stored in the /IXADJE/ commun run. This mechanism can involve accesses disc when zones associated with segments with values déchargeables are re-used. The new segment of values is allocated with a tolerance of 8 entireties which correspond to minimum space associated with one segment of values (1 entirety by descriptor). When the search for place memory fails, one cause a stop of the application in error <S> (stop by the supervisor with safeguard of concepts created).

A call to the function system LOC through routine JXLOCS makes it possible to obtain the relative address of beginning of the segment of values compared to table ISZON by using the value of the position of

reference of the beginning of the zone memory obtained in JXALLM and stored in the commun run /ILOCJE/. It is the use of the routine JJALTY which makes it possible to switch on table Z. and to obtain according to the type the address compared to the good reference.
The allowance of a segment of values associated with an object of the type of which the length is greater than
*the unit of addressing used (for example for type CHARACTER *24) does not allow*
automatically to align itself compared to table ISZON, it is sometimes necessary to shift
some words. The value of this shift is stored in the second descriptor preceding it
segment of values and the effective size of the segment of values is adjusted by taking account of sound associated type.

It then remains to bring up to date the descriptors associated with the segment with values, this operation is
realized by routine JJECRS.

Page:
6/50

Search for place available

The call to routine JEDISP makes it possible to know at the time of the call, the size of the zones memory available, it carries out research by traversing the whole of the segmentation memory and progressively deposit the size of the free zones or déchargeables in a table provided by argument of call.

Checking of the segmentation memory

Crushings report which affect the descriptors (state or statute) or the chaining front can to be detected by using routine JXVERI. This routine examines the descriptors one by one of segments of values in memory and, on CRAY, calls utility HPCHECK. A fatal error message is emitted during the detection of an anomaly, if not the routine remains dumb.

4

Management of the releases

The release is the most complex mechanism implemented in JEVEUX. One conceives easily that when that one manages a finished memory capacity, it arrives one moment when it is not possible any more of to find of place. It is then necessary to cause unloadings on disc or to recover the place of zones become useless. JEVEUX deals with these mechanisms, in condition of course, that it programmer indicated the objects concerned to him; it is necessary to take some care before releasing an object, several units of program which can use an address simultaneously memory. The strategy of release calls on the one hand upon an internal mechanism with the manager of memory that we will describe and in addition with rules of programming which are the subject of document [D2.06.99] "New strategy of release of objects JEVEUX".

The release of a segment of values materializes by positioning with value X of the state in place and place of the value U. There is not an other immediate effect, it is only at the time of a research later of storage position which one will treat indeed the contents of the segments of values.

The setting in memory of an object JEVEUX is accompanied by the assignment of an attribute system: mark. This attribute, of whole type, takes the value of a meter incremented with each call to routine JEMARQ and décrémenté with each call to routine JEDEMA. It is possible to obtain the value current mark by calling routine JEVEMA.

The current marks have a strictly positive value thus. Values -1, -2 and -3 are used to treat the following exceptions.

Value -1 is used to keep permanently (throughout the execution of an order Aster) certain objects which will be released by a specific call. This mark is used at the time of the call to routine JEVEUT.

Value -2 is used by JEVEUX to bring back in a temporary way certain objects which will be released at once the finished action (put in memory of the system objects of collection,…).

Value -3 is used to permanently keep (throughout the execution of Code_Aster) them objects used by the Supervisor.

Mark -3 can come to replace any existing mark, mark -1 can replace a mark (positive) existing. The system object containing the list of the addresses of the segments of values must then be modified. Mark -3 is used at the time of the call to routine JEVEUS.

One thus builds a hierarchy of the segments of values associated with the objects. Each call to routine JEDEMA will cause the release of the segments of values having the current mark.
In order to optimize the releases pulled by a call to JEDEMA, the setting in memory of each segment of values is accompanied by the storage of its position (its address memory) in an object system (segment of values of the whole type). Thus the whole of the segments of values associated with an identical mark is easily identifiable and their location requires only one simple sweeping of a vector of entireties. The loop on the segments of values is carried out in two times: one treats first of all overall collections, then simple objects and associated segments of values with the contiguous collections are released.
The actualization of the mark of an object is carried out at the time of the call to routine JJECRS, the need,
the object system KDESMA is redimensionné (this object is indicated here by the name of variable FORTRAN
used to store its address within the units of program).
It is the routine JJLIDE which carries out indeed the release of the segments of values. The first argument of this routine is the name of appealing, it conditions the type of operation to be carried out:
LIBE
standard mechanism of release with examination of the state, the statute and the mark,
TASS
mechanism of release with immediate writing used at the time of the retassage of the files or in mode debug JEVEUX,
LIBF
mechanism used at the end of the work during the closing of a base JEVEUX.
Concerning the simple objects, this release does not pose a particular problem: the routine JJLIDE checks that the mark associated with the segment with values is identical to the current mark stored in the /IADMJE/ commun run, it modifies the descriptors (state and statute) of the segment of values and assigns to 0 the associated mark, possibly it causes an unloading (JXECRO)

and the contents of the attributes modify address memory and address disc. Release of an object of dispersed collection follows the same process, the attributes being modified within the system objects of collection. The release of a collection is more delicate, the system objects having to be maintained accessible in memory as long as a segment of values associated with the one with the objects with collection is present in memory (used, déchargeable, and even removable).

The call to routine JJLIBP makes it possible to mark free the segment of values (XX) and to incorporate, if they exist, the free zones around this last. One uses for this purpose the chaining before and the chaining back to position on the descriptors of the close segments. This routine can affect it contents of the segment of values released to value UNDEF if this type of operation is asked (mode debug JEVEUX). The /UNDFJE/ commun run contains a variable initialized by JEDEBU with this value which is judicious to cause a brutal stop during any use in one arithmetic operation. The placement of this mechanism is carried out at the time of the call to the routine JEDBUG. It must be used with precaution, only during debugging, indeed, it modifies appreciably operating mode of the manager of memory, unloadings not being more differed, the number of inputs/outputs increases and the time spent to the research of positioning segments of values in memory increases.

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

---

5 Management of the files of direct access

The manager of memory JEVEUX manages report unloadings on disc, to release from place in memory during the execution and to file the results at the end of the work. One uses with this end of the files of direct access. On CRAY they are utilities OPENDR, WRITDR, READDR and CLOSDR which are called.

The address disc of objects JEVEUX is obtained by combination of the number of the recording of file of direct access used to deposit the values, and possibly the position within this recording.

The length of the recordings is fixed, its value is selected at the time of the opening of bases JEVEUX via routine JEINIF.

On CRAY, the number of recordings which belongs to the parameters, is given in Code_Aster according to the conditions of operating (currently, it is not possible to lay out of more than 4 Go of space file at the time of a passage batch CRAY). Each base is cut out in unit logic length 523.468.800 bytes (concept of “extend”), this value is affected through function ENVIMA LOFIEM and stored in the /FENVJE/ commun run. This cutting allows to profit as well as possible from the possibilities offered by utility DMF of migration of files. JEVEUX manages a total index that it then cuts out for each extend, the address disc is measured compared to the total index, then modulo the number of recordings, one obtains the number of the extend easily and the relative address. The various logical units are accessible by a local name which is composed starting from the first four characters in small letters of the name of the base associated and with number of extend.

The size of the recordings defines two classes of objects JEVEUX:

1) the small objects whose size is lower than the length of a recording, they are accumulated in a space of the size of a recording before transfer on disc,

the 2) large objects which require several recordings to store their contents.

At the time of a request of writing on disc, the contents of the large objects will be directly transferred on disc, whereas a plug of writing is used for the small objects in order to cumulate them and to reach approximately the length of a recording before their transfer. At the time of a request of reading, at least it is a recording which will be used, in the case of the small objects one uses a plug of reading. The plugs of reading and writing belong to the system objects associated with each base JEVEUX.

On CRAY, the closing of the files of direct access is essential to bring up to date the index of access, it is the routine JXFERM which call utility CLOSDR.

Description of the recordings

Each recording car-is described in order to easily be able to identify its contents. Like for the zone memory, the recordings are seen like a succession of words of the whole type (INTEGER*8 on CRAY). The first two words give total information on the size of stored objects:

· if Ident1=Ident2=0 the recording contains small objects, three whole words (descriptors) are placed in front of each segment of values, it contain respectively, when they exist, the identifier of collection (Idc1), the simple identifier of object or it

---

number of object of collection (Ids1) and the length of the segment of value, follows the segment of values, and one starts again for the following until Idcn=Idsn=0;

· if Ident1 or Ident2 is different from 0, the recording contains whole or part of the segment values associated with a large object.

At the time of the destruction of a large object the identifiers Ident1 and Ident2 are positioned with value opposite (assignment of the sign -). In the same way, at the time of the destruction of a small object, them identifiers Idci and Idsi are affected sign -.

Writing of the objects

It is the routine JXECRO which treats the writing of objects JEVEUX. It also ensures, when it is necessary, the opening of the logical units associated the partition in extend of the bases. The routine examine the various recordings to find a succession of recordings being able to accomodate it segment of values or the plug following the cases. Recordings corresponding to large objects destroyed can thus be recovered. The writing of a small object results in a displacement of

contents of the segment of values in the plug of writing by routine JXDEPS with actualization
descriptors. The plug is transferred on disc only if the segment of values is of one
size higher than open space remaining. The segment of values associated with large objects is transferred
on disc by routine JXECRB. JXECRB is a hat calling upon the utility WRITDR which
bring up to date the descriptors Ident1 and Ident2 as well as a meter associated with the recording. At the time
later unloadings, the plug of reading can be used to bring up to date the image disc; one
logic indicates this type of use then.

Reading of the objects

It is the routine JXLIRO which treats the reading of objects JEVEUX. The segments values associated with
small objects is reloaded in memory from one of the plugs of reading or writing. The plug of
reading can be possibly discharged on disc before charging a new recording.
segment of values associated with the large objects is directly read again using routine JXLIRB.

---

6 Access by names: associative addressing

Simple objects JEVEUX, objects and objects of collections, are accessible by name. Names
handled by the routines I…. comprise 24 characters, the names used in-house by JEVEUX
comprise 32 characters to treat the case of the collections. The access by name, if it facilitates legibility and
allows to structure the data, cannot be used directly in-house. One thus has recourse to one
algorithm of associative addressing which, using a function of coding, makes it possible to associate one
whole identifier with a name. This system of coding is used to manage the names of the objects
JEVEUX for each definite class (partner at each base), but also for the names of the objects
of collection.

One uses for this purpose is a couple of objects formed by a vector of entireties and a vector of chains of
characters for the management of different the classes, is an object of repertory kind of names having

heterogeneous characteristic to be of contents (storage of the character strings and the identifiers entireties) for the named collections. The use of these repertories requires functions of access particular. The dimensioning of these objects is carried out in order to contain the number necessary identifiers by minimizing their size and collisions on the level of the function of coding.
Our choice to dimension the pointer of entireties associated with the result with the function with addressing is
decree on the following condition:

nrep = nprem where nprem > 1.3 * nmax

nprem is a prime number and nmax the maximum number of identifiers to be stored.
The calculation of the size of the repertories is carried out by the function JJPREM in which is stored under
form WENT BACK a list to 56 prime numbers up to value 611957 what limits to this value capacity of the repertories of names (and that of the pointer of entireties to approximately 795.000).
On CRAY, the selected function of coding JXHCOD calls upon the function system STRMOV which allows
to transfer byte by byte a character string in a table from entireties and to function XOR

---

to cumulate the results in an entirety (INTEGER). The identifier is finally obtained by one congruence modulo the length of the repertory nrep.

Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
12/50

Insertion of a name

The insertion of a new name in the repertories associated with the various bases is carried out with the assistance
routine JJCREN. The whole of the repertories of the open bases is examined in order to ensure
the unicity of the name of object, routines JEVEUX not accepting like argument the class associated with the base.
If the repertory of names starts to be saturated, the function of coding, although selected to be
dispersive, can give an identical identifier for two distinct chains, there is then collision.
A new identifier must be given by taking account of the value obtained previously.

Scan for noun in a repertory

This algorithm requires a certain number of comparisons of character strings and thus can
to become expensive. A commun run, brought up to date by last research in the repertories of different
bases contains the identifier and the associated chain in order to reduce the cost of research
(/IATCJE/). It is the routine JJVERN which carries out a comparison with the contents of the commun run
before the call to JJCREN. The code return of routine JJCREN depends on the type of research in
repertory: with insertion in the repertory of names (ICRE=1) this code return is obligatorily not
no one (there is possibly stop in error), without insertion it can be worth 0, if not 1 corresponds to an object
simple and 2 with a collection. It is the presence of a nonwhite chain between positions 25 to 32
who indicates that one treats an object of collection.
The characters composing the names of the objects are limited to the alphanumerics
supplemented by the special characters:
' '
the white,
'.'
the point,
“_”
the underlined white,
'$'
the symbol dollar,
“&”
and commercial one.
The conformity of the character strings is checked after insertion in the repertories (at the time of
creation of the name only) by comparison character by character with the contents of the commun run
/JCHAJE/ initialized in routine JEDEBU.

Destruction of a name
The destruction of a name uses the same algorithm as insertion, the position in the repertory can be released because of possible collisions, one thus proceeds while making negative the identifier and while affecting to "?" the character string of the name to be destroyed. Thus it will be always possible of to recover this position later on.

Redimensioning of the repertories
The redimensioning of the repertories is ensured in an automatic way using routine JJAREP.
The size of the repertories of the bases is doubled at the time of the operation. This operation rebuilt entirely the new existing repertory by insertion of the names. The order of insertion being preserved, the system objects do not require particular treatment, other that a recopy in one larger receptacle (it follows from there a displacement in memory of the latter) and their actualization on disc.

Case of the repertories of collection
The repertories of collection are objects of nonhomogeneous contents: they store at the same time it result (of whole type) of the function of coding and the character strings composing the names. They car-are described and the routines using them contain the following instructions:
INTEGER ILOREP, IDENO, ILNOM, ILMAX, ILUTI, IDEHC

PARAMETER (ILOREP=1, IDENO=2, ILNOM=3, ILMAX=4, ILUTI=5, IDEHC=6)

The value positioned with the address:
Represent:
ILOREP
size necessary to the whole of the whole codes,
IDENO
the address from which the names are stored,
ILNOM
the length of the stored chains,
ILMAX
the maximum dimension of the repertory,
ILUTI
the number of actually stored names,
IDEHC
the address from which the whole codes are stored.
The information stored with address ILUTI is brought up to date and used in the internal functions of access
with the repertories, its value is only accessible in an external way, using utility JELIRA
with for name of attribute NOMUTI.
The objects of named collection, during their creation by routine JECROC, are inserted using
function JJCODN. Intermediate routine JJCROC allows, according to the value of its second
parameter, to insert a new name or to check its existence and to recover its order
of insertion.

7 The system objects and segments of values not referred

The manager of memory JEVEUX uses part of the memory to manage the associated attributes
with the objects and to treat certain functions. In order not to multiply the routines of access we have
chosen to use the same structures for objects JEVEUX and the memory used for their
management. This is why during the impression of the segmentation memory one sees appearing
segments of values associated with illicit names with the direction user and referring to
various classes open to a given moment, but also of the segments of values which are not
associated any name. The system objects associated with the TOTAL base carry all the prefix
according to: ________GLOBALE________ (the name of the base is in position 9 to 24), the suffix (in
position 25 to 32) makes it possible to distinguish the various objects. The names of the system objects
are

built in the same way for the other bases.
The system objects are created at the time of the first call to routine JEINIF. JEVEUX needing in permanence to reach the segments of values associated, a specific mark (- 2) their is affected. A particular treatment their is reserved during the closing of the bases.

List system objects used by JEVEUX:

Suffix of the name
Contents
Fortan type
Cut
system object
associated (on CRAY)

1
$$CARA
characteristics of the base
INTEGER*8
11
associated
2
$$IADD
addresses disc of the objects
INTEGER*8
2*NREMAX
3
$$GENR
kind of the objects (E, V, NR or X)
CHARACTER*1
NREMAX
4
$$TYPE
type of the objects (I, R, C, L, K)
CHARACTER*1
NREMAX
5
$$DOCU
documentary field
CHARACTER*4
NREMAX
6
$$ORIG
documentary field
CHARACTER*8
NREMAX
7

$$RNOM
list names of objects
CHARACTER*32
NREMAX
8
$$LTYP
types of the segments of values
INTEGER*8
NREMAX
9
$$LONG
length measured in the type
INTEGER*8
NREMAX
segments of values
10
$$LONO
measured effective length
INTEGER*8
NREMAX
in the type of the segments of
values

11
$$DATE
go back to first safeguard
INTEGER*8
NREMAX

12
$$LUTI
length used of the segments
INTEGER*8
NREMAX
values
13
$$HCOD
count of associative addressing
INTEGER*8
NRHCOD
14
$$USADI
description of the contents of
INTEGER*8
2*NBLMAX
recordings
15
$$ACCE
a number of access in
INTEGER*8
NBLMAX
read/write with
recordings
16
$$MARQ
marks associated with the objects
INTEGER*8
2*NREMAX
17
$$INDX
index of the file of direct access
INTEGER*8
2*NBLMAX
associated
18
$$TLEC
plug of reading
INTEGER*8
LONGBL
19
$$TECR
plug of writing

INTEGER*8
LONGBL
20
$$IADM
addresses memory of the objects
INTEGER*8
NREMAX
where
NREMAX
is the maximum number of names associated with a class,
NRHCOD
is obtained starting from NREMAX with function JJPREM,
NBLMAX
is the maximum number of recordings,
LONGBL
is the length of the recordings.
The dimension of the majority of the system objects is likely to be readjusted in the course of calculation according to the needs, only what milked with the size of the files of direct access and with the length of recordings remains fixed. The last 5 objects of the list above do not have an image disc.

Segments of values not referred present in memory

Two segments of values present in memory do not have names to identify them, us
let us indicate starting from the name of the variables which are used in the subroutines.

Contents
Type FORTRAN
Cut
associated

KPOSMA
ISZON (JISZON+KPOSMA+I) is the position in
INTEGER*8
LGD
segment of values associated to KDESMA with the addresses
associated the ième mark
KDESMA
addresses memory of the “marked” objects
INTEGER*8
LGP
Dimensions LGD and LGP are adjusted during the execution, their values initial are
respectively the sum lengths of the vectors $$RNOM of each class and value 50.

8 Them
collections

The collections of objects JEVEUX are structures which allow the pooling of the attributes
and possibly an access named to a group of objects. They can be associated single
segment of values (contiguous collection) or with as many segments of values of objects (collection

dispersed). They are built starting from objects simple JEVEUX, and thus appear under this form among the objects associated with a class. The main object of the collection is the object of kind X, it is a vector of 11 entireties containing the identifiers of the various objects composing collection (inter alia the system objects of the collection which contain a suffix starting with $$). This vector bears the name allotted using routine JECREC (CHARACTER*24). Objects system clean with the collection carry a suffix starting with $$ in position 25, if they are associated a divided object, they carry a suffix starting with &&. Attributes common to the whole of the objects of collection are deposited among the attributes of the system object $$DESO (kind,
type, length, etc).

The system objects associated with a collection are created in the associated class (identical attribute for the whole of the system objects) and charged in memory via routine JJCREC.

17/50

Suffix of the name of the object
Contents
Type FORTRAN
Type of collection
system of collection
associated

1
$$DESO
contiguous collection:
INTEGER*8
dispersed and
attributes associated with this
values associated with
contiguous the segment
object are the attributes
various objects of
values does not exist
commun runs of the collection
collection
that for
contiguous collections

2
$$IADD
addresses disc of
INTEGER*8
dispersed
objects of collections
dispersed

3
$$IADM
addresses memory of
INTEGER*8
dispersed
objects of collections
dispersed

4
$$MARQ
associated marks
INTEGER*8
dispersed
with the objects of
dispersed collections

5
$$NOM or divided object
list names of objects
according to
named
collections
the repertory
named
associated
6
$$LONG or divided object
measured length
INTEGER*8
of variable length
in the type of
segments of values;
receives the values
associated the attribus
LONMAX and NOMMAX
7
$$LONO or &&LONO
effective length
INTEGER*8
of variable length
measured in the type
segments of
values; is used in
intern by the software
8
$$LUTI or &&LUTI
length used of
INTEGER*8
of variable length
segments of values
9
$$NUM
information
INTEGER*8
numbered
concerning
collections
numbered
The routines using the collections, and more precisely the descriptor object of the collection contain

following instructions:

```
INTEGER IVNMAX, IDDESO, IDIADD, IDIADM,
+ IDMARQ, IDNOM, IDREEL, IDLONG,
+ IDLONO, IDLUTI, IDNUM
PARAMETER (IVNMAX = 0, IDDESO = 1, IDIADD = 2, IDIADM = 3,
+ IDMARQ = 4, IDNOM = 5, IDREEL = 6, IDLONG = 7,
+ IDLONO = 8, IDLUTI = 9, IDNUM = 10)
```

What makes it possible to position directly in the zone memory to obtain the identifiers of
system objects (when they exist) in the following order: $$DESO, $$IADD, $$IADM, $$MARQ, $
$NOM,
$$LONG, $$LONO, $$LUTI and $$NUM. IDREEL is not used any more. The maximum number of objects of
collection is stored with address IVNMAX.

Named collections

The objects associated with such a collection are accessible by their name (function JEXNOM) and by their
sequence number of insertion (function JEXNUM). It is possible to use routines JENUNO and JENONU
to pass from the number to the name and conversely. The length of the names of the objects is limited to 8

characters (CHARACTER*8) if the collection is based on an “internal” repertory of name, or can to be worth 8, 16 or 24 if the collection is based on an “external” repertory of name, i.e. created beforehand (repertory of names divided).

Numbered collections

The objects associated with such a collection are only accessible by their sequence number of insertion (function JEXNUM). The system object $$NUM is a vector of 2 entireties containing respectively the maximum number of objects of collection and the number of objects used.

Dispersed collections

Each object is associated a segment of values, it is thus not necessary to bring back the whole of the collection to reach a particular object. In this case it is necessary to manage 3 system objects: one for the addresses memory of the segments of values ($$IADM), the other for addresses disc ($$IADD) and the last to manage the releases ($$MARQ).

Contiguous collections

There is only one segment of values for the whole of the objects of the collection which is created and dimensioned once and for all at the time of the first setting in memory of one of the objects of collection. This segment of value is associated the system object $$DESO.

Collections variable length

Each object must be dimensioned: by affecting the attribute length by routine JEECRA or while providing of a vector length (divided object). In this case 3 system objects are necessary: for the lengths ($$LONG or divided object), for the lengths in the type of associated segments of values ($$LONO or &&LONO in the case of a divided object) and finally for lengths used ($$LUTI or &&LUTI in the case of a divided object).

In the case of the contiguous collections, it is possible to reach directly the vector lengths cumulated by using function JEXATR combined with the call to JEVEUO to obtain the address of this vector. This access makes it possible to be freed from a call to JEVEUO by object of collection. dimension of the system object $$LONO is incremented of 1 compared to the length of the object system $$LONG to this end.

Collections fixed length

Each object has same dimension, this attribute can be affected various ways: in directly affecting the attribute length of an object of the collection or attribute LONT of overall length for a contiguous collection (call to JEECRA).

The mechanism of access to the objects of collection
The requests of access to the objects of collection request the system objects attached to the collection, of same manner that it is necessary to have access to the system objects associated with a class at the time of
requests on the simple objects. It is thus necessary that these objects are present in memory as of
that a request is carried out on one of the objects of collection. Routine JJALLC is charged with
to put in memory the system objects of collection. They obey particular rules
concerning the releases because they can be discharged from the memory only when all the objects
from collection themselves were discharged (actualization of the addresses disc and memory).
management of the divided objects is even more delicate because it is necessary to be able to secure itself against a release
inopportune from the latter, for this purpose, they receive a particular mark which is worth -1.
The various requests on the objects of collection are carried out starting from the routines I used for
the simple objects, but require the use of the functions of synchronization JEXNOM, JEXNUM or
JEXATR. These functions of the type CHARACTER*32 update the contents of the respective commun runs

---

/IDATJE/, /INUMJE/ and /KNOMJE/, moreover they replace the character string associated with
name of object JEVEUX in position 25 to 32 by the respective suffixes $$XNOM, $$XNUM and $
$XATR.
The routines of low level then will seek this information within the various commun runs
according to the type of access.

---

9 Them
continuations

The system object of suffix $$CARA (containing the name of the base associated in position 9 to 24), contains
information necessary to the reopening of the file of direct access, it contains inter alia,
position of the segment of values associated with the addresses disc with the unit with the objects contained in
the base, as well as dimensions characteristic of the system objects. One thus takes the precaution of
to store at the head in the first recording. In the event of continuation on a basis, the first
action carried out will be the second reading of the contents of this object. Routine JXLIR1 opens the file
associated the first “extend” (glob.1) with characteristics which are clean for him (what can
to lead to a message of alarm), reads the first 14 values (3 for the descriptors of the segment of
values on disc and the 11 awaited values) then closes again the file. The length of the index being
known, it is then possible to reopen properly the files (the system objects can have been
deposited on the various files constituting the base). The system objects not having an image disc
are created and initialized (addresses memory, plug of input/output,…).

10
Treatment of objects JEVEUX

Creation of the objects

The creation of the descriptors (name and attribute of class, kind and type) of objects JEVEUX is
realized using routine JECREO for the simple objects, and by routine JECREC for
collections. The decoding of the chain passed in argument to affect the attributes is carried out by
routine JJANAL. In the case of the objects of kind E the attributes length are directly
affected.

The assignment of the attributes

The generic attributes are affected during the creation of the simple name of object or collection, they
appear in the following table:

· for the simple objects and the collections:

CLAS
classify fastening of
V: base Volatile, G: base Globale, L: base
the object at a data base.
Local, C: base Catalogues compiled

GENR
kind of the object
E: simple variable, V: vector, NR: repertory of
names
TYPE
type FORTRAN of the object
I, R, C, L, K8, K16, K24, K32, K80
LTYP
length of the type
managed automatically for types I, R, C,
L, standardized to 8, 16, 24, 32 and 80 for
characters
· for the collections only:
ACCESS
Type of access: NO if named, NAKED if
NO can be followed name of
numbered
repertory of names
STORAGE
CONTIG or DISPERSES
MODELONG
mode of definition length
VARIABLE can be followed name
objects of collection: CONSTANT
pointer length
or VARIABLE
LONT
overall length of a collection
contiguous
NMAXOC
numbers maximum objects of
collection
Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):

The other attributes are affected using routine JEECRA, these attributes appear in the table below:

· for the simple objects or the objects of collection:

LONMAX
length of the object of kind V
NOMMAX
length of the object of kind NR
LONUTI
length used of the object of kind V
NOMUTI
length used of the object of kind NR
DOCU
documentary field (4 characters)

Reading of the attributes

The values associated with the various attributes can be consulted constantly using routine JELIRA, including in-house managed attributes:

· accessible internal attributes:

DATE
go back to last disc unloading of the object
ORIG
not used
IADM
address memory
IADD
address disc
LONO
length measured in the type of the segments of values and according to their kind
USE
use (statute and state) of the segment of values in memory: UD, UA, XD, TESTSTEMXÀ or XX.

The statute and the state of a segment of values in memory can be collected by this routine in using USE for value of the argument of the name of attribute. Value XOUS for this same argument allows to determine if the object is a collection (X) or a simple object (S).

Note:

The consultation of the attributes of the objects of collection can require the setting in memory of attribute objects, and their release at the end of the action. A temporary mark equalizes to -2 is affected in this case.

Request of access to the objects

The whole of the requests of access to objects JEVEUX (objects simple, objects of collection or whole collections), which they are direct (JEVEUO, JEVEUS, JEVEUT) or indirect (JEEXIN, JENONU,…) follows the following process:
· treatment of the name of object passed in argument by JJVERN,
· possibly, put in memory or checking of the presence in memory of the objects system in the case of a collection by routine JJALLC, then using the routine JJCROC, determination of the identifier of object of named collection or checking of sequence number of numbered collection,
· according to the type, call to JJALTY to obtain the address compared to Z* table of commun run of reference,
· possibly put in memory then assignment of the identifiers of the segment of values and of the mark, and determination of the relative address by routine JXVEUO,
· in certain cases (for example consultations of attributes), release of the object and/or of collection by JJLIDE.
The attributes necessary to the description of object JEVEUX are read again or determined by the routine JXVEUO, the treatment are immediate for the simple objects bus one has access directly to the attributes

---

in the system objects associated with the base, some operations are necessary to treat them attributes of collection or object of collection (positioning in the system objects of collection).
Alternative JEVEUS allocates in a permanent way the segment of values in memory.

Destruction of the objects

The destruction of an object JEVEUX (simple object, collection or object of dispersed collection) requires
two interventions: destruction of the segment of values and destruction of the attributes. For an object simple, the segment of values can have an image disc, in this case it is also necessary to destroy this last, the corresponding recordings will be marked free and could be recovered
later. The segment of values in memory will be marked free. The image disc, if it exists will be marked free by using, following the type of object, the descriptors of the recording (system object $$USADI) or descriptors within a recording (assignment of the sign -). This function
is ensured by routine JXLIBD. Attributes (name, length, kind, etc) will be released (routine JJMZAT) and their position in the system objects of the associated base will be available for
new creation of descriptor. The system object containing the address of the marked objects must too to be reactualized. The treatment of an object of collection is identical, the actualization of the attributes is
realized on the system objects of the collection. The segment of values for an object of collection contiguous cannot of course be destroyed. The destruction of a collection is carried out while destroying the whole of the objects of collection and the system objects of the collection provided that they are not divided. Routines JEDETR and JEDETC make it possible to destroy objects JEVEUX, the first
work starting from an identifier, the second, more expensive, carries out a search first of all for descriptor in the repertories of the classes open starting from a character string to one
position given. Routine JEDETV is only used to destroy the objects on the basis
bird associated with the class V between the various orders with Code_Aster.

Release clarifies objects

Although the mechanism of release is implemented with the concept of mark and the calls obligatory with routines JEMARQ and JEDEMA, certain configurations require an explicit call
with the following routines of release:

JELIBE releases the object required by respecting the affected mark,
JELIBS releases the object of name passed in argument when the associated mark is worth -3,
JELIBZ releases the whole of the objects associated with a class with which the mark associated is worth -1.

Recopy objects

Utility JEDUPO makes it possible to duplicate an object JEVEUX (simple object, or collection supplements)
possibly by depositing the result on a different class. The new objects are released in
end of operation. If this action does not raise any difficulty for the objects simple, some care

are to be taken concerning the collections being pressed on external pointers. The latter can to be recreated to become system objects specific to the collection (one does not profit any more a setting jointly attributes concerned) or to be preserved just as they are, but it is then not allowed to deposit the result of the recopy on another class. The receptacle can preexist (the user provides a name or a chain), in this case it is destroyed at the beginning of operation. The recopy does not require not obligatorily the presence in memory of the segments of values to be copied, they can be read again directly on disc.

It is possible to make use of the utility JEDUPC which works starting from a under-chain of characters but requires on the other hand a preliminary research of the names in the repertory (what can prove to be expensive).

Impression of the contents of the segments of values

Utility JEIMPO is charged to print in a pleasant way the contents of (of) the segment (S) with value (S) associated (S) with objects JEVEUX. The objects system (associates with a class or one collection) are treated by routine JEPRAT. A setting in memory being able to be carried out, one particular mark (- 2) is assigned to the segments of values charged. According to the type of object (object simple, object of collection or collection) one recovers the attributes associated with (X) the segment (S) with values to call the routine JJIMPO which carries out the formatting of the data.

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0

Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
25/50

11 Treatment of the bases

Certain operations treat in their entirety bases JEVEUX, they are essential for to initiate the system of management of memory, but are also used at the end of the process. Attention, it contained bases is systematically enriched during the execution of the Aster order CONTINUATION, and it is essential to finish the execution by the order END to close properly files of direct access. Only a stop with a message UTMESS of the <S> type allows SUPERVISOR to validate the concepts created and to properly close the files of direct accesses by call to routine JEFINI

The opening of a base

The length of the recordings of the file of direct access and the initial length of the repertory of names remain the only adjustable parameters associated with bases JEVEUX. They are specified at the time of the call to routine JEINIF, one indicates also the statute of the base at the beginning of work for possibly to reopen an existing file, the statute at the end of the work makes it possible to avoid superfluous inputs/outputs if base is not preserved. The reopening of a base (order CONTINUATION in Code_Aster or reading of the catalogue of the compiled elements) requires the knowledge length of recordings of the file of direct access and the contents of certain system objects, the first recording contains the data essential to the reconstitution of this various information.
Routine JXLIR1 is charged to read again the beginning of the first recording: the file is opened access direct (with an index whose size is fixed at 11), one reads information in beginning of recording, then the file is closed again. One can then open the file of direct access with one table of suitable index length, and to read again the contents of the system objects stored on disc at the time preceding execution.

The closing of a base

The operation of closing of a base, carried out by routine JELIBF, consists in releasing the whole of objects which are attached there, with possibly writing on disc and to bring up to date the system objects. Two loops are necessary to release the objects: the first treats the collections, the second draft simple objects. The system objects are then discharged, the addresses disc are treated in the last, the plugs of input/output are emptied, finally one brings up to date the characteristics of the base on the first recording. The file of direct access is then closed by call to the routine

JXFERM.

The retassage of a base

At the time of the operations of destruction of object JEVEUX, the associated disk space is marked free but
is not systematically recovered. The retassage makes it possible "to fill" the vacuums in "going up" them recordings. It is thus necessary to modify the attribute addresses disc of the objects contained in recordings to be moved. This operation is immediate for the simple objects, concerning
collections it is necessary to have access to the system object containing the addresses disc (which itself can be
located the recording to be moved!). There is no reorganization within the recordings
containing the images of small objects. One uses routine JETASS and one calls upon the alternative "JETASS" of the routine of release JJLIDE. This utility can be directly called by
order FINE in Code_Aster.

The recopy of the bases

This operation must be carried out to take into account indeed the retassage, the files
direct access WRITDR which cannot be reduced in place on CRAY. Routine JXCOPY works to leave closed bases and restores them in the same state. This utility can be called by the order
END in Code_Aster.

---

12 Them
impressions

Routine JEIMPD makes it possible to print the list of objects JEVEUX present on one or more bases. The list is made up starting from the associated catalogue (system object $$RNOM) and they are printed following information for each object:

· the associated identifier in the name of object,
· the name of the object,
· the kind of the object,
· the type of the object,

· the length in the type,
· the length in byte of the segment of values,
· the number of the recording containing the segment of values,
· the position in the recording for the small objects,
· the number of access in reading on the recording,
· the number of access in writing on the recording.

1--
CONTAINED BASE G
NAME OF THE BASE: TOTAL
NB RECORDINGS MAXIMUM: 5242
RECORD LENGTH (BYTES): 819200

---- NUM ------------- NAME ---------------- G T - L - LOTY- - IADD- --LIADD- NB AC
1 ________GLOBALE ________$$CARA - VI 8 11 1 24 0
2 ________GLOBALE ________$$IADD - VI 8 4000 1.136 0
3 ________GLOBALE ________$$GENR - V-K- 1 2000 1 32160 0
4 ________GLOBALE ________$$TYPE - V-K- 1 2000 1 34184 0
5 ________GLOBALE ________$$DOCU - V-K- 4 2000 1 36208 0
6 ________GLOBALE ________$$ORIG - V-K- 8 2000 1 44232 0
7 ________GLOBALE ________$$RNOM - V-K- 32 2000 1 60256 0
8 ________GLOBALE ________$$LTYP - VI 8 2000 1 124280 0
9 ________GLOBALE ________$$LONG - VI 8 2000 1 140304 0
10 ________GLOBALE ________$$LONO - VI 8 2000 1 156328 0
11 ________GLOBALE ________$$DATE - VI 8 2000 1 172352 0
12 ________GLOBALE ________$$LUTI - VI 8 2000 1 188376 0
13 ________GLOBALE ________$$HCOD - NOR 8 4177 1 204400 0
14 ________GLOBALE ________$$USADI - VI 8 10484 1 237840 0
15 ________GLOBALE ________$$ACCE - VI 8 5242 1 321736 0

Impression of the segmentation memory

Routine JEIMPM makes it possible to print the list of objects JEVEUX present in memory and indicates them
free zones. Following information is printed:

· the class of the object,
· the associated identifier of collection in the name of object or 0,
· the simple identifier of object or associated object of collection in the name of object,
· the value (whole) of the mark associated with the segment with values,
· the address relative memory of the segment of values,
· the statute of the segment of values (X or U),
· the length measured in unit of addressing (whole on CRAY) of the segment of values,

· the state of the segment of values (X, A or D),
· the name of the object (possibly supplemented by the number of object of collection).

```
--------------------------------------------------------------------------------
---- SEGMENTATION MEMORY
----
--------------------------------------------------------------------------------
CL --NUM-- - MA --IADM-- - U - LON UA - - S ------------- NAME --------------
0 0 0 9 X 21622 X <<<< FREE >>>>
0 0 0 21639 U 50 D
G 0 1 -2 21697 U 11 D ________GLOBALE ________$$CARA
G 0 16 -2 21716 U 4000 D ________GLOBALE ________$$MARQ
G 0 17 -2 25724 U 9216 D ________GLOBALE ________$$INDX
G 0 15 -2 34948 U 5242 D ________GLOBALE ________$$ACCE
G 0 18 -2 40198 U 102400 D ________GLOBALE ________$$TLEC
G 0 19 -2 142606 U 102400 D ________GLOBALE ________$$TECR
G 0 20 -2 245014 U 2000 D ________GLOBALE ________$$IADM
G 0 2 -2 247022 U 4000 D ________GLOBALE ________$$IADD
G 0 3 -2 251030 U 251 D ________GLOBALE ________$$GENR
G 0 4 -2 251289 U 251 D ________GLOBALE ________$$TYPE
G 0 5 -2 251548 U 1001 D ________GLOBALE ________$$DOCU
G 0 6 -2 252557 U 2000 D ________GLOBALE ________$$ORIG
G 0 7 -2 254565 U 8004 D ________GLOBALE ________$$RNOM
G 0 8 -2 262577 U 2000 D ________GLOBALE ________$$LTYP
G 0 9 -2 264585 U 2000 D ________GLOBALE ________$$LONG
G 0 10 -2 266593 U 2000 D ________GLOBALE ________$$LONO
G 0 11 -2 268601 U 2000 D ________GLOBALE ________$$DATE
G 0 12 -2 270609 U 2000 D ________GLOBALE ________$$LUTI
G 0 13 -2 272617 U 4177 D ________GLOBALE ________$$HCOD
...
V 0 0 0 2177711 X 9680 X <<<< FREE >>>>
G 0.247 0 2187399 NAKED TESTSTEMX2DS .NUME.NEQU
```

G 0.248 0 2187409 X 6 D NAKED .NUME.REFE
G 0.249 0 2187423 X 1680 D NAKED .NUME.NUEQ
G 0.250 0 2189111 X 3360 D NAKED .NUME.DEEQ
G 0.251 0 2192479 X 1680 D NAKED .NUME.DELG
V 0 0 0 2194167 X.400 X <<<< FREE >>>>
G 0.252 0 2194575 X 6 D NAKED .SLCS.REFE
G 0.253 0 2194589 X 1680 D NAKED .SLCS.HCOL
G 0.254 0 2196277 X 1680 D NAKED .SLCS.ADIA
G 0.255 0 2197965 NAKED TESTSTEMX2DS .SLCS.ABLO
G 0.256 0 2197975 X 1680 D NAKED .SLCS.IABL
G 0.257 0 2199663 X 6 D NAKED .SLCS.DESC
0 0 0 2199677 X 946044 X <<<< FREE >>>>

Impression of the repertories system

Routine JEIMPR makes it possible to print the repertories associated with the various opened bases, it traverses the system object sequentially $$RNOM and prints the values of the attributes when it first character of the name is different from “?”. Following information is printed:

· the simple identifier of object,
· the name of the simple object,
· the kind of the object,
· the type of the object,
· the length of the type used,
· the length of the object (attribute LONMAX or NOMMAX),
· the length measured in the type of the segment of values,

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

· the number of the recording containing the image disc,
· the position in the recording for the small objects (in bytes),
· the address relative memory of the segment of values.

---- CATALOGUE CLASS G

NUM ------------- NAME ---------------- G T L --LENGTH--- - LOTY- - IADD- --KADM--
1 ________GLOBALE ________$$CARA - VI 8 11 11 1 21697
2 ________GLOBALE ________$$IADD - VI 8 4000 4000 1 247022
3 ________GLOBALE ________$$GENR - V-K- 1 2000 2000 1 251030
4 ________GLOBALE ________$$TYPE - V-K- 1 2000 2000 1 251289
5 ________GLOBALE ________$$DOCU - V-K- 4 2000 2000 1 251548
6 ________GLOBALE ________$$ORIG - V-K- 8 2000 2000 1 252557
7 ________GLOBALE ________$$RNOM - V-K-32 2000 2000 1 254565
8 ________GLOBALE ________$$LTYP - VI 8 2000 2000 1 262577
9 ________GLOBALE ________$$LONG - VI 8 2000 2000 1 264585
10 ________GLOBALE ________$$LONO - VI 8 2000 2000 1 266593
11 ________GLOBALE ________$$DATE - VI 8 2000 2000 1 268601
12 ________GLOBALE ________$$LUTI - VI 8 2000 2000 1 270609
13 ________GLOBALE ________$$HCOD - NOR 8 4177 4177 1 272617
14 ________GLOBALE ________$$USADI - VI 8 10484 10484 1 276802
15 ________GLOBALE ________$$ACCE - VI 8 5242 5242 1 34948
16 ________GLOBALE ________$$MARQ - VI 8 4000 4000 0 21716
17 ________GLOBALE ________$$INDX - VI 8 9216 9216 0 25724
18 ________GLOBALE ________$$TLEC - VI 8 102400 102400 0 40198
19 ________GLOBALE ________$$TECR - VI 8 102400 102400 0 142606
20 ________GLOBALE ________$$IADM - VI 8 2000 2000 0 245014
21 &&SYS FI.NOMS - V-K- 8.142.142 0 797130
22 &&SYS FI.ARITE - VI 8.142.142 0 797280
23 &&SYS FI.PRIORITE - VI 8.142.142 0 797430

24 &&SYS FI.CLASSE - VI 8.142.142 0 797580
25 &&SYS FI.VALEURS_IS - VI 8.142.142 0 797730
NUM ------------- NAME ---------------- G T L --LENGTH--- - LOTY- - IADD- --KADM--
26 &&SYS FI.VALEURS_R8 - V-R- 8.142.142 0 797880
27 &&SYS FI.VALEURS_C8 - V-C-16 142.142 0 798030
28 &&SYS RESULT.USER - V-K-80 500.500 0 812337
29 &&SYS RESULT.STAT - V-K-80 500.500 0 817355
30 &&SYS .CODE - V-K- 8 3 3 0 828466
31 &CATA.GD.LNOCMP - VI 8.107.107 0 1147885
32 &CATA.GD.NOMGD - N-K- 8.106.280 0 1148000
...
252 NAKED .SLCS.REFE - V-K-24 1 1 0 2194575
253 NAKED .SLCS.HCOL - VI 8 1680 1680 0 2194589
254 NAKED .SLCS.ADIA - VI 8 1680 1680 0 2196277
255 NAKED .SLCS.ABLO - VI 8 2 2 0 2197965
256 NAKED .SLCS.IABL - VI 8 1680 1680 0 2197975
257 NAKED .SLCS.DESC - VI 8 6.6 0 2199663

Impression of the attributes

Routine JEIMPA prints the whole of the attributes for an object JEVEUX. It is the whole of routines of impression of message (JVMESS, JVIMPI,…) who is used to format the different ones arguments.

<X> <JEIMPA> IMPRESSION OF THE ATTRIBUTES OF >&INEL.FACE4 .FFORMES <
> <
CLAS STANDARD >G< GENR >V< >R< LTYP 8 DOCU > < DATE 0 ORIG > <
LONMAX 152 LONUTI 0 LONO 152 IADM 1701910 IADD 0 LADD 0 USE >X D<

Note:

The impression of the attributes of the objects of collection or their contents can require the setting in

memory of the attribute objects, and their release at the end of the action. A temporary mark equalizes with
-2 is affected in this case.

Impression of the contents of a segment of values

Routine JEIMPO makes it possible to print the segments of values associated with an object JEVEUX.
SEGMENT IMPRESSION OF VALUES >&INEL.FACE4 .FFORMES <
>>>>>
1 - 1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
6 - 1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
11 - 1.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
16 - 1.00000E+00 -5.00000E-01 -5.00000E-01 5.00000E-01 0.00000E+00
21 - 0.00000E+00 0.00000E+00 0.00000E+00 5.00000E-01 -5.00000E-01
26 - 0.00000E+00 5.00000E-01 -5.00000E-01 0.00000E+00 5.00000E-01
31 - 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
36 - -5.00000E-01 5.00000E-01 5.00000E-01 -5.00000E-01 0.00000E+00
41 - 0.00000E+00 -5.00000E-01 0.00000E+00 0.00000E+00 5.00000E-01
46 - 0.00000E+00 -5.00000E-01 5.00000E-01 1.00000E+00 1.00000E+00
51 - 1.00000E+00 1.00000E+00 6.22008E-01 1.66667E-01 4.46582E-02
56 - 1.66667E-01 1.66667E-01 4.46582E-02 1.66667E-01 6.22008E-01
61 - 1.66667E-01 6.22008E-01 1.66667E-01 4.46582E-02 4.46582E-02
66 - 1.66667E-01 6.22008E-01 1.66667E-01 -3.94338E-01 -3.94338E-01
71 - 3.94338E-01 -1.05662E-01 1.05662E-01 1.05662E-01 -1.05662E-01
76 - 3.94338E-01 -1.05662E-01 -3.94338E-01 1.05662E-01 -1.05662E-01
81 - 3.94338E-01 1.05662E-01 -3.94338E-01 3.94338E-01 -3.94338E-01
86 - -1.05662E-01 3.94338E-01 -3.94338E-01 1.05662E-01 3.94338E-01
91 - -1.05662E-01 1.05662E-01 -1.05662E-01 -1.05662E-01 1.05662E-01
96 - -3.94338E-01 3.94338E-01 3.94338E-01 -3.94338E-01 1.05662E-01
101 - 1.00000E+00 1.00000E+00 1.00000E+00 1.00000E+00 6.22008E-01
106 - 1.66667E-01 4.46582E-02 1.66667E-01 1.66667E-01 4.46582E-02
111 - 1.66667E-01 6.22008E-01 1.66667E-01 6.22008E-01 1.66667E-01
116 - 4.46582E-02 4.46582E-02 1.66667E-01 6.22008E-01 1.66667E-01
121 - -3.94338E-01 -3.94338E-01 3.94338E-01 -1.05662E-01 1.05662E-01
126 - 1.05662E-01 -1.05662E-01 3.94338E-01 -1.05662E-01 -3.94338E-01
131 - 1.05662E-01 -1.05662E-01 3.94338E-01 1.05662E-01 -3.94338E-01
136 - 3.94338E-01 -3.94338E-01 -1.05662E-01 3.94338E-01 -3.94338E-01
141 - 1.05662E-01 3.94338E-01 -1.05662E-01 1.05662E-01 -1.05662E-01
146 - -1.05662E-01 1.05662E-01 -3.94338E-01 3.94338E-01 3.94338E-01
151 - -3.94338E-01 1.05662E-01

Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
30/50

13 APPENDIX 1: Description of the commun runs used in manager of memory JEVEUX

· /FENVJE/
INTEGER LFIC, MFIC
COMMON /FENVJE/ LFIC, MFIC
LFIC
length maximun in bytes of a extend,
MFIC
cut maximum in bytes of disk space usable.
· /IACCED//JIACCE/
PARAMETER (NR = 5)
COMMON /IACCED/ IACCE (1)
COMMON /JIACCE/ JIACCE (NR)
variable of reference and position of the segment of values associated with
the system object of suffix:
IACCE, JIACCE
$$ACCE
· /IADMJE/
INTEGER IPGC, KDESMA, LGD, LGDUTI, KPOSMA, LGP, LGPUTI
COMMON /IADMJE/ IPGC, KDESMA, LGD, LGDUTI, KPOSMA, LGP, LGPUTI
IPGC
Value of the current mark (varies between -3 and N),
KDESMA
address segments of values containing the addresses of the marked objects,
LGD
length of the segment of value associated to KDESMA,
LGDUTI
length used of the segment of value associated to KDESMA,
KPOSMA
address segments of values containing the positions associated with each
mark,
LGP

length of the segment of value associated to KPOSMA,
LGPUTI
length used of the segment of value associated to KPOSMA.
· /IATCJE/
INTEGER ICLAS, ICLAOS, ICLACO, IDATOS, IDATCO, IDATOC
COMMON /IATCJE/ ICLAS, ICLAOS, ICLACO, IDATOS, IDATCO, IDATOC
ICLAS
current class,
ICLAOS
classify simple object,
ICLACO
classify collection,
IDATOS
identifier of the simple object,
IDATCO
identifier of the collection,
IDATOC
identifier of the object of collection.
· /IATRJE//JIATJE/
PARAMETER (NR = 5)
INTEGER LTYP, LENGTH, DATE, IADD, IADM,
+ LONO, HCOD, CARA, LUTI, IMARQ
COMMON /IATRJE/ LTYP (1), LENGTH (1), DATE (1), IADD (1), IADM (1),
+ LONO (1), HCOD (1), CARA (1), LUTI (1), IMARQ (1)
COMMON /JIATJE/ JLTYP (NR), JLONG (NR), JDATE (NR), JIADD (NR), JIADM (NR),

---

+ JLONO (NR), JHCOD (NR), JCARA (NR), JLUTI (NR), JMARQ (NR)

variable of reference and position of the segment of values associated with
the system object of suffix:
LTYPE, JLTYP
$$LTYP
LENGTH, JLONG
$$LONG
DATE, JDATE
$$DATE
IADD, JIADM
$$IADD
IADM, JIADD
$$IADM
LONO, JLONO
$$LONO
HCOD, JHCOD
$$HCOD
CARA, JCARA
$$CARA
LUTI, JLUTI
$$LUTI
IMARQ, JMARQ
$$MARQ
· /ICODJE/
INTEGER NUMATR
COMMON /IDATJE/ NUMATR
NUMATR
identifier of the system object of collection $$LONO
· /IDATJE/

PARAMETER (NR = 5)
INTEGER NRHCOD, NREMAX, NREUTI
COMMON /ICODJE/ NRHCOD (NR), NREMAX (NR), NREUTI (NR)
NRHCOD
cut (in entirety) system object $$HCOD
NREMAX
cut (in entirety) system object $$RNOM
NREUTI
length used of the system object $$RNOM
· /IENVJE/
INTEGER LBIS, LAWS, LOLS, RENTED, LOR8, LOC8
COMMON /IENVJE/ LBIS, LAWS, LOLS, RENTED, LOR8, LOC8
LBIS
length out of bits of the entirety,
LAWS
length in bytes of the entirety,
LOLS
length in bytes of logic,
RENTED
length in bytes of the unit of addressing,
LOR8
length in bytes of reality,
LOC8
length in bytes of the complex.
· /IEXTJE/
PARAMETER (NR = 5)
INTEGER IDN, IEXT, NBENRG
COMMON /IEXTJE/ IDN (NR), IEXT (NR), NBENRG (NR)
IDN
is not used any more since the use of the named accesses.
IEXT
number open extends
NBENRG
a maximum number of recordings of a extend
· /IFICJE/
PARAMETER (NR = 5)
INTEGER NBLMAX, NBLUTI, LONGBL,
+ KITLEC, KITECR, KINDEF, KIADM,
+ IITLEC, IITECR, NITECR, KMARQ
COMMON /IFICJE/ NBLMAX (NR), NBLUTI (NR), LONGBL (NR),
+ KITLEC (NR), KITECR (NR), KINDEF (NR), KIADM (NR),
Data-processing handbook of Description
D9.02 booklet:

+ IITLEC (NR), IITECR (NR), NITECR (NR), KMARQ (NR)

For each base associated with index I ranging between 1 and NR

NBLMAX
numbers maximun recordings,

NBLUTI
a number of recordings used,

LONGBL
length in bytes of the recordings,

KITLEC
address in K1ZON of the segment of values associated with the plug with reading,

KITECR
address in K1ZON of the segment of values associated with the plug with writing,

KINDEF

address in ISZON of the segment of values associated with the index used for
files of direct access,
KIADM
address in ISZON of the segment of values associated with the addresses memory,
IITLEC
address disc of the plug of reading (number of recording),
IITECR
address disc of the plug of writing (number of recording),
NITECR
cut in bytes of the portion of the plug of writing used,
KMARQ
address in ISZON of the segment of values associated with the marks.
· /ILOCJE/
INTEGER ILOC
COMMON /ILOCJE/ ILOC
ILOC
pointer on the address of the beginning of the zone memory allocated by
JXALLM.
· /INUMJE/
INTEGER NUMEC
COMMON /INUMJE/ NUMEC
NUMEC
number of the object of collection or number of insertion in one
repertory of names.
· /ISTAJE/
INTEGER ISTAT
COMMON /ISTAJE/ ISTAT (4)
ISTAT
code associated with the state and the statute with the segments with values
ISTAT (1) corresponds to X
ISTAT (2) corresponds to U
ISTAT (3) corresponds to A
ISTAT (4) corresponds to D
· /IXADJE/
INTEGER IDINIT, IDXAXD
COMMON /IXADJE/ IDINIT, IDXAXD
IDINIT
is worth 5, beginning of the zone managed memory,
IDXAXD
initial position in ISZON for research.
· /IZONJE/
INTEGER LK1ZON, JK1ZON, LISZON, JISZON, ISZON (1)
COMMON /IZONJE/ LK1ZON, JK1ZON, LISZON, JISZON

EQUIVALENCE (ISZON (1), K1ZON (1))
LK1ZON
length in character (CHARACTER*1) of the managed zone,
JK1ZON
address in K1ZON of the beginning of the zone,
LISZON
length in entirety (INTEGER*8 on CRAY, INTEGER*4 on Sun) of the zone managed,
JISZON
address in ISZON of the beginning of the zone,
ISZON
zone memory in entirety (INTEGER*8 on CRAY, INTEGER*4 on Sun) managed

---

dynamically; this table of the whole type does not form part of the variables deposited in the commun run, but it is put in equivalence with the table of character type.

The order EQUIVALENCE makes it possible to align the two tables of the whole type and character of way to be able indifferently to use one or the other following the needs.

· /KUSADI//JUSADI/

PARAMETER (NR = 5)
COMMON /KUSADI/ IUSADI (1)
COMMON /JUSADI/ JUSADI (NR)

variable of reference and position of the segment of values associated with the system object of suffix:
IACCE, JIACCE
$$USADI

· /JCHAJE/

INTEGER ILLICI, JCLASS (0: 255)

COMMON /JCHAJE/ ILLICI, JCLASS
ILLICI
is worth -1,
JCLASS
is affected with the result of function ICHAR on the characters
licit, if not ILLICI is worth.
· /JENVJE/
INTEGER MSLOIS
COMMON /JENVJE/ MSLOIS
MSLOIS
mask being worth the sum of the weights of the LOIS-1 first entireties,
intended to replace the operation modulo (LAWS) by function AND
· /JCONJE/
INTEGER MSSTAT, LSSTAT
COMMON /JCONJE/ MSSTAT, LSSTAT
MSSTAT
is not used any more, mask being worth the sum of the weights of the LSSTAT first
entireties.
LSSTAT
(LBISEM - 4) where LBISEM is the length out of bit of the entirety, is used in
JELIRA to obtain the equivalent in the form of character of the statute or of
the state associated with a segment with values.
· /KATRJE/, /JKATJE
PARAMETER (NR = 5)
CHARACTER*1 GENR, STANDARD
CHARACTER*4 DOCU
CHARACTER*8 ORIG
CHARACTER*32 RNOM
COMMON /KATRJE/ GENR (8), STANDARD (8), DOCU (2), ORIG (1), RNOM (1)
COMMON /JKATJE/ JGENR (NR), JTYPE (NR), JDOCU (NR), JORIG (NR), JRNOM (NR)
variable of reference and position of the segment of values associated with the object
system of suffix:
GENR
$$GENR
GENR
$$TYPE
DOCU
$$DOCU
ORIG
$$ORIG
Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

RNOM
$$RNOM

· /KBASJE/

```
PARAMETER (NR = 5)
CHARACTER*8 NOMBAS
COMMON /KBASJE/ NOMBAS (NR)
```

NOMBA
name of the base (used for the error messages)

· /KFICJE/

```
PARAMETER (NR = 5)
CHARACTER*2 DN2
CHARACTER*5 CLASSIFIES
CHARACTER*8 NOMFIC, KSTOUT, KSTINI
COMMON /KFICJE/ CLASSIFIES, NOMFIC (NR), KSTOUT (NR), KSTINI (NR),
+ DN2 (NR)
```

DN2
unutilised!
CLASSIFY
allows to store the whole of the names of the open classes (first
letter of the basic name)
NOMFIC
name of the open bases
KSTOUT
statute at exit of the bases ("SAVES" or "DESTROYS")
KSTINI
statute in continuation of the bases ("DUMMY", "BEGINNING" or "CONTINUE")
· /KNOMJE/
CHARACTER*24 NOMEC
COMMON /KNOMJE/ NOMEC
NOMEC
name of object of collection or name to be inserted in a repertory
· /KZONJE/
CHARACTER*1 K1ZON
COMMON /KZONJE/ K1ZON (8)
K1ZON
zone memory in character (CHARACTER*1) managed dynamically.
· /NFICJE/
INTEGER NBCLA
COMMON /NFICJE/ NBCLA
NBCLA
a number of classes opened simultaneously
· /NOMCJE/
CHARACTER *24 NOMCO
CHARACTER *32 NOMUTI, NOMOS, NOMOC, BL32
COMMON /NOMCJE/ NOMUTI, NOMOS, NOMCO, NOMOC, BL32
NOMUTI
name used (in the calls to the routine I….),
NOMOS
name of the simple object,
NOMCO
name of the collection,
NOMOC
name of the object of collection,
BL32
white chain length 32.
· /UNDFJE
INTEGER LUNDEF, IDEBUG
COMMON /UNDFJE/ LUNDEF, IDEBUG

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

LUNDEF
affected in JEDEBU by the function envima ISNNEM (not has number),
IDEBUG
is worth 1 in debugg mode, 0 if not.

14 APPENDIX 2: Trees of call simplified of some subroutines

One presents below the trees of principal subroutines JEVEUX, one voluntarily has limited to three levels of subroutines to facilitate comprehension. Truncated branches indicate that there are other calls JEVEUX in the subroutine.

Tree of call of routine JEECRA

Tree of call of routine JECREO

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
40/50

Tree of call of routine JEVEUO

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
41/50

Tree of call of routine JEEXIN

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:

D9.02.02-A
Page:
42/50
Tree of call of routine JEMARQ
Tree of call of routine JEDEMA
Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
43/50
Tree of call of routine JELIBE
Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
44/50
Tree of call of routine JEDETR
Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
45/50

Tree of call of routine JECREC

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:
D9.02.02-A
Page:
46/50

Tree of call of routine JENONU

Data-processing handbook of Description
D9.02 booklet:
HI-75/97/010 - Ind A

Code_Aster ®
Version
4.0
Titrate:
Documentation of development and maintenance of the manager
Date: 28/01/1999
Author (S):
J.P. LEFEBVRE
Key:

Tree of call of routine JEDUPO

---

15 APPENDIX 3: List subroutines and of theirs principal functions

JECREC
Creation of a collection
JECREO
Creation of a simple object
JECROC
Creation of an object of collection
JEDBUG
Mode switch-over debug Jeveux
JEDEBU
Initialization of the parameters of the software
JEDEMA
Décrémente the mark and releases the objects
JEDETC
Destroyed a whole of objects
JEDETR
Destroyed an object
JEDETV
Destroyed the objects of the Volatile base
JEDISP
Determine greatest spaces available in memory
JEDUPC
Duplicate a whole of objects
JEDUPO
Duplicate an object
JEECRA
Affect the attributes of an object
JEEXIN
Test the existence of a descriptor of object
JEFINI

Finish the execution of the software
JEIMPA
Print the attributes of an object
JEIMPD
Print the list of the objects present on a basis
JEIMPM
Print the contents of the segmentation memory
JEIMPO
Print the contents of the segment of value of an object
JEIMPR
Print the contents of a repertory
JEINIF
Initialize the parameters associated with a base
JELIBE
Release an object
JELIBF
Release the whole of the objects associated with a base
JELIBZ
Release the whole of the objects associated with the mark -1
JELIRA
Reading of the value of an attribute of an object
JELSTC
Turn over the list of the objects containing a character string in their identifier
JEMARQ
Increment the current mark
JENONU
Determine the number of object of collection according to the name
JENUNO
Determine the name of object of collection according to the number
JEPRAT
Print the contents of the system objects
JERAZO
To 0 the contents of an object give
JETASS
Move the recordings within the file of direct access in order to recover space free
JEVEMA
Deliver the value of the current mark
JEVEUO
Return the position in table Z. segment of values associated with an object
JEVEUS
Return the position in table Z. segment of values associated with an object and position the mark with -3

JEVEUT
Return the position in table Z. segment of values associated with an object and position the mark with -1
JEXATR
Function of synchronization giving access the vector cumulated lengths of a contiguous collection variable length
JEXNOM
Function of synchronization allowing to reach by name an object of collection
JEXNUM
Function of synchronization allowing to reach by number an object of collection
JJALLC
Allowance of the system objects of collection
JJALLS
Allowance of a segment of values
JJALTY
Sorting on the type of object (before setting in memory of a segment of values)
JJANAL
Analyze chain containing the class, the kind and the type of the object
JJAREP
Enlarging of the repertories
JJCODN
Insertion or research in a repertory of collection
JJCREC
Creation of the system objects of collection

JJCREN
Insertion or research in the repertories of the bases
JJCROC

Treatment of the name of object of collection
JJECRS
Actualization of the descriptors
JJIMPO
Impression of the contents of a segment of values
JJLIBP
Release of the zone memory occupied by a segment of values
JJLIDE
Release of the segments of values
JJLIRS
Reading of the descriptors
JJMZAT
Restoring of the attributes associated with the identifier with a segment with values
JJPREM
At the time of the initialization of a repertory, the nearest prime number (appearing returns in one dated)
JJVERN
Treatment of the name Jeveux
JXALLM
Dynamic allocation of the zone memory managed by the software
JXCOPY
Recopy file of direct access associated a base with elimination of recording marked unutilised
JXDEPS
Displacement in memory of a segment of values
JXECRB
Writing of one or more recordings
JXECRO
Writing on disc of a segment of values
JXFERM
Closing of a file of direct access
JXHCOD
Function of associative addressing
JXLIBD
Release of the zone disc occupied by a segment of values
JXLIBM
Final release of the zone dynamically allocated memory
JXLIR1
Reading of the first recording associated with a base in order to recover its characteristics
JXLIRB
Reading of one or more recordings
JXLIRO

Reading on disc of a segment of values
JXLOCS
Determination of the relative address of a segment of values
JXOUVR
Opening of the logical units associated the bases
JXVERI
Examine the integrity of the segmentation memory
JXVEUO
Setting in memory of a segment of values

16

APPENDIX 4: GLOSSARY

Base
Together of files of direct access: the TOTAL base is
constituted of the files glob.1, glob.2, glob.3,…
Classify
Named by the first letter of the base, the class allows
to associate an object JEVEUX a file.
Identifier of object JEVEUX
For a simple object or a collection it is the number
of order of insertion in the repertory of the base, for an object
of collection it is the identifier of collection and the number
of order of insertion in the collection.
Descriptor of a segment of values
One of the 8 entireties framing a segment of values in
memory or one of the 3 entireties preceding a segment by
values on a recording (disc or plug)
File
access
direct
File from which the recordings are accessible
directly by name or number
Object JEVEUX
Indicate at the same time the simple objects, the collections and them
objects of collection
Simple object
Object from which the attributes are directly accessible among
the system objects associated with the various classes
Objects of collection
Objects whose attributes are shared and managed in
simple objects created with the collection
Segment of values
Together values associated with an object JEVEUX and
positioned in a contiguous way in memory or on disc

System object
Simple objects managed by the software and intended to collect them
values of the various attributes.

Debug
Jeveux
Option of use of Jeveux allowing to cause it
immediate unloading of the segments of values at the time of
releases and the assignment with value UNDEF (CRAY) or NaN
(Sun Solaris) of the released segment.

Index:
With
Diffusion:
Developers
Data-processing handbook of Description
D9.02 booklet:
D9.02.03 document
Description of the routine CALCULATION

1 Introduction

The routine CALCULATION is the routine of encapsulation of all elementary calculations: matrices, vectors
or elementary fields. Its general description is made in [D3.02.01].
In this description being used for maintenance of the routine CALCULATION, we will pésenterons successively
:

1) simplified flow charts of the principal routines of CALCULATION (aspect "dynamic" of program) [§2] and [§3],
2) then handled **data** ("static" aspect). It is:
   - nodes, meshs [D3.01.01], type_element, finite elements, sizes [§4], of the GREL, a LIGREL, an option [D3.02.01] and [D4.06.02],
   - catalogues of finite elements [D4.04.01],
   - FIELDS: CHAM_NO, CHAM_ELEM, CHART and RESUELEM [D4.06.05],
   - "wide CHARTS" [§5],
   - objects JEVEUX of work specific to the "package" CALCULATION [§6],
   - as well as COMMONS specific to the "package" CALCULATION [§7].
3) finally, us listels uses of the COMMONS by the various routines [§8]

---

2 Flow charts

2.1 Simplified flow chart of the routine CALCULATION

The names of the routines are written in capital letters Italic.

CALCULATION (...)
DEBCAL! “prologue” of the routine CALCULATION:
! setting in COMMON of addresses JEVEUX
! “extension” of the CHARTS “IN” [§2]
! various checks
ALCHLO! allowance of the local fields
ALRSLT! allowances of the total FIELDS results (“OUT”)
buckle on the GREL of the LIGREL: grel
INIGRL! allowances of the objects “&INEL.XXX” for the GREL grel
MECOE1! update of COMMONS ICOELXX
buckle on the parameres “IN”: bread
EXTRAI! extraction of the local field associated bread
! for the GREL grel
if necessary
CONVER! “conversion” of the local field associated bread
end if
fine bread buckles
ZECHLO! zero setting of the local fields “out”
TE0000! call to the effective TE000I (elementary calculations)
RISE! recopy local fields “out” in the fields
! total results
fine grel buckles

2.2 Simplified flow charts of routine EXTRAI

EXTRAI (champ_in,…)
so STANDARD (champ_in) = ' CARTE'

EXCART (champ_in)
so STANDARD (champ_in) = ' CHAM_ELEM
EXCHML (champ_in)
so STANDARD (champ_in) = ' CHAM_NO'
EXCHNO (champ_in)
so STANDARD (champ_in) = ' RESUELEM
EXRESL (champ_in)
end if
EXCART (champ_in,…)
buckle on the elements of the GREL: iel
TRIGD
!
recovery of the size carried
!
by the mesh ima associated the element iel
fine iel buckles
EXCHNO (champ_in,…)
buckle on the elements of the GREL: iel
buckle on the nodes of the mesh associated with iel: ino
TRIGD
!
recovery of the size carried
!
by the node ino of the element iel
fine ino buckles
fine iel buckles
EXCHML (champ_in,…)
buckle on the elements of the GREL: iel
JACOPO
! recopy “end” of the total field in the local field
fine iel buckles
Data-processing handbook of Description
D9.02 booklet: Index A

EXRESL (idem EXCHML)

2.3
Simplified flow chart of routine ALRSLT

ALRSLT (l_champ_out,…)
buckle on the fields of l_champ_out: chout
if EXISTS (chout)
ASSDEL (chout)! destruction of the field
end if
so STANDARD (chout) = ' CHAM_ELEM'
ALCHML (chout)
so STANDARD (chout) = ' RESUELEM'
ALRESL (chout)
end if
fine chout buckles

2.4
Simplified flow chart of routine CONVER

CONVER (ch_loc, numero_conversion)
JACOPO
! recopy local field ch_loc in “&&CALCUL.ORIGINAL”
if numero_conversion = -1
EXPAND ()
! conversion of the type “expand”:
! “&&CALCUL.ORIGINAL” - > “&&CALCUL.CONVERTI”
if numero_conversion = -2
MOYENN ()
! “average” conversion of the type:
! “&&CALCUL.ORIGINAL” - > “&&CALCUL.CONVERTI”
if not
TE0000 ()
! “particular” conversion:
! “&&CALCUL.ORIGINAL” - > “&&CALCUL.CONVERTI”
end if
JACOPO
! recopy “&&CALCUL.CONVERTI” in the local field ch_loc

3
Summary description of the utility routines

ALCHLO
allocate the local fields
ALCHML
allocate a field result of the type CHAM_ELEM
ALRESL
allocate a field result of the type RESUELEM
ALRSLT
allocate the total fields result
ASSDEL
destroyed a field
CONVER
fact the conversion of a local field of a local mode towards another
DCHLMX
the dimension max. of a local field makes
DEBCAL
"prologue" with the routine CALCULATION
DIGDE2
the dimension of a local field returns
ETENCA
a CHART "extends":
creation of the objects CHART ".PTMA" and CHART ".PTMS"
EXCART
extract a local field starting from a CHART
EXCHML
extract a local field starting from a CHAM_ELEM
EXCHNO
extract a local field starting from a CHAM_NO
EXISDG
test the existence of a CMP in a DESCRIPTEUR_GRANDEUR
EXPAND

fact a “conversion” of type “expand” to a local field
EXRESL
extract a local field starting from a RESUELEM
EXTRAI
extract a local field starting from a total field
GRDEUR
give the name of the size associated with a parameter
INIGRL
initialize the objects “&INEL.XXX” for a GREL
INPARA
returns the number of a parameter for a couple (type_element, option) given
NETTED
the name of the GRID associated with a LIGREL returns
MODATT
returns the mode local waited by type_element for a given parameter
RISE
recopy a local field result in a total field.
MOYENN
fact a “conversion” of type “average” to a local field
NBEC
the number of entireties coded for a given size returns
NBELEM
the number of elements of a GREL returns
NBGREL
the number of GREL of a LIGREL returns
NBPARA
returns the number of parameters for a couple (type_element, option) given
NBPARC
the number of parameters for a given option returns
NOPARA
returns the name of a parameter for a couple (type_element, option) given
NOPARC
the name of a parameter for a given option returns
NUCALC
the number of routine TE000I for a couple (type_element, option returns) given
OPCONV
the number of routine TE000I returns corresponding to a given conversion.
SCALAI
the type makes scalar: R, I, C,… of a size
TE0000
routine “hat” which calls all the TE000I
TRIGD
sort the CMPS of a size according to a DESCRIPTEUR_GRANDEUR

TYPELE
returns the type_element associated one with a GREL
ZECHLO
puts a local field at "zero" between 2 GRELS
Data-processing handbook of Description
D9.02 booklet: Index A

4
Recall on the sizes

We will appelerons "instanciée size" (or "size" very runs to reduce):

·
a reference to a "size catalogued" defined in the catalogue of the sizes,
·
a vector of realities (or complexes, entireties,…) with which the components are associated
with the CMPS of the catalogued size,
·
a DESCRIPTEUR_GRANDEUR: it is a vector of coded entireties which informs about the presence
(or the abscence) of the CMPS of the size catalogued in the instanciée size

For example, a field of displacements contains a whole of instanciées sizes. Each
size of this field is defined by:

·
a reference to size DEPL_R of the catalogue: real type, named CMPS: DX, DY,…
·
a vector of 2 realities (for example): (2.3, 3.4)
·
a coded entirety: ICODE=2 ** 2 + 2 ** 3 = 14 (for example), which makes it possible to say that for
this size, DX misses, DY is worth 2.3 and DZ is worth 3.4

A field (total or local) is primarily a list of instanciées sizes affected to
geometrical entities:

·

nodes for a CHAM_NO
·
meshs for a CHART
·
be finite elements for a RESUELEM or a CHAM_ELEM

5
"Wide" charts

A CHART is an affected field by meshs or groups of meshs. The structure of data CHART "is condensed": one stores in with respect to each instanciée size the list of the affected meshs by this size. In CALCULATION, the basic problem for the use of the CHARTS is as follows: "how to find the instanciée size associated with the mesh ima to be able to recopy it in local field provided to routines TE00IJ? "
This problem cannot be solved effectively with the structure CHART (especially if one thinks that a mesh can be affected several times: principle of "overload"), this is why at the beginning of CALCULATION (routine DEBCAL), one "extends" the CHARTS "IN" (routine ETENCA). This extension consists with
to create for each CHART, 2 temporary additional objects which make it possible to make association (mesh, size) once and for all.
These 2 objects correspond to following SD CARTE_ETENDUE:
CARTE_ETENDUE (K19):: = record
"$VIDE"
: CHART
".PTMA"
: OJB
S V I
LENGTH = nb_ma
".PTMS"
: OJB
S V I
LENGTH = nb_ms
Object ".PTMA":
that is to say nb_ma the number of meshs of the GRID associated with the chart,
for ima=1, nb_ma:
.PTMA (ima): number of the size associated with the mesh ima
Object ".PTMS":
that is to say nb_ms the number of additional meshs of the LIGREL associated with the chart,
for ims=1, nb_ms:
.PTMA (ims): number of the size associated with the additional mesh ims

---

6 Objects of work

The routine CALCULATION allocates objects JEVEUX of work (bases VOLATILE) that we will describe Ci
below. All these objects have a name starting with "&&CALCUL".

6.1 &&CALCUL.OBJETS_TRAV OJB S V K24

This object contains the name of all the objects of work created by CALCULATION. It is used to do the "housework"
(destruction) at the end of the routine CALCULATION.

6.2 Local fields

One calls "local field" object JEVEUX (and the zone associated memory) where they are stored information extracted from a total field if "IN" (or calculated by a routine TE000I if "OUT"). These information "is arranged" in accordance with the description given in the catalogue of type_element (mode_local of the parameter associated with the field).

The local fields are vectors JEVEUX whose names are form
"&&CALCUL" //nom_parameter, for example:
"&&CALCUL.PGEOMER", "&&CALCUL.PCACOQU",…

Note:

- One continues to use the "local" vocabulary although since 1993, the "local" field contains (put end to end) information concerning all the finite elements of the GREL running.
- When a routine TE00IJ makes for example:
  CALL JEVECH ("PGEOMER", "It, 'IAD)
  returned address IAD is the address of the object "&&CALCUL.PGEOMER" shifted of what it is necessary so that one reaches information concerning the current element.

The organization of these objects is as follows:
1st CMP
1st size

1st element
2nd CMP
1st size
1st element
3rd CMP
1st size
1st element
1st CMP
2nd size
1st element
2nd CMP
2nd size
1st element
...
1st CMP
1st size
2nd element
2nd CMP
1st size
2nd element
...

One finds there end to end the various elements of the GREL (which has the same one type_element and thus it
even mode_local). For each element, the description of the local field is in conformity with [D3.02.01 §4].

---

example for a local field of geometry to the nodes of a GREL of TRIA3:
DX

Node: 1
1st element
DY
Node: 1
1st element
DX
Node: 2
1st element
DY
Node: 2
1st element
DX
Node: 3
1st element
DY
Node: 3
1st element
DX
Node: 1
2nd element
DY
Node: 1
2nd element
...
.

Local fields of work for conversions:
For conversions, CALCULATION needs zones memory “plug”. These zones memory have even organization that local fields above. It is necessary to allocate of them 2 for each scalar type associated the sizes of the fields of calculation: I, R, C, K8,…
Their names are:
&&CALCUL.ORIGINAL.C
&&CALCUL.CONVERTI.C
&&CALCUL.ORIGINAL.I
&&CALCUL.CONVERTI.I
&&CALCUL.ORIGINAL.K8
&&CALCUL.CONVERTI.K8
&&CALCUL.ORIGINAL.K16
&&CALCUL.CONVERTI.K16
&&CALCUL.ORIGINAL.K24
&&CALCUL.CONVERTI.K24
&&CALCUL.ORIGINAL.R
&&CALCUL.CONVERTI.R
For the conversion of a local field of real type (for example), conversion will use in entry it

local field: &&CALCUL.ORIGINAL.R and at exit: &&CALCUL.CONVERTI.R
These local fields “plug” are allocated with the max. length of the local fields “ordinary”.

6.3
&&CALCUL.NOM_&INEL objects and &&CALCUL.IAD_&INEL

·

&&CALCUL.NOM_&INEL
This object contains the **names** of the objects related to the initialization of type_element of the courrant GREL:
objects “&INEL.XXX”.

·

&&CALCUL.IAD_&INEL
This object contains the **addresses** of the objects related to the initialization of type_element of the courrant GREL
: objects “&INEL.XXX”.

Data-processing handbook of Description
D9.02 booklet: Index A

6.4
Objects &&CALCUL.TYPE_SCA &&CALCUL.IA_CHLOC and &&CALCUL.MODELO

·
&&CALCUL.TYPE_SCA
S V K8
will dim=nb_para
·
&&CALCUL.IA_CHLOC
S V I
will dim=7*nb_para
·
&&CALCUL.MODELO
S V I
will dim=nb_para

That is to say will nb_para the number of parameters ("in" and "out") of the option of calculation and ipar the number of such
parameter.

&&CALCUL.TYPE_SCA (ipar)
scalar type (I, R, C,…) size associated with
parameter ipar
&&CALCUL.IA_CHLOC (7* (ipar-1) +1)
address local field associated with ipar
&&CALCUL.IA_CHLOC (7* (ipar-1) +2)
length of the local field associated ipar
&&CALCUL.IA_CHLOC (7* (ipar-1) +3)
local mode awaited for the parameter
&&CALCUL.IA_CHLOC (7* (ipar-1) +4)
working length of the local field for 1 element.
(account of the ICOEF takes)
&&CALCUL.IA_CHLOC (7* (ipar-1) +5)

type of the local field:
1: CHART
2: CHAM_NO
3: CHAM_ELEM
4: VECTOR 2ND MEMBER
5: STAMP
&&CALCUL.IA_CHLOC (7* (ipar-1) +6)
a number of points of discretization of the local field:
1 for
one
CHART
nb_no for
one
CHAM_NO
nb_pg for
one
CHAM_ELEM
nb_no for
one
VECTOR
O for
one
STAMP
&&CALCUL.IA_CHLOC (7* (ipar-1) +7)
value of the "multiplying" coefficient (ICOEF) for
CHAM_ELEM with dynamic size (local modes Zxxxxx)
&&CALCUL.MODELO (ipar)
mode_local associated the parameter ipar

9/16

6.5

&&CALCUL.LCHIN_EXI objects, &&CALCUL.LCHIN_K8 and &&CALCUL.LCHIN_I

·

&&CALCUL.LCHIN_EXI
S V L
dim= nb_in

·

&&CALCUL.LCHIN_K8
S V K8
dim= 11*nb_in

·

&&CALCUL.LCHIN_I
S V I
dim= 2*nb_in

That is to say nb_in the number of fields “in” and iin =1, nb_in

&&CALCUL.LCHIN_EXI (iin)
.FALSE. : the field does not exist
&&CALCUL.LCHIN_K8 (2* (iin-1) +1)
type of the field:
“CHNO”, “CART”, “CHML” OR “RESL”
&&CALCUL.LCHIN_K8 (2* (iin-1) +2)
scalar type of the field:
“It, “R”, “I”, “K8”,…
&&CALCUL.LCHIN_I (11* (iin-1) +1)
IGD: size associated with the field
&&CALCUL.LCHIN_I (11* (iin-1) +2)
NEC: a number of coded entireties
&&CALCUL.LCHIN_I (11* (iin-1) +3)
NCMPMX: a max. number of CMPS for IGD
&&CALCUL.LCHIN_I (11* (iin-1) +4)
IADESC: address CHIN//“.DESC”
&&CALCUL.LCHIN_I (11* (iin-1) +5)
IAVALE: address CHIN//“.VALE”
&&CALCUL.LCHIN_I (11* (iin-1) +6)
IAPTMA: address CHIN//“.PTMA” (if CHART)
&&CALCUL.LCHIN_I (11* (iin-1) +7)
IAPTMS: address CHIN//“.PTMS” (if CHART)
&&CALCUL.LCHIN_I (11* (iin-1) +8)
IAPRN1: address .PRNO ($MAILLA) (if CHAM_NO)
&&CALCUL.LCHIN_I (11* (iin-1) +9)
IAPRN2: address .PRNO (LIGREL) (if CHAM_NO)
&&CALCUL.LCHIN_I (11* (iin-1) +10)

IANUEQ: address .NUEQ (if CHAM_NO)
&&CALCUL.LCHIN_I (11* (iin-1) +11)
1: IANUEQ is valid
0: if not

6.6
&&CALCUL.LCHOU_K8 objects and &&CALCUL.LCHOU_I

.
&&CALCUL.LCHOU_K8
S V K8
dim= 2*nb_out
.
&&CALCUL.LCHOU_I
S V I
dim= 2*nb_out
That is to say nb_out the number of fields “out” and iout =1, nb_out
&&CALCUL.LCHOU_K8 (2* (iout-1) +1)
type of the field:
“CHML” (CHAM_ELEM) or “RESL” (RESUELEM)
&&CALCUL.LCHOU_K8 (2* (iout-1) +2)
scalar type of the field:
“It, “R”
&&CALCUL.LCHOU_I (2* (iout-1) +1)
IADESC: address CHOUT//“.DESC”
&&CALCUL.LCHOU_I (2* (iout-1) +2)
IAVALE: address CHOUT//“.VALE”
(if CHAM_ELEM)

6.7

&&CALCUL.SCALAIRE objects and &&CALCUL.IA_CONVERS

·

&&CALCUL.SCALAIRE
S V K8
dim= nb_scal

·

&&CALCUL.IA_CONVERS
S V I
dim= 2*nb_scal + 1
That is to say nb_scal the number of the scalar types: I, R, C,…
&&CALCUL.SCALAIRE (I)
ième possible scalar type of the fields “in” field:
SCAL (I)
&&CALCUL.IA_CONVERS (1)
nb_scal
&&CALCUL.IA_CONVERS (2* (i-1) +2)
address in ZR or ZC or… of the object:
“&&CALCUL.ORIGINAL. ” //SCAL (I)
&&CALCUL.IA_CONVERS (2* (i-1) +3)
address in ZR or ZC or… of the object:
“&&CALCUL.CONVERTI. ” //SCAL (I)

6.8

&&CALCUL.TECAEL_K24 objects and &&CALCUL.TECAEL_I

·

&&CALCUL.TECAEL_K24
S V K24

·

&&CALCUL.TECAEL_I
S V I
that is to say nb_no the number of nodes of the mesh associated with the courrant element.
&&CALCUL.TECAEL_K24 (1)
name of the GRID
&&CALCUL.TECAEL_K24 (2)
name of the LIGREL
&&CALCUL.TECAEL_K24 (3)
name of the mesh
&&CALCUL.TECAEL_K24 (3+1)
name of the 1st node of the mesh
&&CALCUL.TECAEL_K24 (3+2)
name of the 2nd node of the mesh
...
...
&&CALCUL.TECAEL_I (1)

number of the mesh
&&CALCUL.TECAEL_I (2)
a number of nodes of the mesh (nb_no)
&&CALCUL.TECAEL_I (2+1)
number of the 1st node of the mesh
&&CALCUL.TECAEL_I (2+2)
number of the 2nd node of the mesh
...
...

6.9
&&CALCUL.NOMOP objects and &&CALCUL.NOMTE

·
&&CALCUL.NOMOP
S V K16
·
&&CALCUL.NOMTE
S V K16
&&CALCUL.NOMOP (iopt)
name of the option of number iopt
&&CALCUL.NOMTE (ite)
name of type_element of number ite

7
Description of the COMMONS suitable for the routine CALCULATION: ICOELXX

7.1
COMMONS ICOEL1 and ICOEL2

Summary description:
Information concerning the total field and the local field associated the parameter running in the loop

on the extraction.
COMMON /ICOEL1/IGD, NEC, NCMPMX, IACHIN, IACHLO, IICHIN, IANUEQ, LPRNO
COMMON /ICOEL2/TYPEGD
CHARACTER*8 TYPEGD
C THESE COMMONS ARE UPDATED BY EXTRAI.
C IGD: NUMBER OF THE SIZE ASSOCIATED WITH THE FIELD TO BE EXTRACTED
C NEC: A NUMBER Of ENTIRETIES CODES OF IGD
C NCMPMX: MAX OF CMPS FOR IGD NUMBERS
C IACHIN: ADDRESS JEVEUX OF CHIN.VALE
C IACHLO: ADDRESS JEVEUX OF CHLOC.VALE (&&CALCUL.NOMPAR)
C IICHIN: NUMBER OF FIELD CHIN IN LIST LCHIN.
C IANUEQ: ADDRESS OBJECT .NUEQ OF THE PROF_CHNO ASSOCIATES POSSIBLE
C - LIES TO FIELD CHIN. (IF LPRNO=1).
C LPRNO: 1-> OBJECT .NUEQ EAST TO BE TAKEN INTO ACCOUNT
C (CHAM_NO A PROF_CHNO)
C 0-> OBJECT .NUEQ IS NOT TO TAKE INTO ACCOUNT
C (CHAM_NO A CONSTANT REPRESENTATION OR ANOTHER FIELD)
C TYPEGD: SCALAR TYPE OF SIZE IGD: “R”, “I”, “K8”,…

7.2

COMMON ICOEL3, ICOELD, ICOELE and ICOELH

Summary description:
Information concerning the conversion of a local field.
COMMON /ICOEL3/IAMOD1, IAMOD2, IAORIG, IACONV
C EC COMMON EAST UPDATED BY CONVER.
C IAMOD1: ADDRESS MODE-LOCAL BEFORE CONVERSION
C IAMOD2: ADDRESS MODE-LOCAL AFTER CONVERSION
C IAORIG: ADDRESS CHAMP-LOCAL “&&CALCUL.ORIGINAL.SCAL”
C IACONV: ADDRESS CHAMP-LOCAL “&&CALCUL.CONVERTI.SCAL”
COMMON /ICOELD/IACVRS
COMMON /ICOELE/IASCAL
C THESE COMMONS ARE UPDATED BY ALCHLO.
C THESE COMMONS ARE USE BY CONVER.
C IASCAL: ADDRESS “&&CALCUL.SCALAIRE” V (K8)
C V (1). , V (NR): POSSIBLE TYPE_SCALAIRES OF THE CHIN.
C IACVRS: ADDRESS “&&CALCUL.IA_CONVERS” V (I)
C - DIM (V) = 2*DIM (“&&CALCUL.SCALAIRE”) +1
C - V (1) = DIMENSION OF “&&CALCUL.SCALAIRE”
C - FOR I =1, NB_SCALAIRE
C - V (1+ 2* (I-1) +1) = ADDRESS IN ZR, OR ZC, OR ZI,…
C OF “&&CALCUL.ORIGINAL.SCAL (I)”
C - V (1+ 2* (I-1) +2) = ADDRESS IN ZR, OR ZC, OR ZI,…
C OF “&&CALCUL.CONVERTI.SCAL (I)”
COMMON /ICOELH/PARAM

CHARACTER*8 PARAM
C EC COMMON EAST UPDATED BY CONVER AND EAST USES IN
C TE00IJ OF CONVERSION.
C PARAM: NAME OF THE PARAMETER TO BE CONVERTED

7.3

COMMON ICOEL4 and ICOEL7

Summary description:
Information concerning the objects of the catalogue of the finite elements “&CATA.XXX”.
COMMON /ICOEL4/IAOPTT, LGCO, IAOPMO, ILOPMO, IAOPNO, ILOPNO, IAOPDS,

---

+ IAOPPA, LGOPPA, IAMLOC, ILMLOC, IACOVE, ILCOVE, IADSGD
C EC COMMON EAST INITIALIZES BY DEBCAL
C EC COMMON EAST USES A LITTLE EVERYWHERE
C IAOPTT: ADDRESS OBJECT OF THE CATALOGUE: “&CATA.TE.OPTTE”
C LGCO: LENGTH Of a COLUMN OF “&CATA.TE.OPTTE”
C (a TOTAL NUMBER Of POSSIBLE OPTIONS OF the CATALOGUE)
C IAOPMO: ADDRESS “&CATA.TE.OPTMOD”
C ILOPMO: ADDRESS PT_LONG OF “&CATA.TE.OPTMOD”
C IAOPNO: ADDRESS “&CATA.TE.OPTNOM”
C ILOPNO: ADDRESS PT_LONG OF “&CATA.TE.OPTNOM”
C IAOPDS: ADDRESS “&CATA.OP.DESCOPT (OPT)”
C IAOPPA: ADDRESS “&CATA.OP.OPTPARA (OPT)”
C LGOPPA: LENGTH OF “&CATA.OP.OPTPARA (OPT)”
C IAMLOC: ADDRESS “&CATA.TE.MODELOC”
C ILMLOC: ADDRESS PT_LONG OF “&CATA.TE.MODELOC”
C IACOVE: ADDRESS “&CATA.TE.CONVERS”
C ILCOVE: ADDRESS PT_LONG OF “&CATA.TE.CONVERS”
C IADSGD: ADDRESS “&CATA.GD.DESCRIGD”

COMMON /ICOEL7/IADFNO, IADFCA, LGCONO, LGCOCA
C EC COMMON EAST UPDATED BY DEBCAL
C EC COMMON EAST USES IN EXTRAI AND ALCHLO
C IADFNO: ADDRESS “&CATA.TE.MODEFNO”
C IADFCA: ADDRESS “&CATA.TE.MODEFCA”
C LGCONO: LENGTH D'1 COLUMN OF MODEFNO.
C LGCOCA: LENGTH D'1 COLUMN OF MODEFCA.

7.4

COMMON ICOEL5

Summary description:

Information concerning connectivities of the meshs of the grid and the late meshs

COMMON /ICOEL5/IAMACO, ILMACO, IAMSCO, ILMSCO, IALIEL, ILLIEL
C EC COMMON EAST UPDATED BY DEBCAL (OR TERLIG)
C EC COMMON EAST USES IN NUMAIL, EXCHNO,…
C IAMACO: ADDRESS CONNECTIVITY OF THE GRID
C ILMACO: ADDRESS POINTER LENGTH OF IAMACO
C IAMSCO: ADDRESS CONNECTIVITY OF MESHS SUPPL. D'1 LIGREL
C ILMSCO: ADDRESS POINTER LENGTH OF IAMSCO
C IALIEL: ADDRESS OBJECT “.LIEL” OF THE LIGREL.
C ILLIEL: ADDRESS POINTER LENGTH OF “.LIEL”.

7.5

COMMON ICOEL8

Summary description:

Information concerning the whole of all the total fields “in”.

COMMON /ICOEL8/IACHII, IACHIK, IACHIX
C EC COMMON EAST UPDATED BY DEBCAL
C EC COMMON EAST USES IN EXTRAI, EXCHNO, EXCART, EXRESL, EXCHML

C, DCHLMX
C IACHII: ADDRESS "&&CALCUL.LCHIN_I"
C IACHIK: ADDRESS "&&CALCUL.LCHIN_K8"
C IACHIX: ADDRESS "&&CALCUL.LCHIN_EXI"
C
C'&&CALCUL.LCHIN_EXI":: = V (L) (DIM = NIN)
C V (1): .FALSE. : The FIELD PARAMETER DOES NOT EXIST.
C
C'&&CALCUL.LCHIN_K8":: = V (K8) (DIM = NIN*2)
C V (1): TYPE_CHAMP: "CHNO", "CART", "CHML" OR "RESL".
C V (2): TYPE_GD: "It, "R", "I", "K8",…
C
C'&&CALCUL.LCHIN_I":: = V (I) (DIM = NIN*11)
C V (1): IGD ASSOCIATED SIZE A LCHIN (I)
C V (2): NEC NUMBERS ENTIRETIES CODES
C V (3): NCMPMX NUMBERS MAX OF CMP FOR IGD
C V (4): IADESC ADDRESSES .DESC
C V (5): IAVALE ADDRESSES .VALE
C V (6): IAPTMA ADDRESSES .PTMA (FOR 1 CHART)
C V (7): IAPTMS ADDRESSES .PTMS (FOR 1 CHART)
C V (8): IAPRN1 ADDRESSES PRNO ($MAILLA) (FOR 1 CHAM_NO)
C V (9): IAPRN2 ADDRESSES PRNO (LIGREL) (FOR 1 CHAM_NO)
C V (10): IANUEQ ADDRESSES .NUEQ (FOR 1 CHAM_NO)
C V (11): LPRNO (KNOWN AS IF IANUEQ EAST USES FOR 1 CHAM_NO)

7.6
COMMON ICOEL9

Summary description:
Information concerning the names of the options and type_element as well as the list of the objects of work
routine CALCULATION.

COMMON /ICOEL9/IANOOP, IANOTE, NBOBTR, IAOBTR, NBOBMX
C EC COMMON EAST UPDATED BY DEBCAL
C EC COMMON EAST USES IN TE0000 FOR
C IANOOP: ADDRESS IN ZK16 OF "&&CALCUL.NOMOP" V (K16)
C V (IOP) --> NAME OF OPTION IOP
C IANOTE: ADDRESS IN ZK16 OF "&&CALCUL.NOMTE" V (K16)
C V (ITE) --> NAME OF TYPE_ELEMENT ITE
C EC COMMON EAST USES IN ALCHLO, ALRSLT AND CALCULATION FOR:
C NBOBTR: A NUMBER Of OBJECTS OF WORK "&&CALCUL…." WHICH
C WILL HAVE TO BE DESTROYED AT THE END OF CALCULATION.
C IAOBTR: ADDRESS IN ZK24 OF THE OBJECT "&&CALCUL.OBJETS_TRAV"
C NBOBMX: LENGTH OF THE OBJECT "&&CALCUL.OBJETS_TRAV"

7.7 COMMON ICOELA

Summary description:
number of the GREL running, number of the element running (in the GREL), characteristic of the unit local fields.

```
COMMON /ICOELA/IAWMOL, NPARIO, IAWLOC, IAWTYP, NBELGR, IGR, IEL
C EC COMMON EAST INITILISE BY ALCHLO
C EC COMMON EAST MODIFIES BY MECOE1 (OBJECT .IA_CHLOC)
C EC COMMON EAST MODIFIES BY EXTRAI, RISE, CALCULATION, CONVER, ZECHLO
C (OBJECT .MODELO)
C EC COMMON EAST MODIFIES A LITTLE EVERYWHERE FOR NBELGR, IGR, IEL
C EC COMMON EAST USES IN EXTRAI, RISE, CALCULATION, CONVER,
C JEVECH, ZECHLO, TECACH
C
C IGR: NUMBER OF THE GREL WHICH ONE TREATS
C NBELGR: A NUMBER Of ELEMENTS IN GREL IGR
C (IGR AND NBELGR ARE UPDATED BY CALCULATION)
C IEL: NUMBER OF THE ELEMENT (IN GREL IGR)
C (IEL EAST UPDATED BY EXTRAI, TE0000, CONVER, ASSEMBLED,…)
C IAWMOL: ADDRESS IN ZI OF “&&CALCUL.MODELO” V (I)
C V (IPAR) --> LOCAL MODE OF PARAMETER IPAR (OPT)
C NPARIO: LENGTH OF “&&CALCUL.MODELO” (NB OF PARAM IN/OUT (OPT))
C
C IAWLOC: ADDRESS IN ZI OF “&&CALCUL.IA_CHLOC” V (I)
C THIS OBJECT CONTAINS INFORMATION ON THE CHAMP_LOCAUX
C V (7* (IPAR-1) +1) --> ADDRESS OF THE CHAMP_LOCAL '&&CALCUL. //NOMPAR (IPAR)
C V (7* (IPAR-1) +2) --> LENGTH OF THE OBJECT '&&CALCUL. //NOMPAR (IPAR)
C L=MAX (TYPE_ELEM PRESENT) *NBELGR
```

C V (7* (IPAR-1) +3) --> LOCAL MODE AWAITED FOR PARAMETER (IPAR)
C V (7* (IPAR-1) +4) --> WORKING LENGTH OF THE CHAMP_LOCAL FOR 1 ELEMENT
C TYPE_ELEM ASSOCIATES GREL IGR: NCMPEL (MODE)
C THIS LENGTH TAKES ACCOUNT Of a POSSIBLE ICOEF
C = 1 (CF V (7* (IPAR-1) +7).
C V (7* (IPAR-1) +5) --> STANDARD OF CHAMP_LOCAL:
C 1: CHART
C 2: CHAM_NO
C 3: CHAM_ELEM
C 4: VECTOR 2ND MEMBER
C 5: STAMP
C V (7* (IPAR-1) +6) --> A NUMBER OF POINTS OF DISCRETIZATION OF THE CHAMP_LOC
C (0 IF MATRIX)
C V (7* (IPAR-1) +7) --> VALUE OF THE MULTIPLYING COEFFICIENT FOR
C LOCAL MODES OF CHAM_ELEM “” (ICOEF).
C
C IAWTYP: ADDRESS IN ZK8 OF “&&CALCUL.TYPE_SCA” V (K8)
C V (IPAR) --> TYPE_SCALAIRE OF THE CHAMP_LOCAL

7.8

COMMON ICOELC

Summary description:
Information concerning the whole of all the total fields “out”.
COMMON /ICOELC/IACHOI, IACHOK
C EC COMMON EAST UPDATED BY ALRSLT
C EC COMMON EAST USES IN RISE, DCHLMX
C IACHOI: ADDRESS “&&CALCUL.LCHOU_I”
C IACHOK: ADDRESS “&&CALCUL.LCHOU_K8”
C
C'&&CALCUL.LCHOU_K8”:: = V (K8) (DIM = NIN*2)
C V (1): TYPE_CHAMP: “CHML” OR “RESL”.
C V (2): TYPE_GD: “It, “R”
C
C'&&CALCUL.LCHOU_I”:: = V (I) (DIM = NOUT*2)
C V (1): ADDRESS L_CHOUT (I) .DESC
C V (2): ADDRESS L_CHOUT (I) .VALE (IF CHAM_ELEM)
C

7.9

COMMON ICOELF

Summary description:

Information concerning the whole of the objects related to the intialisation of type_element: "&INEL. XXX"

COMMON /ICOELF/NBOBJ, IAINEL, ININEL
C NBOBJ: A MANY OBJECTS "" CREATE BY INITIALIZATION
C TYPE_ELEM
C ININEL: ADDRESS IN ZK24 OF THE OBJECT "&&CALCUL.NOM_&INEL"
C WHO CONTAINS THE NAMES OF THE OBJECTS ""
C IAINEL: ADDRESS IN ZI OF THE OBJECT "&&CALCUL.IAD_&INEL"
C WHO CONTAINS THE ADDRESSES OF THE OBJECTS ""
C EC COMMON EAST INITIALIZES BY DEBCAL
C EC COMMON EAST USES BY CALCULATION AND JEVETE

7.10

COMMON ICOELG

Summary description:

Information concerning the mesh subjacent with the current element

COMMON /ICOELG/ICAELI, ICAELK
C EC COMMON EAST CREATES BY DEBCAL.
C IT EAST USES BY TECAEL
C ICAELK EAST the ADDRESS Of a VECTOR DE K24 CONTAINING:
C V (1): NAME OF the GRID (K8)
C V (2): NAME OF the LIGREL (K19)
C V (3): NAME OF the MESH (K8)
C V (3+ 1): NAME OF the 1st NODE OF the MESH
C V (3+ 1): NAME OF DER NODE OF THE MESH
C ICAELI EAST the ADDRESS Of a VECTOR OF IS CONTAINING:
C V (1): NUMBER OF THE MESH
C V (2): A NUMBER OF NODES OF MESH (NBNO)
C V (2+ 1): NUMBER OF the 1st NODE OF the MESH
C V (2+NBNO): NUMBER OF DER NODE OF THE MESH
C V (2+NBNO +1): NUMBER OF THE GREL
C V (2+NBNO +2): NUMBER OF THE ELEMENT IN THE GREL

8 Use of the COMMONS: ICOELXX

8.1 List routines using COMMONS ICOELXX

These routines should not be called apart from the routine CALCULATION

ALCHLO EXRESL NBPARA
ALRSLT EXTRAI NOPARA
CALCULATION GRDEUR NUCALC
CONVER INPARA OPCONV
DCHLMX JEVECH TE0000
DEBCAL JEVETE TECACH
DIGDE2 MECOE1 TECAEL
EXCART MECOEL TRIGD
EXCHML MODATT ZECHLO
ASSEMBLED EXCHNO
EXPAND MOYENN

8.2 Use of COMMONS ICOELXX

Common block cross-country race-reference list:

Common Block ICOEL1 used in:
EXCART EXCHML EXCHNO EXRESL EXTRAI TRIGD

Common Block ICOEL2 used in:
CONVER EXCHML EXPAND EXRESL EXTRAI MOYENN
TRIGD

Common Block ICOEL3 used in:
CONVER EXPAND MOYENN (+ routines TEOOIJ of conversion)

Common Block ICOEL4 used in:

CALCULATION CONVER DEBCAL DIGDE2 EXCART EXCHNO
EXTRAI GRDEUR INPARA JEVECH MECOE1 MODATT
RISE NBPARA NOPARA NUCALC OPCONV TECACH
ZECHLO
Common Block ICOEL5 used in:
DEBCAL EXCART EXCHNO TECAEL
Common Block ICOEL7 used in:
DCHLMX DEBCAL EXTRAI
Common Block ICOEL8 used in:
CALCULATION DCHLMX DEBCAL EXCART EXCHML EXCHNO
EXRESL EXTRAI
Common Block ICOEL9 used in:
ALCHLO ALRSLT CALCULATION DEBCAL TE0000
Common Block ICOELA used in:
ALCHLO CALCULATION CONVER EXCART EXCHML EXCHNO
ASSEMBLED EXPAND EXRESL EXTRAI JEVECH MECOE1
MOYENN TE0000 TECACH TECAEL ZECHLO
Common Block ICOELC used in:
ASSEMBLED ALRSLT DCHLMX
Common Block ICOELD used in:
ALCHLO CONVER
Common Block ICOELE used in:
ALCHLO CONVER
Common Block ICOELF used in:
CALCULATION DEBCAL JEVETE
Common Block ICOELG used in:
DEBCAL TECAEL
Common Block ICOELH used in:
CONVER
(+ routines TEOOIJ of conversion)
Data-processing handbook of Description
D9.02 booklet: Index A

Code_Aster ®
Version
7.2

Titrate:
Description of storage JEVEUX to format HDF

Date:
01/10/03
Author (S):

Key ***J.P. LEFEBVRE***
:

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D9.02 booklet: -
Document: D9.02.04

Description of storage JEVEUX to format HDF

Summary:

One describes here the organization of file HDF adopted to store the contents of a base JEVEUX. This file
an image of the whole of objects JEVEUX created on the Globale basis contains at the time of the safeguard
and can be used to launch an execution in continuation on the platform of initial execution or all another platform compatible and having the version carried of Code_Aster.

---

1 General information

By defect, the manager of memory JEVEUX employed in Code_Aster uses several files direct access of binary type. These files constitute "bases" JEVEUX, the structures of data which will again be used in the event of continuation of calculation are stored in the base TOTAL in a format specific to the manager of memory. During calculation, the manager of memory is brought to use the bases to discharge the memory temporarily, thus the mode access chosen for its performances in this mode of use is not adapted to a safeguard especially if he one wants to make them independent of the platform.
The use of bookshop HDF, already used within the framework of the interchange format of data MED, for a writing of the contents of bases JEVEUX appears much more adapted. The file obtained afterwards safeguard can be easily transferred, after a possible compression, on a platform local to carry out for example operations of postprocessing with a version carried of Code_Aster.
A file with format HDF is organized like a tree structure file Unix, the concept of "group" are connected with the concept of repertory, the concept of "dataset" corresponds to the file. Moreover it is possible to assign attributes to each level of "group" and/or to each "dataset". We have exploited these some concepts to organize the recopy of the whole of objects JEVEUX contained in a base.
HDF provides a utility (h5dump) making it possible to write with the ASCII format the contents of a whole file, it is thus very easy, for files of reasonable size, to obtain the contents of all the objects constituting a base JEVEUX.

2
Writing of objects simple JEVEUX

Objects simple JEVEUX are in general of homogeneous type, they are stored in a “dataset” bearing the name of the simple object on the level of the “group”/. Simple objects of repertory kind of names contain at the same time a table of h-coding of the type INTEGER and the list of the names stored of type CHARACTER and must be treated separately. They are divided to store two “datasets” respective names T_HCOD and T_NOM on the level of a group bearing the name of the simple object on the level of “group”/.
One associates a named list “ATTRIBUTES JEVEUX” of 5 attributes all the associated “datasets” with the simple objects and all the groups associated with the repertories with names containing respectively:

· a text (“SIMPLE OBJECT”),
· an identifier (identifying of object simple JEVEUX),
· a chain containing the class, the kind and the type with direction JEVEUX (argument of JECREO),
· associated the Fortan type,
· a null string

The objects systems JEVEUX contain all information of the type attribute JEVEUX and allow to rebuild the structure of associated data. They do not differ from the simple objects JEVEUX, accessible to the user, that by their name, they are treated in the same way. Objects systems are of homogeneous type and thus do not pose a particular problem of storage. One store each object JEVEUX in a “dataset” bearing the name of the system object on the level of “group”/.
One associates a named list “ATTRIBUTES JEVEUX” of 5 attributes all the associated “datasets” with the system objects containing respectively:

· a text (“SYSTEM OBJECT”),
· an identifier (identifying of object simple JEVEUX),
· a chain containing the class, the kind and the type with direction JEVEUX,
· associated the Fortan type,
· a null string

The attributes associated with the “dataset” or the “groups” will be used during the second reading of the file HDF to rebuild the structures of data associated with objects JEVEUX.

3
Writing of collections JEVEUX

3.1 Collections contiguous

The contiguous collections are built starting from simple objects, it is thus easy to use them methods associated with the simple objects to store these last. Objects of contiguous collection are stored in the segment of values associated with the system object $$DESO.
The object simple descriptor of collection is stored in a "dataset" bearing the name of the collection under the "group"/.
One associates a named list "ATTRIBUTES JEVEUX" of 5 attributes the associated "dataset" containing
respectively:

· a text ("COLLECTION"),
· an identifier (identifying of object simple JEVEUX),
· a chain containing the class, the kind (X) and the type with direction JEVEUX (I),
· associated the Fortan type,
· a null string.

The system objects of collection are stored in "datasets" bearing the name of each object system of collection under the "group"/.
One associates a named list "ATTRIBUTES JEVEUX" of 5 attributes the associated "dataset" containing
respectively:

· a text ("OBJ. SYSTEM COLLECTION"),
· an identifier (identifying of object simple JEVEUX),

· a chain containing the class, the kind and the type with direction JEVEUX,
· associated the Fortan type,
· a null string.

3.2 Collections
dispersed

The dispersed collections are built starting from simple objects for the systems objects of collection and of segment of values associated with each object with collection. The systems objects are stored in the same way that for the collection contiguous. The system object $$DESO is particular because there is not associated in the case of the collections dispersed with a segment with values, it is not thus not associated "dataset", and it is only by charging the contents of the objects sytèmes JEVEUX that the dispersed collection is rebuilt during the second reading of file HDF.

A "group" bearing the name of the collection supplemented by __OBJETS is created on the level of the "group"/ to write the objects of dispersed collection. One associates a named list "ATTRIBUTES JEVEUX" of 5 attributes to the "group", only the first element is nonempty and contains a text (COLLECTION).

Each object of collection is then written in a "dataset" bearing the name of the collection supplemented by the number of the object of collection (including for the named collections) under the "group" describes above.

One associates a named list "ATTRIBUTES JEVEUX" of 5 attributes the associated "dataset" containing respectively:

· a text (OBJECT OF COLLECTION),
· the name or the number of the object of collection and the identifier of collection,
· a chain containing the class, the kind and the type with direction JEVEUX,
· associated the Fortan type,
· a null string.

Date:
01/10/03
Author (S):
Key J.P. LEFEBVRE
:
D9.02.04-A Page
: 4/6

4
Heading of the file

One stores a named list "TOTAL BASE JEVEUX" of 5 attributes associated with the "group "/ containing respectively:

· a text identifying the version of Code_Aster used to build the file,
· the name of the waiter of calculation used,
· the name of the system on the waiter,
· the date of execution of the code,
· 3 characteristics machine (length out of bits of the standard entirety, length in bytes of the entirety standard, length in bytes of the unit of addressing).

The first information is used during the second reading for if required emitting an alarm when the version of Code_Aster used differs. Certain structures of data or catalogues can appear incompatible.

5
Example of file

Here an example of file HDF obtained starting from the carrying out of the test TTLL01A and of which
ASCII representation results from the utility h5dump. One gives only some extracts here illustrating descriptions of principal objects JEVEUX.

```
HDF5 "resu64.hdf" {
GROUP "/" {
ATTRIBUTE "BASES TOTAL JEVEUX" {
DATATYPE H5T_STRING {
STRSIZE 24;
STRPAD H5T_STR_SPACEPAD;
CSET H5T_CSET_ASCII;
```

CTYPE H5T_C_S1;
}
SIMPLE DATASPACE {(5)/(5)}
DATED {
“13/08/2003 7. 1.13”, “CLA4ASTR.CLA.EDF”,
“OSF1”, “MA-19-AOUT-2003 10:35: 24”,
“LBIS=64 LOIS= 8 LOUA= 1”
}
}
DATASET “&&SYS .CODE” {
DATATYPE H5T_STRING {
STRSIZE 8;
STRPAD H5T_STR_SPACEPAD;
CSET H5T_CSET_ASCII;
CTYPE H5T_C_S1;
}
SIMPLE DATASPACE {(3)/(3)}
DATED {
“TTLL01A”, “15”,” 95 "
}
ATTRIBUTE “ATTRIBUTES JEVEUX” {
DATATYPE H5T_STRING {
STRSIZE 24;
STRPAD H5T_STR_SPACEPAD;
CSET H5T_CSET_ASCII;
CTYPE H5T_C_S1;
}

D9.02.04-A Page
: 5/6

SIMPLE DATASPACE {(5)/(5)}
DATED {
"SIMPLE OBJECT", "30",
*"G V K8", "CHARACTER*8",*
" "
}
}
}
...
GROUP "TEMPLE .DESC" {
ATTRIBUTE "ATTRIBUTES JEVEUX" {
DATATYPE H5T_STRING {
STRSIZE 24;
STRPAD H5T_STR_SPACEPAD;
CSET H5T_CSET_ASCII;
CTYPE H5T_C_S1;
}
SIMPLE DATASPACE {(5)/(5)}
DATED {
"SIMPLE OBJECT", "287",
*"G NR K16", "CHARACTER*16",*
" "
}
}
DATASET "T_HCOD" {
DATATYPE H5T_STD_I64LE
SIMPLE DATASPACE {(43)/(43)}
DATED {
37, 344, 16, 18, 18, 6, 0, 0, 0, 18, 8, 6, 4, 0, 0, 0, 7, 0, 0, 0,
0, 13, 17, 12, 0, 16, 11, 2, 0, 5, 0, 15, 14, 1, 0, 9, 0, 3, 0, 10,
0, 0, 0
}
}
DATASET "T_NOM" {
DATATYPE H5T_STRING {
STRSIZE 16;
STRPAD H5T_STR_SPACEPAD;
CSET H5T_CSET_ASCII;
CTYPE H5T_C_S1;

}
SIMPLE DATASPACE {(18)/(18)}
DATED {
"TEMP", "FLUX_ELGA_TEMP", "FLUX_ELNO_TEMP",
"FLUX_NOEU_TEMP", "META_ELGA_TEMP", "META_ELNO_TEMP",
"META_NOEU_TEMP", "DURT_ELGA_META", "DURT_ELNO_META",
"DURT_NOEU_META", "HYDR_ELGA", "HYDR_ELNO_ELGA",
"HYDR_NOEU_ELGA", "DETE_ELNO_DLTE", "DETE_NOEU_DLTE",
"COMPORTHER", "ERTH_ELEM_TEMP", "ERTH_ELNO_ELEM"
}
}
}
DATASET "TEMPLE .INST" {
DATATYPE H5T_IEEE_F64LE
SIMPLE DATASPACE {(42)/(42)}
DATED {
0, 0.0001, 0.0002, 0.0003, 0.0004, 0.0005, 0.0006, 0.0007, 0.0008,
0.0009, 0.001, 0.002, 0.003, 0.004, 0.005, 0.006, 0.007, 0.008, 0.009,
0.01, 0.02, 0.1, 0.2, 0.7, 2, 1.79769e+308, 1.79769e+308,
1.79769e+308,
1.79769e+308, 1.79769e+308, 1.79769e+308, 1.79769e+308, 1.79769e+308,
1.79769e+308, 1.79769e+308, 1.79769e+308, 1.79769e+308, 1.79769e+308,
1.79769e+308, 1.79769e+308, 1.79769e+308, 1.79769e+308
}

ATTRIBUTE "ATTRIBUTES JEVEUX" {
DATATYPE H5T_STRING {
STRSIZE 24;
STRPAD H5T_STR_SPACEPAD;
CSET H5T_CSET_ASCII;
CTYPE H5T_C_S1;
}
SIMPLE DATASPACE {(5)/(5)}
DATED {
"SIMPLE OBJECT", "293",
"G V R", "REAL*8",
" "
}
}
}
...

Data-processing handbook of Description
D9.02 booklet: -
HT-66/03/007/A

Data-processing handbook of Description
D9.03 booklet: -
Document: D9.03.01

Implementation of algorithm FETI

Summary:

One describes here implementation the data-processing of the algorithm of resolution of linear systems FETI. One takes the notations of the notes of Reference [R6.01.03] [bib1] and one again support also on those of Use [U4.50.01] and of Development [D4.06.05], [D4.06.07], [D4.06.10], [D4.06.11] and [D4.06.21]. One will find in this document the simplified flow chart of the process of resolution, in sequential mode as in parallel, allowing to distinguish its principal logical articulations, its tree of call, the principal variables accused like their contents. Specificities and the philosophy of parallelism set up are particularly detailed.
One has on the other hand chooses not to weigh down the talk by mentioning the calls to the routines supervisor, with those of management of objects JEVEUX, handling of low level (VTDEFS, VTCREB…) and details of the routines preliminaries with the unfolding of algorithm FETI (ASSMAM, MEACMV, MERESO…).

Titrate:
Implementation of algorithm FETI

Date:
15/09/05
Author (S):
O. BOITEAU Key
:
D9.03.01-A Page
: 2/14

1 Flow chart
simplified

Principal routines
Total function

Specificity of parallelism

Reading of the data of calculation
All the processors carried out
Before
(grid, materials…), same operations until it
order
pretreatments… Entry in
level there and thus have access to
calling FETI
possible loops of calculation:
entirety of data JEVEUX
increment of load, not of
known (grid, fields resulting
Newton…
pretreatments…).
Preparation of the data solvor: SD With the processor J are associated them
CRESOL
Main SOLVEUR and slaves.
j1 under-fields, j2… jk. The SD
Main SOLVEUR is built by
each proc. and its .FETS
point that on SD SOLVEUR of
under-fields slaves jk:

.FETS SD SOLVEUR jk
Renumérotation and factorization Idem on
NUMBER
symbolic system, constitution of the SD
NUME_DDL:
Main NUME_DDL and slaves.
.FETN SD NUME_DDL jk
Assemblies of the matrices of rigidity and
Idem on
vectors second local members.
MATR_ASSE/CHAM_NO:
ASSMAM
Filling of
.FETM
SD
SD MATR_ASSE jk
ASSVEC
MATR_ASSE/CHAM_NO
Masters and .FETC SD CHAM_NO jk
slaves.
Factorization of matrices of rigidity
Each proc. J calculates them
PRERES
data relating to sound
local (Ki) + and seeks of theirs
perimeter of under-fields
:
rigid modes Bi.
(Kj) + and B.
K
jk
Resolution via
the algorithm
RESOUD/
ALFETI
FETI itself, to see
[Figure 1-b].

Appear 1-a: Flow chart simplified before ALFETI
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

G: = R B
R B
Known Objets JEVEUX that
FETGGT
Construction of
I
[1 1 K Q Q] of the proc. 0
and of G T
I GI.
0
T
T
E: = F B
F B
and
1
1
K Q
e0 known of the proc 0 and 0 of
FETINL
Calculation of
[
] T
Q
-1
all.

0
: = G
.
I [G T G
E
I
I
]
of
(
)
+
Construction of 0
R: =
Known that proc. 0.
FETRIN
R
-
.
I (K I) (
T
0
F
R
I
I
)
0
-
G: = Pr =
FETPRJ
(
1
0
I - G G T G
G T
.
I
I
I
I
)
Calculation of

([)] () 0r Connu of all proc.
~
FETSCA/
Calculation of 0
1
-
0
H: = AM Ag.
Known that proc. 0.
FETPRC
Known of all proc.
Calculation of 0
~0
H: = pH and p0=h0.
FETPRJ
Calculation of the local solution with
ground
U known of the proc.
yes
I
Test of stop
each ground under-field
U
and
I
associated, ground
U of the proc. 0.
0
G 0?
rebuilding of total the ground
U.
not

Buckle

GCPPC

to see [Figure 1-c].

Appear 1-b: Flow chart simplified in ALFETI (level 1)

Data-processing handbook of Description
D9.03 booklet: -

Iteration K
+
Construction of K
Z =
:
Known that proc. 0.
FETFIV
R K R p.
I (
I)
T
K
I
Calculation of the parameter of descent K.
Known that proc. 0.
DDOT
K +1
K
K
K
FETPRJ

= + p
DAXPY
Updates:

.
k+1 known that proc.
K +1
K
K
K
G
= G - Pz
0, gk+1 of all.
ground
yes
Calculation of the local solution with
U known of the proc.
I
Test of stop?
K 1
+
0
each ground under-field
U
and
I
associated, ground
U of the proc. 0.
G
< RESI_RELA G
rebuilding of total the ground
U.
not
FETSCA/
K +1
-1
Known that proc. 0.
Calculation of
1
H
:
+
=
K
WFP Ag
.
FETPRC/

FETPRJ
Update of the direction of descent
Known of all.
FETREO
pk+1 (reorthogonalized or not).

Appear 1-c: Flow chart simplified in ALFETI (level 2)

2
Detailed flow chart and tree of call

Routine
Routines
Routines
Details
Information
appealing
called
called
relative to
level 1
level 2

processors
Before

Reading of the data of calculation (grid, All proc. have
order
materials,
SD_FETI [D4.06.21]…), carried out them
calling FETI
pretreatments…
same
*(MECA_*
Entry in possible loops of calculation:
operations
STATICS…)
increment of load, not of Newton…
up to this level
and thus access has
with the entirety of
data
Known JEVEUX
(grid,
fields resulting from
pretreatments…)
.
CRESOL

· Lecture of the parameters inherent in
O/O*
key word factor SOLVEUR [U4.50.01],

· Création and initialization of the objects
P/P
related “&FETI.INFO…” (for

monitoring [D4.06.21 §4]) and

SDFETI (1: 8)/“.MAILLE.NUMSD”
P/P
(for the routines of assembly
SD SOLVEUR
[D4.06.21 §4]),
Master is

· Création of the pointer
built by
SOLVEUR_maître.FETS towards
each proc. and
SOLVEUR slaves.
its .FETS
will point that on
SD SOLVEUR
under
fields
slaves jk
Note:

** Code O/N means that the operation is carried out, or that the variable is known, only proc. Master (O*
for Yes) but not of the others proc. (NR for Not). One uses also P (Partially) to notify that the carried out one
relate to only the under-fields of the perimeter of the proc. running.

FETMPI

· Répartition of the under-fields by proc. MPI_COMM_SIZE
and determination of the number of proc.
MPI_COMM_RANK
Buckle on
CRESO1

· Constitution of SD SOLVEUR [D4.06.11]
P/P
under
under-fields ("slaves"),

fields
· Vérifications meshs of the model/meshs Into sequential
*concerned ***
under-fields.
by the proc.
running
End of loop

CRESO1

· Constitution of SD SOLVEUR of
O/O

total field ("main"),

· Vérifications meshs of the model/meshs Into sequential
under-fields.

Note:

*** Into sequential, all the under-fields are concerned with the processor running which is also the processor*
Master or proc. 0. In parallel, the proc. running J sees itself allotted a whole of under-fields contiguous: j1,
j2… jk. In the loops on the under-fields, this information is formulated via object JEVEUX
"&FETI.LISTE.SD.MPI" ([D4.06.21 §4]) which filters the indices of loop.
NUMBER

· Constitution of the list of loads
O/O
total with all the model.
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

Routine
Routines
Routines
Details
Information
appealing
called
called
relative to
level 1
level 2
processors

NUMER2

· Constitution of the NUME_DDL [D4.06.07]
O/O
Master and of his pointer .FETN.
The .FETN
will point that on
SD of under
fields
slaves jk

NUMER2 NUEFFE
· Création of the main NUME_EQUA.
O/O

NUMER2 PROFMA
· One created not the SD main STORAGE.
O/O

· Vérification of the coherence of
O/O
SD_FETI with respect to the model and of
loads (controlled by the key word
VERIF_SDFETI).

FETMPI

· Détermination of the number of proc. and of the MPI_COMM_SIZE
row of the proc. running.
MPI_COMM_RANK
Buckle on
EXLIM1
· Création of the LIGREL of the meshs
P/P
under
physiques of the under-field.
fields
concerned with
the proc.
running

EXLIM2
· Constitution of the list of the LIGREL of
P/P
charge (with late meshs) impacting it

under-field,

· Their possible projections on
P/P
several under-fields. Filling
ad hoc of the SD .FELi associated these
projections.

NUMER2

· Constitution of the NUME_DDL slave.
P/P

NUMER2 NUEFFE
· Création of the NUME_EQUA slave.
P/P
End of loop
NUMER2 PROFMA
· One created the SD slave STORAGE.
P/P

· Test of the identity of the PROF_CHNO.NUEQ
O/N
([D4.06.07 §5.3])
ASSMAM/

· Constitution of the MATR_ASSE/CHAM_NO
O/O
ASSVEC
[D4.06.10], [D4.06.05] main and of theirs
Theirs
pointers .FETM/.FETC. One does not build .FETM/.FETC
not the useless MATR_ASSE.VALE.
will point that
on the SD of
under-fields
slaves jk
Buckle on

Constitution of the MATR_ASSE/CHAM_NO
P/P
under
slaves while being pressed on the objects
fields
auxiliaries:
concerned with
· SDFETI (1: 8)/".MAILLE.NUMSD"
the proc.
who determines in the loops on
running
total data if the mesh considered
End of loop

interest the under-field,
· .FELi to be able to make the joint
between the classification of meshs or of
late nodes and their local classification
with the under-field (that of the NUME_DDL
slave).
PRERES

· Up to date Mise of field "DOCU" of
O/O
MATR_ASSE maître.REFA to cross
RESOUD and the recopy of the second member
in vector solution.
Buckle on
FETFAC TLDLG2
· Remplissage of the MATR ADZE
P/P
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

Routine
Routines
Routines
Details
Information
appealing

called
called
relative to
level 1
level 2
processors
under
esclaves.VALF of the under-fields

fields
*associated the proc. running: (**K***
,

I) +
concerned with

the proc.
· Calcul of the rigid modes and filling
P/P
running
temporary objects

“&&FETFAC.FETI.MOCR” (modes

rigid) and “&&FETFAC.FETI.INPN”

(index of the pivots quasi-null),

· Vérifications of the rigid modes and one

conditions of Moore-Penrose (if
P/P
INFO_FETI (6: 6) = ' activated You
[U4.50.01 §3.5])
End of loop
FETFAC
· Remplissage of objects P/P
MATR_ASSE.FETF, .FETP and .FETR
*([D4.06.10 §3]): **Bi**.*
RESOUD
Buckle on

· Vérifie that the PROF_CHNO of
P/P
under

MATR_ASSE is identical to that of

fields
second member (for the Master and the SD

concerned
slaves),

by the proc.
· Vérifie that the total MATR_ASSE and its

running
MATR_ASSE slaves indeed were
P/P
End of
factorized.
buckle

RESFET
Buckle on · up to date Mise of the temporary objects &INT
P/P
under
local MATR_ASSE.
fields
concerned
by the proc.
running
End of
buckle

ALFETI
· Algorithme FETI itself, to see

following tables [Table 2-2] and
[Table 2-3].

Table 2-1: Detailed flow chart and tree of call before ALFETI

Routine
Routines
Routines
Details
Information
calling
called called level

relative to
E
level 1
2
processors
ALFETI
FETMPI
· Détermination of the number of proc. and of the MPI_COMM_SIZE
row of the proc. running.
MPI_COMM_RANK

Buckle on

· Initialisation of the collections of vectors
P/P
under
dimensioned with the number of DDLs
fields
(physics and late) of each under
concerned
fields: "&&FETI.COLLECTIONR" and
by the proc.
"&&FETI.COLLECTIONL". They
running
will be used for the matric operations of
End of
size of the local problems (
buckle
second is limited to the product stamps
vector of prepacking).

FETING
Buckle on
· Constitution of the collection of vectors
P/P
under-fields
dimensioned with the number of Lagranges
concerned with
of interface of the under-fields
proc. running
"&&FETI.COLLECTIONI". It is used for
End of loop
to make the joint enters classification
of Lagrange of interface in the list

nodes of under-fields
(SDFETI.FETB) and that of TEACHER CHNO
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

Routine
Routines
Routines
Details
Information
calling
called called level
relative to
E
level 1
2
processors
room resulting from factorization symbolic system.

· Initialisation of temporary objects related to
O/O

reorthogonalisation (REORTH,

Buckle on
NBREOR…), of temporary vectors

under-fields
(K24IRR, K24LAI…),

concerned with the · Création of temporary objects for
P/P
proc. running
to save calls later
End of loop
JEVEUO: K24REX, K24FIV….

FETMPI
· Réduction then total diffusion of In parallel (tested
object MATR_ASSE maître.FETF for by the presence
that all proc. know how much
from at least 2
rigid modes has under
processors)
field given.
O/O
MPI_ALLREDUCE+
MPI_SUM

FETMPI
· Idem for the object of monitoring In parallel
"&FETI.INFO.STOCKAGE.FVAL",
O/O

MPI_ALLREDUCE+
· As long as to synchronize one makes the same one
MPI_SUM
thing, without total diffusion, for
O/N
other objects of monitoring
MPI_REDUCE+
"&FETI.INFO…".
MPI_SUM

FETGGT
Buckle on
· Construction of the rectangular matrix
P/P
under-fields
G**: = **R B

R B

I
[1 1 K Q Q]
concerned with

proc. running
(NOMGI='&&FETI.GI.R').

End of loop

FETREX

FETGGT FETMPI
*· Construction of **G***
In parallel
I complete by collection
selective towards the proc. 0.
O/N
***ATTENTION**, it is here that MPI_GATHERV intervene*
constraints: STOCKAGE_GI=' OUI'
obligatory in parallel and distribution of
under-fields in a contiguous way by
proc.

FETGGT
BLAS DDOT
*· Construction of the square matrix **G** T*
If proc 0
*I **GI***
(NOMGGT='&&FETI.GITGI.R').
O/N

FETMON

· Monitoring if INFOFE (9: 9) = ' You: sizes
If proc 0
under-fields, profiling of theirs
O/N
time CPU of assembly, of
factorization…

FETINL

Buckle on
· Construction of vector P/P
under-fields
***e0**: = [T*
T
F B
F B

1
1
K Q Q] T
concerned with
proc. running
(K24ER= `&&FETINL.E.R').
End of loop

FETINL FETMPI
*· Construction of the complete **e0** by reduction*
In parallel

towards the proc. 0.
O/N
MPI_REDUCE+
MPI_SUM

FETINL
FETREX and
· Calcul of the Lagrange vector of interface
If proc 0
BLAS DAXPY
-1
0

O/N
*: = **G***

*I [**G** T **G***
E
I
I
]
(if
initial
(

)
STOCKAGE_GI
(VLAGI/K24LAI/ZR (IVLAGI)).
= ' OUI'),
LAPACK
DSPTRF/S
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

Routine
Routines
Routines
Details
Information
calling
called called level
relative to
E
level 1
2
processors

FETINL
· Distribution of 0 with all proc.
In parallel
O/O
MPI_BCAST

FETRIN
Buckle on
· Calcul of initial residue P/P
OPTION=1 under-fields
0
***R**: =*
R
+

-

*I (**K** I) (*
T
0
F
R
I
I
)
concerned with
proc. running,
(K24IRR= `&&FETI.RESIDU.R'/
BLAS DAXPY,
ZR (IRR))
FETREX,
RLTFR8,
End of loop

FETRIN
FETMPI
*· Construction of the complete **r0** by reduction*
In parallel
OPTION=1
towards the proc. 0.
O/N
MPI_REDUCE+
MPI_SUM

FETPRJ
BLAS
· Calcul of the initial projected residue
If proc 0
OPTION=1 DGEMV/DCOPY,

0
-
O/N
G**: = **Pr** = (**I** - **GI** [**G** T **G
I
I] 1
0
***G** T*
I
) 0
LAPACK
( )
*() **R***
DSPTRS,
(K24IRG= `&&FETI.REPROJ.G'/
FETREX and
ZR (IRG)).
BLAS DAXPY
(if
STOCKAGE_GI
= ' OUI').

FETPRJ
FETMPI
*· Distribution of **g0** to all proc.*
In parallel

O/O
MPI_BCAST

FETSCA
· Mise on an initial projected residue scale
O/O
~0
0
***G** = **Ag** (K24IR1/ZR (IR1)).*

FETPRC
Buckle on
· Calcul of packaged projected residue
P/P
under-fields
initial
0

1
***-** ~0*
G**: = **M G
concerned with
proc. running,
(K24IR2= `&&FETI.VECNBI.AUX2'/
BLAS DAXPY,
ZR (IR2)).
FETREX,
MRMULT,
End of loop

FETPRC FETMPI
· Construction of the 0
***G** complete by In parallel*
O/N
reduction towards the proc. 0.
MPI_REDUCE+
MPI_SUM

FETSCA
· Mise on a projected residue scale
If proc. 0
~
O/N
packaged initial 0
0
H = Ag
(K24IR3='&&FETI.VECNBI.AUX3'/
ZR (IR3)).

FETPRJ
BLAS
·
0
~
H**: = **pH
If proc. 0
OPTION=1
Calculation of
0
DGEMV/DCOPY,
O/N
LAPACK

(K24IRH= `&&FETI.REPCPJ.H'/
DSPTRS,
ZR (IRH)).
FETREX and
BLAS DAXPY
(if
STOCKAGE_GI
= ' OUI').

FETPRJ FETMPI
*· Distribution of **h0** to all proc.*
In parallel
O/O
MPI_BCAST
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

Routine
Routines
Routines
Details
Information
calling
called called level
relative to
E
level 1

2
processors
BLAS

*· The variable **p0** receives **h0***
O/O
DCOPY
(K24IRP='&&FETI.DD.P'/ZR (IRP))
BLAS

· Calcul of the numerator of the parameter of
If proc 0
DDOT
0
0
*: = **G** .p*
O/N

descent
0 (
NR
ALPHAN),

· Calcul of the initial standard of the residue
If proc 0

O/N
BLAS
projected
0
***G** (ANORM) and of the criterion of stop*

DNRM2
0
:=
(
K
*RESI_RELA **G***
EPSIK).

FETMPI
· Distribution of ANORM to all procs and
In parallel

calculation of EPSIK.
O/O
MPI_BCAST

· Préparation of object JEVEUX CRITER.
O/O

FETRIN

OPTION=2

Test of stop if residue
0
***G quasi-**no one*

FETPRJ
(i.e. lower than
If proc. 0
OPTION=2
*R8MIEM () ** (2.D0/3.D0)) :*
O/N

1
ground
T
-
T

FETPRJ
· Calcul of
*: = [**G G***

I
I]
0
G R
I
In parallel
FETMPI
(K24ALP='&&FETI.ALPHA.MCR'),
O/O

MPI_BCAST
Buckle on
· Distribution of 0 with all procs

under-fields
(variables K24ALP/ZR (IALPHA)).
P/P
concerned with

proc. running,
· Calcul of the local solution

BLAS DAXPY,
ground
U
= ***K*** *+* ***F*** *-* ***R*** *0*
:
- B
I
(I) (
T
I
I
) ground

FETREX,
I

RLTFR8,
(K24IRR/ZR (IRR)).

Subloops on
P/P
LIGRELs

physics and
· Reconstruction of the CHAM_NO
ground
U
I

late of
solution slave specific to the under-field
P/P

Local CHAM_NO
(CHAMLS/ZR (IVALCS)),

*· Reconstruction of the CHAM_NO **usol** solution*

Master associated with the proc. For the nodes

physics, one adds their contribution

divided beforehand by the multiplicity

End of the loops
geometrical of the aforesaid node

(
In parallel
FETMPI
K24VAL/ZR (IVALS)).

O/N
·
MPI_REDUCE+
*Construction of the complete **usol** by*
MPI_SUM
reduction towards the proc. 0.

FETARP FETPRJ
· Test of the definite-positivity of the operator
Into sequential
FETFIV
of interface
PF
ARPACK
IP if
INFO_FETI (7: 7) = ' You
DNAUPD/DNEUPD
([U4.50.01 §3.5])
BLAS DCOPY

· Allocation of the large objects related to

If proc 0
reorthogonalisation:
O/N
K24PSR='&&FETI.PS.REORTHO.R',
K24DDR='&&FETI.DD.REORTHO.R',
K24FIR='&&FETI.FIDD.REORTHO.R'.
Buckle
on

Algorithm FETI level 2 to see table

[Table 2-3].
iterations
GCPPC

Table 2-2: Detailed flow chart and tree of call in ALFETI (level 1)

Routine
Routines
Routines
Details
Information
appealing called
called

relative to
level 1
level 2
processors
ALFETI
BLAS

· If reorthogonalisation, storage of
If proc 0
DCOPY
direction of descent ***km No*** *in K24DDR.*
O/N
Buckle on
FETFIV
Buckle on the · Calcul of the result of operator FETI
P/P

under-fields
of interface on the direction of descent
iterations of
concerned with
K
Z *=*
:
R K *+*

R p
I (
I)
T
K
GCPPC
the proc. running,
I
BLAS DAXPY,
(K24IRZ= `&&FETI.FIDD.Z'/ZR (IRZ))
FETREX,
RLTFR8,
End of loop

FETFIV FETMPI
· Construction of the complete ***zk*** *by reduction*
In parallel
towards the proc. 0.

O/N
MPI_REDUCE+
MPI_SUM
BLAS
*· If reorthogonalisation, storage of **zk***
If proc 0
DCOPY
in K24FIR.
O/N
BLAS
DDOT

· Calcul of the denominator of the parameter of
If proc 0
descent running K
K
K
*: = **Z p***
. (
O/N
D
ALPHAD),

· Calcul of the parameter of descent
Idem
K

running K
NR

=
:
(ALPHA).
K
D

· If reorthogonalisation, storage of
If proc 0
ALPHAD in K24PSR.
O/N

FETTOR FETPRJ

· Test of orthogonalities of the GCPPC if
If proc 0
BLAS DDOT,
INFO_FETI (8: 8) = ' You
O/N
DCOPY
([U4.50.01 §3.5]).
BLAS
· Up to date Mise of the Lagrange vector
If proc 0
DAXPY

K +
K
K
1 = + ***p***
O/N

of interface current
K

(K24LAI/ZR (IVLAGI)),

· Calcul of projected intermediate K
K
R *=* ***Pz***
If proc 0
FETPRJ

1
O/N
OPTION=1
(K24IR1='&&FETI.VECNBI.AUX1'/

ZR (IR1)),

· Up to date Mise of the vector projected residue
If proc 0

DAXPY
K +1
K
K K
G
= ***G - R*** *(*
1
ZR (IRG)).
O/N

FETMPI
· Distribution of ***gk+1*** *to all proc.*
In parallel
O/O
MPI_BCAST
BLAS
· Calcul of the standard of the projected residue
O/O
DNRM2
K +1
G
(ANORM).

Test of stop if
K +1
G
< :
O/O

K

FETRIN
Buckle on the · Recalcul of the residue with the vector
P/P
OPTION=1 under-fields
of interface solution

concerned with
ground
R
= ***R K F R***

I (
I) + (
T
K 1
:
+
-
I
I
)

the proc. running,

BLAS DAXPY,
(K24IRR/ZR (IRR)),

FETREX,

RLTFR8,

End of loop

FETRIN
FETMPI
· Construction of the complete ***rsol*** *by In parallel*
OPTION=1

reduction towards the proc. 0,
O/N

MPI_REDUCE+
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

Code_Aster ®

Routine
Routines
Routines
Details
Information
appealing called
called
relative to
level 1
level 2
processors

MPI_SUM

Calculation of ground and rebuilding of the CHAM_NO

FETRIN
FETPRJ
solutions Master and slaves as in the cf test of stop of
OPTION=2
OPTION=2
test of stop of the table [Table 2-2].
2.b.

FETSCA
· Mise on a projected residue scale running
O/O

~k+1
+1
G
=
K
Ag
(K24IR1/ZR (IR1)).

FETPRC
Buckle on the · Calcul of packaged projected residue
P/P
under-fields
running
K +1
1
- *~ 1*
G
:
+
=
K
M G

concerned with
the proc. running,
(K24IR2= `&&FETI.VECNBI.AUX2'/
BLAS DAXPY,
ZR (IR2)).
FETREX,
MRMULT,
End of loop

FETPRC FETMPI
· Construction of the k+1
G
complete by
In parallel
O/N
reduction towards the proc. 0.
MPI_REDUCE+
MPI_SUM

FETSCA
· Mise on a projected residue scale

If proc. 0
~
O/N
packaged running K +1
+1
H
=
K
Ag

(K24IR3='&&FETI.VECNBI.AUX3'/
ZR (IR3)).

FETPRJ

.
K +1
~ +
H
=
K
PH
If proc. 0
OPTION=1
Calculation of projected running
1
O/N
(K24IRH /ZR (IRH)).

FETREO
BLAS DDOT, · up to date Mise of the direction of descent
If proc. 0
DAXPY,
current
pk+1
(ZR (IRP)) in
O/N
DCOPY.
réorthogonalisant, or not, by report/ratio

with the preceding directions,

· Calcul of the numerator of the parameter of

If proc. 0
descent current
O/N
K 1
+
K +1
K 1

: = ***G****. +*
p
(
NR
ALPHAN).

FETREO FETMPI
· Distribution of ZR (IRP) to all proc.
In parallel
O/O
MPI_BCAST
Fine buckles

Cleaning objects JEVEUX following option and

GCPPC
number of proc.
Table 2-3: Detailed flow chart and tree of call in ALFETI (level 2)
Data-processing handbook of Description
D9.03 booklet: -
HT-66/05/003/A

3
Installation of parallelism

First of all, algorithm FETI was coded into sequential then this establishment was adapted for to support a parallelism by sending of message in MPI-1. Indeed, the priority was to measure impact of such a solvor multidomaine on the architecture and the SD of the code, to limit them consequences (legibility, effectiveness, maintainability) and to secure of sound correct operation on standard cases. Moreover, for many authors, such a solvor appears often very effective (in CPU and in occupation memory), even into sequential, when one goes up in DDL (cf [bib2]). Then, the strategy of parallelization was as follows:

· To the main line operator calling solvor FETI (MECA_STATIQUE…), all them processors carries out the same sequence of operations and thus know the same objects JEVEUX: grid, materials, fields resulting from pretreatments, SD FETI… It is relatively sub-optimal, but taking into account the architecture of the code and its current use, it is the only possible option. It has however the merit not to impact the code sequential and, when the pretreatments are compared to the solvor, not very greedy in CPU and in memory, it is also often the strategy retained by the developers of codes parallels.
· Once in the main line operator, one will direct the operations carried out jointly by the processors by taring the volume of data which is affected for them. And this, of preparation of the data solvor, with numerical factorizations symbolic systems and, while passing by the assembly runs (of the matrix and contributions to the second members) and of course the algorithm of resolution itself. That will be done very simply, without sending of particular message, via the object "&&FETI.LISTE.SD.MPI" which will filter the loops on the under-fields:

CALL JEVEUO ("FETI.LISTE.SD.MPI", "It, ILIMPI)
C 50 I=1, NBSD <boucle ***on the*** *sous-domaines>*
IF (ZI (ILIMPI+I) .EQ.1) THEN
…. <on carries out the continuation of expected instructions that if it under-field is contained in the perimeter of the proc. courant>
ENDIF
50 CONTINUOUS

Concerning large usual objects JEVEUX, each proc. thus will build only them data which it needs: Main SD SOLVEUR and those slaves depending on the perimeter of processor running and the same thing for the NUME_DDL, the MATR_ASSE, the CHAM_NO. By against, the data of small volume, are they calculated by all the proc. because they often allow to direct calculation and it is of course important that all the procs. make *the same software advance.*

On the other hand, the NUME_DDL of each under-field being known only of its processor of reception, them
sendings of message are done with vectors of sizes homogeneous: the number of DDL of interface with run of the algorithm or that of the number of total DDL at the time of the final phase of rebuilding. The main processor manages the stages of reorthogonalisation and projection and their (potentially) large associated objects JEVEUX.

The cost of communication is roughly speaking:

***Initialization**: 3 MPI_REDUCE (size nbi (nbi are the number of Lagrange of interface, i.e. size of problem FETI to be solved)) + 4 MPI_BCAST (size nbi) + MPI_GATHERV*
***With each iteration of the GCPPC:** 2 MPI_REDUCE (size nbi) + 2 MPI_BCAST (size nbi)*
***Final rebuilding:** 2 MPI_REDUCE (size nbi) + 2 MPI_BCAST (size nbi) + MPI_REDUCE (size nbddl (nbddl is the total number of unknown (physics and late) of the problem))*

Rather than of the loops of communications points at points between the processors slaves and the Master
(MPI_SEND/RECV), one retained in a first approach of the collective communications (MPI_REDUCE…) who encapsulate the first and manage in manner transparent the problems of synchronization and of buffering. That ensures a better legibility, maintainability and portability but, has

contrario, one cannot optimize them by overlapping calculations and the communications, by limiting them
latency time or the buffering.
However, within sight of current software architecture, it seems that these optimizations are not if promising that that, because they are very dependent on the configuration machine, of that of network, of the chart network, implementation MPI and type of problem. Profits would be without rather doubt to seek the with dimensions one of a purely parallel implementation of the algorithm (without
vision proc. Master/proc. slaves) where the exchanges of messages would be limited to the neighbors of under-fields and on more reduced floods of data.

4 Bibliography

[1]
O. BOITEAU: Decomposition of field and parallelism in mechanics of the structures: State art and benchmark for an establishment reasoned in Code_Aster. Note intern EDF-R & D HI-23/03/009 (2003).
[2]
O. BOITEAU: Management report 2004 for UMR EDF-CNRS-LaMSID. Report intern EDF-R & D CR-I23/2005/006 (2005).

D9.05 booklet
D9.05.01 document
Implementation of STAT_NON_LINE

Summary:
One describes here implementation the data-processing of the algorithm of resolution of the problems quasi nonlinear statics. The document [R5.03.01] describing in detail this algorithm is supposed to be known, one will recall only the principal stages of them. One will find in this document a recall of the notations, the simplified flow chart of the routine op0070, allowing to distinguish the principal articulations logics of operator STAT_NON_LINE of Code_Aster, his tree of call, a description of data-processing objects and of the principal routines, and some traps to be avoided during the development in this operator.

EDF
Direction of the Studies and Research
Electricity of France
Project Codes of Mechanics

Contents

Error! No table of contents entry was found.

---

1 Notations and algorithm

One gives here a summary of the algorithm explained in detail in the document [R5.03.01] to be able to establish the link with the data-processing objects and operations.

1.1 Notations

One notes **U the** field of displacement and the multipliers of Lagrange associated with the conditions with the limits. They are the unknown factors of the problem. These fields are arranged in the same vector
data processing, mixed with the liking of the classification of the unknown factors. Thereafter, this vector will be noted
in manner symbolic system, without supposing about appearance of the unknown factors:

U

In the same way, requests applied to the structure (mechanical loadings and conditions to limits) are arranged in a vector loading, in an order which imports us little. One will note symbolically this vector:
Lméca

D
U
The length of these vectors is arranged in the majority of the routines in variable LONCH.
One must solve for a succession of moments of calculation *Ti* (often fictitious, allowing to parameterize it
loading), the following system:
T
T
méca
Q
+ =
I
B
I
Li
Drunk

=
D
I
ui
with the notations of the document [R5.03.01].
NB:
In the case of piloting, one has an additional unknown factor and an equation
additional (cf [§1.4]).
The algorithm is as follows: for each step of time, one carries out a prediction of Euler, then one correct using iterations of the method of Newton with linear research.
The diagram hereafter gives the notations used. One indicates by index *I* (like moment) the number step of time, and by exhibitor *N* (like newton) the number of the iteration of Newton.
increments are noted when they are measured compared to the state obtained with the last step of time (balance), and when they are measured compared to the state obtained with the last iteration of Newton.

One has as follows:

U = **-**
I
ui ui **-**1

= **-**
I
I
i-1

Lméca = *méca* - *méca*
I
Li
L *i-1*

ud = *D* - *D*
I
ui
ui-1
un+1 = *N* + *n+1* **-**
I
ui
ui
ui-1
n+1
un+1 = **u0** + the **U.K.**
I
I
I
k=1

un+1
I
linked
u1i
u0i
1
un+1
ui
I
U
u0
1
one
U
un+1
i-1
U
I
I
I
I
I
abstr. ultat
Pr édict ion
Newton
Newton
R és ultat
converged
I térat ion 1
it érati one n+1
conver gé
precedent
NB:
N +1
N
n+1
When linear research is made, **U** is written
=
+
I
ui
ui instead of
one +1 =

N + n+1
I
ui
ui
, the symbol thus does not have completely any more the same significance (it is in
this case direction of research).

1.2 Algorithm

One restricts oneself to give the systems of equations here composing the algorithm (without following loads and
without piloting): the logic of the algorithm is detailed in [§2]. The taking into account of the loads following is in [§1.3]. The complete equations (including piloting) are in [§1.4].

1.2.1 Phase of prediction

It consists in solving a linear system representing the tangent problem of speed
(cf [R5.03.01] and [R5.03.02]). That makes it possible to better converge in the later method of Newton
when one imposes anelastic deformations given (example: thermics, metallurgy).

No time running (other that the first)

0
T
0
méca
ther
K

+

=

+
i-1
ui
B
I
L *I*
Li

éq 1.2-1
B
u0 = *D*
I
ui
where **Ki-1** is the tangent matrix calculated in merimo via the option of calculation RIGI_MECA_TANG.
First step of time
K
u0 + **BT0** = **Lméca - QT** + **Lther**
0
1
1
1
0
1

éq 1.2-2
B
u0 = **ud - Drunk**
1
1
0
Recovery with boundary conditions of differential the Dirichlet type ("DIDI")
0
T
0
méca
T
ther
K
+
=
-
+

0
u1
B
1
L1
Q
0
L1
B
u0 =
D
1
u1

1.2.2

Iteration of Newton

N
n+1
T
n+1
méca
T
N
T
N
K

+

=
-
-

I
ui
B
I
Li
Q
I
B
I

éq 1.2-3
B

un+1 =
I
0
N
where **Ki** is the tangent matrix calculated in merimo via the option of calculation FULL_MECA.

1.2.3 Linear research

G ()
=
un+1
[] *T* **QT** *N* +un+1
N + *n+1*
méca
[
] =
I
(**ui**
I
) + **BT** (*I*
I
) - **Li**
0
p+1 = p - p - p-1 G p = p-1gp - PGP-1
gp - gp -1
G p - gp-1
One makes iterations on *p*, on the basis of 0 = 0 and 1 = 1.

1.2.4
Actualization of the unknown factors

n+1

N
n+1
U
=
+
I
ui
ui

idem for **U** and
n+1

= *N* + *n+1*
I
I
I
1.2.5
Tests of convergence
QTn + **BT** *N* **- Lméca**
I
I
I

RESI_GLOB_RELA:
1
Lméca + **Lther - BT** *N*
I
I
I

T
N
T
N
méca
RESI_GLOB_MAXI:
Q +
-

I
B
I
Li

2
un+1
I

INCO_GLOB_RELA:
3
un+1
I

ITER_GLOB_MAXI: if *N* > *nmax* one stops

1.3
Following loadings

These are loadings whose value depends on that of the unknown factors (for example the pressure in great displacements, cf [R3.03.04]). They thus should be revalued, not only with each step of time, but also with each iteration of Newton (i.e. with each change of the unknown factors). Moreover, they lead to a term of additional rigidity.

One breaks up the mechanical loading into a fixed part (nonfollowing) and a following part:

Lméca
=
fix +
I
Li
Lsuiv (**ui**, *Ti*)

The system to be solved is then:

QT
+ **BT** = **Lfixe** + **Lsuiv**
I
I
I
(**ui**, *Ti*)
Drunk

= **ud**
I
I

1.3.1
Phase of prediction

Lsuiv
(
K
-
) 0 +
0

= + *suiv fixes* + *ther*
i-1

U
BT
L
L
L
U
I
I
I
I
I

ui -1

0
D
B
U
=
I
ui
Data-processing handbook of Description
D9.05 booklet: Index A

1.3.2
Iteration of Newton
suiv
(
KN - L

) *n+1* +
n+1
=
fix +
N
N -
N
I

U
BT
L
Lsuiv (**U**) - **QT**
BT
U
I
I
I
I
I
I

linked
B
un+1 =
I
0

1.4
Controlled loadings

The amplitude of part of the loading to each step of time is controlled by an equation additional imposing the value of a degree of freedom. The loading (in the broad sense, including them boundary conditions), breaks up then into a part known as "constant" (even if it depends on time for example) of predetermined amplitude (given by the user) and of a "controlled" part known as whose amplitude *I* is an additional unknown factor of the problem:

Lcste

+

I
I **Lpilo**

.
ucste

+
I
I **U** *pilo*
The loading known as “constant” includes the possible following forces:
Lcste

= **Lfixe** + **Lsuiv**
I
I
I
ucste

= **U** *fixed*
I
I
In the same way the matrix of tangent rigidity, noted **K** will incluera the possible matrix of tangent rigidity
had with following forces.
The system to be solved is written:
QT
+ **BT** = **Lcste** + **Lpilo**
I
I
I
I
B
U
cste + **upilo**
I
= **ui**
I
D
(**ui**) = *I*

The third relation is scalar and *I* is worth the moment of calculation *Ti* in the current state of the code. For more
details, one will refer to the document [R5.03.01].

1.4.1 Phase of prediction

K

u0 + **BT** 0 - 0 **Lpilo** = **Lcste** + **Lther**
i-1
I

I
I
I
I

B
u0
0 **upilo** = **ucste**
I - *I*
I
P u0
- D
i-1
I

= *I*
(**ui-1**)
D is the operator of piloting (that which imposes the additional relation) and **P** its operator tangent. As for the moment one imposes the value of a degree of freedom, the operator **D** is linear and **P** = **D.**

---

cste

+ *ther*
-

() - 1 **Li**
Li

I
D (ui-1) - Ci-1 Have-1

cste

0 =
ui
I
pilo

C

() - 1 **L**
i-1 **A**

i-1
pilo
U

éq 1.4-1

0

cste
ther

U

+

I

=
L
0
With

() - 1 **Li**
I

+ **A**
() - 1 **Lpilo**

0

i-1

cste

I
i-1
U *pilo*

I
ui

BT

with **A**
= **Ki** -1

= (
)
i-1

and **Ci-1**
Pi-1 0.
B
0
1.4.2
Iteration of Newton
KN
n+1 +
n+1 - *n+1*
cste + *N*
N +
N
I
ui

BTi
I
Lpilo
= **Li**
I **Lpilo** - (**QT** *I*
BTi)

B
un+1
n+1
I

- *I* **upilo** = 0
N
N +1
N
P
= **-**
I **ui**
I
D (**ui**)
cste

N
N

+
N
-
Lpilo - (**QT**
BT)
D
N
N **Year**

() - 1 **Li**

- Cn Year
() - 1 *I*
I
I

I
(ui) - Ci
I

I
I

n+1 =
0
0
I

Cn Year

() - 1 **Lpilo**

I
I

U *pilo*

éq 1.4-2

one +1
cste

N
Lpilo -
N + **BTn**

I

=
(**QT**
)
Year
() - 1 **Li**

+ **Year**
() - 1 *I*
I
I

+ *n+1* **Year**
() - 1 **Lpilo**

n+1
I

I

I
I
U *pilo*

I
0
0

2

Simplified flow chart of STAT_NON_LINE
nmlect
reading of the operands
nminit
initializations and end of reading
initial state
not
virgin?
rebuilding

nmrepr
ther
yes
QT0 + **L**
initial loading

U
(
)
0
i-1, *I* -1

B U

0
known
nmacmv
actualization of the matrices and vectors
0
0
phase of prediction **U**
(
)
I, *I*
nmpred
R C
)
K u0 + **BT0** = **Lméca** + **Lther**
(
T I
i-1
I

I
I
I

L
0
D
=
one
N
(
B
U = **U**
I, *I*)
I
I
I
L
known
n+1
n+1
iteration of Newton
(
)
:
ui, *I*
nmnewt
R C
T I
KN un+1 + **BTn+1** = **Lméca - QTn - BTn**
I
I
I
I
I
I

B
un+1 =
I
0
(exhibitor N)
(index I)

nmrech
R L
iterations of linear research
n+1
n+1
actualizations of the unknown factors **U**
(
)
I
, *I*
majour
un+1 = **one** + **un+1**
I
I
I
N
+1 = *N* + *n+1*
I
I
I
nmcrar
tests of convergence
Iterations of Newton
QTn + **BT** *N* **- Lméca**
un+1
I
I
I

I

méca + *ther* -
N
1
n+1
3
not
L
L
BT
U
I
I

I
I
convergence?

Buckle on the "steps of time"
QTn + **BT** *N* **- Lméca**

N > *N*
I
I
I

2
max?
yes
actualizations and filing **U**
(
)
I, *I*, *I*, *I*
nmstoc
U = **un+1**
= *n+1*
I
I
,
I
I

= *N*
= *N*
I
I,
I
I
Data-processing handbook of Description
D9.05 booklet: Index A

The loops with symbol RC indicate the expression of the relation of behavior (through routine merimo, cf [§5.10]).

The loop with symbol RL indicates the iterations of linear research (cf [§5.5]).

One gave in the flow chart of the simplified equations which do not take account of following loadings nor of the controlled loadings.

3 Tree of call of the op0070

One gives here the principal routines called by the op0070 like their function. One does not speak calls with the routines supervisor, the routines of creation of objects JEVEUX, assembly of vectors and of matrices, impression of messages. A more detailed description of the routines principal is given in [§5].

op0070
nmlect
beginning of the reading of the command file (cf key words)
nmdome
model, material, loading
(MODEL, CHAM_MATER, CARA_ELEM, EXCIT)
nmdorc
relation of behavior (COMP_ELAS, COMP_INCR)
nmdomt
parameters of the method of resolution (NEWTON)
cresol
parameters of solvor (SOLVEUR)
nmdocn
criteria of convergence (CONVERGENCE)
nmdoop
options of calculation (OPTION)
nminit
initializations and end of the reading of the command file
number
creation of classification - profile of the matrix
nmdoet
seizure of the initial state (ETAT_INIT)
nminst
calculation of the initial moment
nmdoin
list moments of calculation

nminin
initial moment
nmpogd
initializations for beams in great rotations
nmdopi
data acquisition of piloting (PILOTING)
medime
elementary matrices of rigidity (contributing to the matrix **B)**
associated the boundary conditions
nmdete
recovery of the initial field of temperature
nmrepr
taking into account of an initial state not no one
vefnme
calculation of the initial nodal forces (**QT0**)
vebume
calculation of displacements imposed initially (**B u0**)
vectme
ther
calculation of the initial thermal loading (**L** 0)
vaczme
ther
modification due to the metallurgy (**L0**)
veanme
modification due to anelastic deformations given
ther
(**L0**)
nmitin
recovery of the moment of current calculation *Ti*
Data-processing handbook of Description
D9.05 booklet: Index A

Page: 11/22
nmacmv
actualization of the matrices and vectors for new T
nmrema
re-assembly of the tangent matrix? (Boolean REASMA)
merimo
calculation of the tangent matrix and the nodal forces
mecgme
calculation of the tangent matrix of the following loadings
preres
factorization or calculation of the preconditionnor
vedime
vector of not controlled imposed displacements
vechme
vector of the nonfollowing mechanical loads not controlled
vedpme
vector of controlled imposed displacements
vefpme
vector of the controlled nonfollowing mechanical loads
vecgme
vector of the following mechanical loads
vectme
vector of the thermal loading
vaczme
modification of the thermal loading due to the metallurgy
veanme
modification of the thermal loading due to the deformations
anelastic data
nmpred
preliminary phase of prediction
vatzme
increment of thermal load due to the metallurgy
nmreso
resolution of system linear
nmmico
treatment of the unilateral contact
nmnewt
treatment of an iteration of Newton
vecgme
vector of the following mechanical loads
nmrema
re-assembly of the tangent matrix? (Boolean REASMA)
merimo

calculation of the tangent matrix and the nodal forces
mecgme
calculation of the tangent matrix of the following loadings
preres
factorization or calculation of the preconditionnor
nmreso
resolution of system linear
nmrech
linear research
majour
actualization of the unknown factors
merimo
calculation of the nodal forces
nmcrrl
test of stop of linear research
majour
actualization of the unknown factors
nmcrar
test of stop of the iterations of Newton
nmstoc
actualizations and filing of the results
ndarch
filing of the results of the step of time
Data-processing handbook of Description
D9.05 booklet: Index A

4
Contents of the principal handled variables

4.1
Loadings

The total mechanical loading breaks up in the following way:

méca
fixed
pilo
L

suiv

L

I

= **Li**

+ **Li**

+
I

D

fixed
I

pilo
U

U

U

I
I
0
I
variable
address
contents
description
routine
vectors
JEVEUX
creative
elementary
CNCHMP
JCHMP
fixed
loadings
vechme
VECHMP
L
I

of Neumann
0
not not controlled follower
CNFPIP
JFPIP
pilo
loadings
vefpme
VEFPIP
L
I

of Neumann
0
not controlled follower
CNCGMP
JCGMP
L
suiv
loadings
vecgme
VECGMP

I

of Neumann
0
not controlled follower
CNDIRP
JDIRP
0

loadings
vedime
VEDIRP
fixed
of Dirichlet
U

I
not controlled
CNDPIP
JDPIP
0

loadings
vedpme
VEDPIP
pilo
of Dirichlet

U

I
controlled
CNCHTP
JCHTP
thermal loading
vectme
VECHTP
Lther
I

+ metallurgy
vaczme
0
+ déf. anelastic
veanme
CHFIXE
Lfixe

+ **Lpilo**
total of the loadings of
I
I
I

Neumann not follower

0

CHCSTE
JCHCS
Lfixe

+ **Lsuiv** + **Lther**
total of the loadings
I
I
I

not controlled

U *fixed*

I
CHPILO
JCHPI
Lpilo

total of the loadings
I

controlled
U *pilo*

I
4.2
Unknown factors, reactions of support and forces nodal
The notations in increments are pointed out:

un+1 = *n+1* **-**
I
ui
ui-1

un+1 = *n+1* **-** *N*
I
ui
ui
Caution:
Data-processing handbook of Description
D9.05 booklet: Index A

The contents of the objects (in particular unknown factors) are variable during routines: one give in [§5] the detail of the objects according to crossed routines'. In the following table one gives the significance **planned** for these objects.

For example, object DDEPLA is intended to collect the increment of the unknown factors between two iterations of Newton. However, in nmpred one stores there temporarily the second member linear system to solve, then the solution of this system, which is the increment enters preceding balance and the prediction.

object
address
contents
description
JEVEUX
DEPMOI
JDEPM
U

displacement and Lagrange
i-1

with preceding balance

i-1
DEPPLU
JDEPP
one

displacement and Lagrange
I

currents
N

I
DEPDEL
JDEPDE
N

increment of the unknown factors
ui

since balance
N

precedent
I
DEPDET
JDEPDT
N
n+1

auxiliary vector for
U + *p*
I
ui

linear research
N

+ *pn+1*
I
I
DDEPLA
JDDEPL
N

increment of the unknown factors
ui

since reiterated Newton
N

precedent
I
CNRESI

JCNRE
nodal forces
QT
N +
N
I
BTi

(in the broad sense, including them

0

reactions of support)
CNDIRI
JDIRI
reactions of support
BT
nor

0
SIGMOI

constraints
i-1
with preceding balance
SIGPLU
N
constraints
I
current
VARMOI

internal variables
I -1
with preceding balance
VARMOJ
n-1
internal variables
I -1
with reiterated preceding
VARPLU

N
internal variables
I -1
current
TEMMOI
T
field of temperature
I -1
with preceding balance
TEMPLU
T
field of temperature
I
running
NB:
One does a distinction between the values of preceding balance (i.e. out of *Ti* -1) and them
values of reiterated preceding, i.e. those which one obtains during iterations of
Newton and which is not a priori in balance.

---

5 Detail of the principal routines

5.1 Routine nmrepr

This routine makes it possible to treat to take into account a nonvirgin initial state (case of the recoveries in particular). One supposes known (read under key word ETAT_INIT in the routine nmdoet) the field **u0** displacements and constraints 0. In this case the Boolean NEWCAL is worth .FALSE.

Principal routines called:

Q
T

.
0
vefnme: calculation of
, initial nodal forces; calculation will be carried out on
0

initial deformed geometry (taking into account of **u0**) if one indicated
GEOMETRY: “DEFORMED” under key word ETAT_INIT;
0

.
vebume: calculation of **Drunk**

, initial imposed displacements;
0
Lther

.
0

vectme, vaczme, veanme: calculation of

, initial loading due to thermics, with
0
metallurgy and with the imposed anelastic deformations.
One stores the initial loading (rebuilt) in CHCSTE:
ther

+ *T*
Q 0
CHCSTE = **L** 0

Drunk

0
If NEWCAL=.true., the initial state is virgin and CHCSTE is worth 0.
5.2

Routine nmacmv
It brings up to date the matrices and vectors at the beginning again of step of time (in the operator THER_MECA_NOLI, this routine are also called within a step of time for each new iteration of thermomechanical coupling).
Principal routines called:
·
merimo: calculation of the tangent matrix if OPTION=' RIGI_MECA_TANG', if not nothing.
·
mecgme: calculation of the tangent matrix associated the following loadings.
Data-processing handbook of Description
D9.05 booklet: Index A

NB:
The elementary matrices resulting from merimo (object MERIGI) and mecgme (object MESUIV) are assemblies with the matrices relating to Lagrange (calculated once and for all by medime at the beginning of the op0070 and stored in object MEDIRI) in a large matrix (object MATASS), that one can symbolically note:

- Lsuiv
BT
With
= **Ki**
-1

i-1
U

U

I -1

B
0
.
preres: factorization of the tangent matrix (if solvor LDLT or MULT_FRONT) or calculation of preconditionnor (if solvor GCPC),
.
vedime, vechme, vedpme, vefpme, vecgme, vectme, vaczme, veanme: actualizations of vectors representing the various contributions to the loading (cf table of [§ 4.1]).

5.3
Routine nmpred

0
U
This routine carries out the calculation of the predictor
I

, while solving in the case general (with piloting)
0

I
the system [éq 1.4-1].
U

One affects to
i-1
DEPPLU the values obtained with preceding balance, is

.

i-1
One stores in DDEPLA the increment of loading not controlled (mechanical + “thermal”):
DDEPLA =
CNCHMP + CNCGMP + CNDIRP + CNCHTP - CHCSTE (current-precedent)

Lfixe + **Lsuiv** + *ther*
I
L *I*
that is to say
I

DDEPLA =

U *fixed*

I
The routine vatzme supplements this second member by the taking into account of the increment of load due to
the plasticity of transformation (metallurgy).
NB:
T
ther
Q
+ **L**
With the first step of time,
0
0
CHCSTE contains

thus DDEPLA will be worth:

Drunk

0
fixed

+ **Lsuiv -**
+ *ther*
1
QT0
L1

DDEPLA = **L1**
(cf [éq 1.2-2]).

U *fixed* - **Drunk**

1
0
One brings up to date the loading not controlled:
CHCSTE = CNCHMP + CNCGMP + CNDIRP + CNCHTP

(running)
fixed

+ **Lméca** + *ther*
I
Li
that is to say

CHCSTE = **L** *I*

U *fixed*

I
Data-processing handbook of Description
D9.05 booklet: Index A

The controlled loading is brought up to date:
CHPILO = CNFPIP + CNDPIP
(running)
pilo

that is to say

CHPILO =
Li
upilo

I
One solves the linear system [éq 1.4-1] by calling nmreso (cf [§5.9]) with in arguments CHPILO and DDEPLA (second member CHRESU in entry, solution at exit).
NB:
DEPDEL is given to zero in the routine initiated.
After nmpred in the op0070, one brings up to date DEPPLU and DEPDEL (routine majour):

+ **u0i**
DEPPLU = DEPPLU + DDEPLA = **ui-1**

+ 0
i-1
I
0

DEPDEL = DEPDEL + DDEPLA = 0 + **ui**

0
+ 0

I
5.4
Routine nmnewt
One brings up to date the second associate member with following loadings CNCGMP.
Call of merimo: one calculates the nodal forces (if OPTION=' RAPH_MECA') or the nodal forces and the tangent matrix (if OPTION=' FULL_MECA'). If there is re-assembly (REASMA=.TRUE.), one calculates
the tangent matrix associated the following loadings, then one reassembles and one factorizes (preres)
N
the noted total tangent matrix **Have**. All this is identical to the operations carried out in nmacmv (only difference: the option of calculation of the tangent matrix is “FULL_MECA” and not “RIGI_MECA_TANG”, cf [R5.03.01] and [R5.03.02]).
One assembles the nodal forces and the reactions of support relating to reiterated preceding, respectively in CNRESI and CNDIRI.
One stores in DDEPLA the second member of the linear system to solve:
DDEPLA =
CNCHMP + CNCGMP + ETA*CNFPIP - CNRESI
fixed

+ **Lsuiv** + *N* **Lpilo -**
N + **BTn**
I
I
I
(**QTi**
I)

DDEPLA = **Li**

0

One solves the linear system [éq 1.4-2] by calling nmreso (cf [§5.9]) with in arguments CHPILO (unchanged) and DDEPLA (second member in entry, solution at exit).

5.5
Routine nmrech

This routine calculates the scalar (RHO) which will be the step of advance in the direction given by DDEPLA.

NB:
For the moment, linear research is called only if ETA = 0 (not of piloting).
The vectors separately are sent: CNCHMP, CNCGMP, CNFPIP.

Linear research consists in cancelling the product:
G = DDEPLA * [CNRESI-CNCHMP-CNCGMP-ETA*CNFPIP]
by using a method of secant.

One makes a loop on an index *p*, on the basis of 0 = 0 and 1 = 1. Knowing *p-1* (stored in RHO0), *G p-1* (stored in F0) and *p* (stored in RHO):

·

calculation for *p* of DEPDEL+RHO*DDEPLA stored in DEPDET (routine majour):

N

n+

U + *p*

1

I

ui

·

calculation of nodal forces CNRESI (option “RAPH_MECA”) correspondent with DEPDET (routine merimo)

·

calculation of DDEPLA* [CNRESI-CNCHMP-CNCGMP-ETA*CNFPIP], is:

N +1

T

N

n+1

T

N

n+1

méca

G (*p*) =

[**U**] *T* [+p

+ *p*

]

I

Q

(**ui**

ui

) + **B** (*I*

I

) - **Li**

stored in F1

·

checking of the criterion of stop of linear research (routine nmcrrl): value of CNRESI-CNCHMP-CNCGMP-ETA*CNFPIP lower than RESI_LINE_RELA, or *p* higher than ITER_LINE_MAXI

·

calculation of new RHO (auxiliary variable RHOT of transfer):

- *p*

-

p+1 =

G p-1 + *p-1gp* = RHO*F0 + RHO0* F1

gp - *gp* -1
F1 - F0
After nmrech in the op0070, one brings up to date DEPPLU and DEPDEL (routine majour):
N
+ **un+1**
I

DEPPLU = DEPPLU + RHO * DDEPLA = **ui**

N
+ *n+1*
I
I

one + **un+1**
I
I
DEPDEL = DEPDEL + RHO * DDEPLA =

N

+ *n+1*

I
I
Data-processing handbook of Description
D9.05 booklet: Index A

5.6

Routine nmcrar
This routine tests the various criteria of stop of the iterations of Newton.
One separately sends the vectors representative of the loading to him:
CHFIXE = CNCHMP + ETA*CHFPIP
CNCGMP
CNCHTP
CNDIRI
One calculates the quantities specified with [the § 1.2.5] and one compares with the limiting values fixed by
the user. One prints the values of the various criteria and the iteration count of research
linear. The checked criteria are affublés of one * beside their value. Convergence is issued
if all the specified criteria are checked. The op0070 is thus informed that CONVER=.TRUE.
NB:
Before looking at convergence, one looks at if ITERAT *N*
() is strictly higher than
value given under ITER_GLOB_MAXI, and if it is the case one stops. That wants to say in it
case which one carried out an useless iteration of Newton (since one does not test
convergence): to correct in a forthcoming version.
5.7
Routine nmstoc
This routine carries out actualizations necessary to the end of the step of time (swing of
values XXXPLU and XXXMOI: internal displacements, constraints, variables, temperatures, ETA) thus
that filings (routine ndarch) in the structure of data evol_noli results.
The stored solution is:
n+1
n+1
U
=
=
I
ui, *I*
I

= *N*
= *N*
I
I,
I
I
It is necessary to bring up to date the loading (which will be taken as loading of the step of previous time in
next nmpred). One needs to recompute CHCSTE since the following forces changed since
nmpred (in nmnewt and nmrech):

CHCSTE = CNCHMP + CNCGMP (new) + CNDIRP + CNCHTP
fixed

+ **Lsuiv** + *ther*
I
Li
CHCSTE = **L** *I*

U *fixed*

I
Vector CHPILO is not used to calculate increments and is built when one needs some, one does not have thus not to update it. Idem for the vector CHFIXE which is useful only in the criterion of stop.
NB:
One could store CHFIXE and CHPILO in the same data-processing object since one them recompute with each time one needs some. But it is preferable for the clearness of the op0070 to have these two objects.
5.8
Routine merimo
It expresses the relation of behavior:

N
N
.
calculation of constraints *I* (SIGPLU) and the variables intern *I* (VARPLU) from
values *i-1* (SIGMOI) and *i-1* (VARMOI) with preceding balance (out of *Ti-1*) and of the increment

N
N
of displacement **U** =
-
I
ui
ui-1 since this balance (DEPDEL);
N
N
.
calculation of the "nodal forces" (forces intern + reactions of support): **Q** +
I
BTi (VERESI)
N
and of the reactions of support **BT** *I* (VEDIRI), which one assembles then;
N
.
possible calculation of the matrix of tangent stiffness: **Ki** (MERIGI).
For memory one gives here the sequence of the routines which, on the basis of merimo, treats the relation of
behavior VMIS_ISOT_TRAC for the isoparametric elements 2D:

- merimo: filling of the exit and inlet limits for calculation, call of calculation for
to calculate the nodal forces, call of calculation to calculate the reactions of support
- calculation: buckle on the elements
- te0100: for each element, shunting of the options of calculation (FULL_MECA,…) and call of nmpl2d
- nmpl2d: buckle on the points of Gauss, calculation of the geometric standards, of
deformations, call of nmcomp, filling of the elementary matrix of rigidity and of
elementary vector of the nodal forces
- nmcomp: shunting of the relations of behavior
- nmisot: calculation of the constraints, variables internal, and derived from the constraints by report/ratio with the deformations for the relation of behavior VMIS_ISOT_TRAC

5.9
Routine nmreso

It is pointed out that:

cste
fixed
suiv
L

=
+
I
Li
Li
ucste

= *fixed*
I
ui

Lsuiv

With =
K -

BT
U

B
0
C = **P**
(0)

In entry of nmreso coming from nmpred:

In entry of nmreso coming from nmnewt:
pilo

pilo

CHPILO = **Li**

CHPILO =
Li

upilo

upilo

I
I
cste
ther
L +
cste

+ *N* **Lpilo - (QTn** + **BTn**)
I
L *I*
I

I
I
I
CHRESU =

CHRESU =
Li

cste
U

0

I
N
MATASS = **A** = **Have** -1
MATASS = **A** = **Have**
N
PILOT **C** = **C**
PILOT **C** = **C**
i-1
I
N
DEPPLU = **U**
DEPPLU = **U**

i-1

I
In nmreso: one solves [éq 1.4-1] or [éq 1.4-2]:
-1
CHSOL1 = **A** .CHPILO
-1
CHSOL2 = **A** .CHRESU
= **D** (**U**)
RD = **C.CHSOL1**
RN = **C.CHSOL2**
n+1
DETA =
= BETA - - RN
I
RD

u0
n+1

I
ui
CHRESU =

or

= DETA.CHSOL1 + CHSOL2
0

n+1
I
I
n+1
N
n+1
ETA =
= +
= ETA + DETA

I
I
I
Data-processing handbook of Description
D9.05 booklet: Index A

---

6 Traps and easy ways

In this chapter one even gives in disorder a certain number of important information (essential) for which wants to develop in the op0070.

- VERESI, RESIDUE, CNRESI,…: they are the nodal forces, i.e. the forces intern + them reactions of support. As opposed to what could let think the name of the variable, **it is not not a residue**: that thus does not tend towards 0 when one converges (that tends towards the loading outside)!!!
- the calculation of the elementary matrices associated Lagrange is made once and for all at the beginning op0070 (call of medime): indeed, even if the value of Dirichlet imposed changes with run from time, that modifies the second member **ud** and not the matrix **B**.

·

the thermal loading intervenes only by its (except for the criterion of stop): nevertheless one always calculate its value at the previous moment and at the current moment and one takes the difference of it
(to take account of the possible variation of the dilation coefficient according to temperature).

·

object CNCHTP (thermal loading) is called a little differently other loadings. Indeed, for the mechanical loadings (CNCHMP for example), the object is a list of assembled vectors of type VACHMP. Since only one load can be thermal, the object CNCHTP is in fact the equivalent of only one VACHMP: his address JTP is taken, and the values are with the address of object ZK24 (JTP), that is to say JCHTP.

·

to limit the number of arguments of the routines, one gathered certain objects in a table giving their names:

-
VALMOI contains to the maximum DEPMOI, SIGMOI, TEMMOI, VARMOI, and VARMOJ

-
VALPLU contains to the maximum DEPPLU, SIGPLU, TEMPLU, and VARPLU

-
SECMBR contains to the maximum CNCHMP, CNFPIP, CNCGMP, CNDIRP, CNDPIP, CNCHTP, CNRESI, and CNDIRI

-
POUGD contains vectors relating to calculations of beams in great rotations

The “filling” and “emptying” of these tables of character strings are made by the intermediary routines agglom and desagg.

·

the treatment of the contact consists in calling a routine nmmico after nmpred and nmnewt, which correct the field of displacement, and to add the forces of contact obtained (stored with address JATMU) with nodal forces CNRESI and the reactions of support CNDIRI.

·

for a reason not included/understood well of the author of this document, the call to the routine nmreso invalid the address of object DDEPLA: it thus should be resaisir by a call to jeveuo after nmpred and nmnewt.

·

in the routines nmxxx all inevitably did not pass in argument: for example, MEDIRI in nmacmv is known by one DATED and used thus.

·

attention with the handling of DATED: the substitution of the names is made only during compilation. It is preferable when one wants to impose fixed names for objects calculated on several recoveries, to declare them like instruction at the beginning of routine (cf nmacmv).

Data-processing handbook of Description
D9.05 booklet: Index A

·
if an elementary vector is stored in a variable of name &&VECHME…, the vector assembled is stored in a variable of name &&VACH00… Only discriminating characters for the names of vectors are thus 1st, 3rd and 4th. Thus two vectors &&VECHME and &&VECHTP would lead to an ambiguity and thus bugs. Therefore they were used names &&VEMCHA… (for VECHMP), &&VEMSUI… (for VECGMP), &&VEMTPP… (for VECHTP), etc (cf nmacmv for example). Since the creation of operator THER_MECA_NOLI, it is also necessary to pay attention to avoid ambiguities between the thermal vectors (flow, imposed temperatures…) and mechanical vectors.
·
the routines initiated and majour refers partially to the beams in great rotations.
·
the test of convergence is made after the first iteration of Newton (one thus makes some always with less one): thus for an elastic design where one with the solution at the end of the phase of prediction, one makes an iteration of Newton for nothing…

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D9.05 booklet: -
Document: D9.05.03

New architecture THM. Integration of
equilibrium equations

Summary:

In order to allow the development of rather general nonlinear laws of behavior in the module THM of Code_Aster, it appeared necessary to completely separate the equilibrium equations and the relations of behavior.

This document defines the principles of this separation, and described the specifications of under program credits EQUTHM, integrating one or more equilibrium equations and calling the laws of behavior. One supposes that the medium can be made up with more than one solid and of two components, each of these two components being able to exist under two phases. Each one of these elements can or not exist, it thermal can to be taken into account or not. The thermal equation not takes into account a formulation in entropy, but in energy utilizing mass enthali of the components.

Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
7.4

Titrate:
New architecture THM. Integration of the equilibrium equations Dates
:

22/06/05
Author (S):
C. CHAVANT Key
:
D9.05.03-A Page
: 2/36

Count

matters

Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
7.4

Titrate:
New architecture THM. Integration of the equilibrium equations Dates
:

22/06/05
Author (S):
C. CHAVANT Key
:

D9.05.03-A Page
: 3/36

1
Variational writings of the equilibrium equations

1.1 Mechanics

One leaves the following differential writing:

Div
R m
+ F = 0

éq 1.1-1

We will further see we always adopt the decomposition = + I, where indicates
p
the effective constraint.

It is thus with the load of the module of integration of the equilibrium equations to make the sum:
= + I.
p
One will then write a variational form of [éq 1.1-1] at time T +.

+

= +
+ +

I

p

éq
1.1-2
+
+

m
ext.

+.

(v) = +rF .v +

F
v
v U

AD

1.2 Hydraulics

One leaves the following differential writing:
DM + Div (M) = 0 éq
1.2-1
dt
It is considered that there can be two components, and for each one D `them two phases.

More precisely, the variables m, M and m, M refer each one to a component of mass
1
1
2
2
conservative.

One poses by principle:
m = m1 + m2
1
2
;
M = M + M
1
1
1
1
1
1
m = m1 + m2
1
2
;

M = M + M
2
2
2
2
2
2

What we will write:
m
= mphase
component
component
Nb phasedu
component

phase
M
= M
component
component
Nb phasedu
component

In the applications, one could for example have:

2 components: air and water
2 phases for water
1 phase for the air
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

:
D9.05.03-A Page
: 4/36

One would have then:
m1 and
1
***M**: contribution of mass and liquid water flow*
1
1

m2 and
2
***M**: contribution of mass and vapor flow*
1
1

m1 and
1
***M**: contribution of mass and flow of dry air*
2
2

m2 and
2
***M**: non-existent*
2
2

It is considered that there are two pressures. No assumption is made on what the pressures mean p and p, that will depend on the laws of behavior and the way which one will choose to write them: one
1
2
could for example choose:

p = pressioncapillaire (p (gas) - p (liquid))
1

p = pressionde Z
ga (vapor + air)
2

One will write then a variational form of [éq 1.2-1].

D (m1
2
1 + m1)
-
1
2
1
2

M
Mr.
M
M
.

éq
1.2-2
1
(1
1)
1
(1ext
1 ext.)
P

+
+
=
+
1
1
dt
1ad

D (m1 + m2
2
2)
-
2 + (1
2
***M2** + **m2**). 2 =*
(1
2

M
+ ***M***
.
P
2ext
2ext)
2
2
2ad ***éq***
1.2-3

dt

After discretization by a teta method:
-

M
M
.
(1+
m
2
1
2

1
+ +
m1) 1 + T (+1 + +1) 1 =
-

M
M
.
éq
1.2-4
(1-
m
2
1
2

1

1
+ -
m1) 1 - (-) T (-1 + -1) 1
+ T

M
M
.
(
1
2

1
+ 1) 1
1 P
ext.
ext.
has
1 D
-

M
M
.
(1+
m
2
1
2

2
+ +
m2) 2 + T (+2 + +2) 2 =
-

M
M
.
éq
1.2-5
(1-

m
2
1
2

1

2
+ -
m2) 2 - (-) T (-2 + -2) 2
+ T

M
M
.
(
1
2

2
+ 2) 2
2 P
ext.
ext.
2ad

1.3 Thermics

We introduce the enthali of each phase of each component: H p
C m

We note: Np the number of phases of the component C.
C

We adopt the rule of summation of the dumb indices:

npc
dmp
npc
dmi
HP
p
M =
hi

I
M

HP
C =
hi
C

C m
C
cm
C
C m
dt
C m
I 1
dt
I =1
=

The equation of thermics (or energy) is written:

dQ'
dmp
+ HP

C + Div p
p
p
m
M + Q = + M. F
éq
1.3-1
C m
(hc m c) R
dt
dt
C

One will then write a variational form of [éq 1.3-1] without injecting the hydraulic equilibrium equation there:

dQ'
dmp

HP
C
HP
p
M
Q.
R
p
M. F
HP
p
M
Q
.

+
dt
C m

-
+
=

+
-
+
dt
(C m c)
(
C
) (C m cext ext.)

Tad
éq 1.3-2
The discretization of [éq 1.3-2] by teta method leads to:

M
Q
1

M
Q
(Q'+ - Q'-) - T (H p+
p+
+
p
p
-
+
- -
+
+...
C m
C
)
(
) T (hc m C
)
+ hp+ p+
p

p
p+
p
-
+ 1-
-
-
=
C

m

(m m
C m
C m)
(
) hc m

(m m
C m
C m)
+ T

p+
m
p
m

p
p
M. F + 1 -
M. F +
-

M
+
Q
. T
C
(
) T

T
C
R
T

(hc m cext ext.)
AD
éq 1.3-3
One notices in the equation [éq 1.3-3] a term of contribution of heat by the flow of fluid at the edge of

field:
+
. .
(H p
p
M
Q
C m
C ext.
ext.)

One will be able in makes consider that the conditions of heat flux define directly:

~q
p
H M p
Q
=
+

ext.
C m
C ext.
ext.

Titrate:

2 Laws of behavior

2.1 Mechanics

2.1.1 Writing general

+
= +
(+ + + + - - - - - -
, p, p, T;
1
2
, p, p, T,
1
2
,)

éq
2.1.1-1

+
= +
(+ + + + - - - - - -
, p, p, T;
1
2
, p, p, T,
1
2

,)

2.1.2 Case of the effective constraints

In the case of the assumption of the effective constraints, this function will break up in the form:

= + pi

tensor

is
constraint

effective

S
:

scalar

one

is
p
+ = + + + - -
-
-

(, T; , T,)

éq
2.1.2-1
+
= + + + - -
-
-

(, T; , T,)
+

+

=

p+, p+; p, p

-

,

p

p (1 2 1 2 H)

éq

2.1.2-2

+

+

=

p+, p+; p, p

-

,

H

H

(1 2 1 2 H)

It is noticed that in this decomposition:

***· the dependence compared to thermics was left in the effective constraints;
typically, it is thought that the laws on the effective constraints are written as in thermo
traditional mechanics:***

+ = +(+ - + + -

T;

- - -

T, -

-

,)

***· one distinguished the internal variables relating to the law from behavior on the constraints
effective, that one wrote, internal variables of origin hydraulics which one has
written and internal variables of thermal origin which one wrote (see***

H
T
following paragraphs).

2.1.3 Choice of the constraints

Because of rather frequent use of the assumption of the effective constraints, one decides that it vector of the constraints for the mechanical part contains in all the cases the tensor of the constraints effective and the scalar p. In the case general where the assumption of the effective constraints is not not retained, one will have simply: p = 0.

It is thus with the load of the module of integration of the equilibrium equations to make the sum:
= + I.
p
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

2.2 Hydraulics

The hydraulic law of behavior will provide the following relations:

mp+ = mp+ + + + + - - - - Q Q
,
,
,
; ,
,
,

,
, M, _

1
2

1
2

C
C

(p p T p p T Mandelevium D H)

+

, p+,
1
1 p+, p+,
1
2 p+, T +,
2
T+;
it
T pde with npc
p+
p+
M = M

éq 2.2-1
C
C -

, p, p
M
F
1
, p,
1
2 p, T -,
2
T
Q -

_
m+
,
, ;

D
H

+
+
+
, p+, p+, T+ -
;
1
2
, p, p, T, m, m
=
,
1
2
1
2 _
H
H

(
H)

It is noticed that the field of gravity is a data of the hydraulic law of behavior by it that the evolution of the vector of flow follows relations of the type: M
=
fl
fl
m
P
.
H
[- + F]

2.3 Thermics

The laws of behavior will provide:

Q'+ = Q'+ (+
, p+, p+, T+; -
1
2
, p, p, T, S -
1
2
)

hp+

= hp+ + + + + - - - - Q
, , , ;
1
1
2
, , , , -
1
2

C m
C m (
p p T
p p T sd m)
this pde with npc
+
+
Q = Q (+
, p+, p+, T+, T+
Q
1
2
; -
, p, p, T,
1
2
T, -)

éq 2.3-1
+
+
+

=

(, p+, p+, T+, T+ -
; , p, p, T,
1
2
T
-
, T)
T
T
1
2
With hq-
1-
2-
1-
2-
=
,
,
,
D m
(H H H H
1 m
1 m
2m
2 m)

2.4
homogenized density

r+
R
m1+ m2+ m1+ m2+
= 0 + 1 + 1 + 2 + 2
éq
2.4-1

3 Efforts
generalized

It arises of what is written higher than the generalized constraints are:

, ;
p
m1, 1
M, h1; m2, 2
M, H2;
1
1
1 m
1
1
1 m

m1, 1
M, h1; m2, 2
M, H2;
2
2
2m
2
2
2 m
Q', Q

The associated generalized deformations are:

U, (U); p, p; p, p; T, T
1
1
2
2

Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

Code_Aster ®
Version
7.4

Titrate:
New architecture THM. Integration of the equilibrium equations Dates
:

4
Algorithm of resolution

4.1
Nonlinear algorithm of resolution of the equilibrium equations

In the case general of modeling (variable coefficients, desaturation, convection) the problem variational presented above is nonlinear compared to the fields of displacement, pressure and temperature. After discretization by finite elements, one obtains a nonlinear matric system. stamp resolution contains moreover one nonsymmetrical term and is treated like such (not symmetrization of this matrix to use methods of minimum). One uses in all the cases of modeling the nonlinear solvor of Code_Aster STAT_NON_LINE resting on a method of Newton-Raphson, described in [R5.03.01]. Its principle is as follows (the equations corresponding to treatment by dualisation of the boundary conditions are not indicated explicitly here).

The equilibrium equation thermo-poro-mechanics at the moment T +, knowing at the previous moment
(U, -

P, -

T), as well as the possible internal variables is written:

F (U, P
, T
+
=
-
+
+
+)
L (T) G (U, P
, T
,
I
E

-
-
-)

To find the solution of this nonlinear equation, a continuation is built:

· initialized by a prediction which gives (U, P, T) = (U, P, T
- - -) + (U
0
0
0
0, 0
P, 0
T):

DF
+
-
(
· U
,

0
,
0
0 =
-
U
-
-)
, ,
(
P
T) L (T) L (T
I
P
T
E
E
)

· corrected by recurrence giving (U, P, T
, ,

,
,
:
1
1
1 =
+
+
+
+)
(U P T) (U P T
N
N
N
N
N
N
n+1
n+1
n+1)

DF · (U,
,
, ,
, ,

1 P 1 T
+
+
+
+1) = - F (U
P T) + L (T) - G (U P T
I
N
N
N
I
N
N
N
E
-
-

-)

The following notations were adopted:

.
F (U, P
, T
contains the work of deformation, the contributions to the current moment of the terms
I
)
of hydraulic and thermal dissipation expressed within - method, and of the variations
of fluid contribution of mass and entropy;
.
DF appoints the tangent operator, who can not be brought up to date with each iteration in
I
(U, P, T, according to a compromise cost calculation-speed of convergence; convergence is
N
N
N)
checked by a test on the relative standard of the difference of reiterated successive (via the key word
INCO_GLOB_RELA);
.
G (U, P
, T
-
-
-
) contains the contributions to the previous moment of the terms of dissipation
hydraulics and thermics expressed within - method, and of the variations of contribution of
mass fluid and of entropy;
.
L (T indicates the virtual work of the "dead" forces external and external contributions
E
)
hydraulics and of heat expressed by - method.

With convergence with the iteration
N +1, one operates an actualization of the fields
(U, P, T) = (U
+
+
+
+ ,
P

1
+, T
N
N 1
n+
1) .

In the version present of algorithm THM, we decided to gather all the terms y included/understood those due to the following forces and those of time less:

While posing:

- R (U, P, T) = - F (U, P, T) - G (U, -
P, -
T
=
I
N
N
N
I
N
N
N
), therefore DF DR.

I
I

one has finally:

DF · (U, P, T
+

, ,
1
1
+
+
+1) = - R (U
P T) + L (T
I
N
N
N
I
N
N
N
E
)

The algorithm general of balance will be written then, for a step of time:

Initializations:
Calculation of L (t+ (option
E
)
CHAR_MECA)
Calculation of DF
(option
I (U P T
RIGI_MECA-TANG)
-
-
-)
, ,
Calculation of (U,
P,
T by: DF

· U
, P

0
, T

0
0 = L t+ - L T
0
0 0)
I (U P T
-
-
-)
, ,
(
) E () E ()

Iterations of balance of Newton N
If option FULL_MECA:
Calculation of DF
and - (+ + +
R U, P, T:
I
N
N
N)
I (u+, P+, T+
N
N
N)
update stamps tangent: DF = DF

I
I (+, P+, T+
N N)
If option RAPH_MECA
Calculation of - (+ + +
R U, P, T
I
N
N
N)

Calculation of (U, P, T
by:
n+1 n+1 n+1)
DF · (U, P, T
+
+
+
+

,
,

1
1
+
+
+1) = - R (U P T) + L (T
I
N
N
N
I
N
N
N
E
)
Actualization:
(u+, P+, T+
+
+
+
,
,

,
,

1
1
1 =
+
+
+

+)
(U P T) (U P T
N
N
N
N
N
N
n+1
n+1
n+1)
IF test convergence OK
fine Newton: no next time
If not
N = n+1

4.2
Buckle on the elements, the points of Gauss

As in all the codes of finite elements, the terms are calculated by loop on the elements and buckle on the points of gauss:

R
+
+
+
el
el
U,
,

u+
+
+
=
,
,
I (
P T
N
N
N)
W R

G
G I (
P T
N
N
N)
el
G

DF
el
el
+
+
+
=
,
,

I (U P T
+
+
+
N
N
N)
W DF
G
G I (U, P, T
N
N
N)
el
G
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

Let us note: {X el} the vector of the nodal unknown factors, on a finite element el

U
v
W
1

node
p1
p
2
T
U
v
W
for example

{Xel} =
2

node
p1
p
2
T
U
v
W

3

node
p1
p

2
T

In this paragraph, to simplify the presentation, we suppose that we deal with one supporting finite element of the ddl of displacement, two ddl of pressure and a ddl of temperature.

Let us note {el the vector of the deformations generalized at the point of gauss G of the element el G}
For example:

U
*(**U**)*

p
1
{
p

1
el
=

G}

p
2
p

2
T

T

We note {el the vector of constraints generalized for the point of Gauss G of the element el G}
Data-processing handbook of Description

For example, and always in the most complete case:

p

m1
11
M1
h1
1 m
2
m
1
2
M
1
{
H2

1

el
m

=
G}
1
m
2
1
M
2
1
H
2m
m2
2
2
M2
H2
2 m

Q'

Q

el
The routines finite elements calculate the matrix: [B] defined by:
G

{el
el
= B
X
G}
[] G {}

:

22/06/05
Author (S):
***C. CHAVANT** Key*
:
D9.05.03-A Page
: 12/36

The algorithm will become then:

Initializations:
Calculation of L (t+ (option
E
)
CHAR_MECA)
Calculation of DF
(option
*I (**U** P T*
RIGI_MECA-TANG)
-
-
-)
, ,
*Calculation of (**U**,*
P,
T by: DF
*· **U***
, P

0
, T

0
0 = L t+ - L T
0
0 0)
*I (**U** P T*
-
-
-)

, ,
(
) E () E ()
Iterations of balance of Newton N
Buckle elements el
Buckle points of gauss G
el
calculation [B]
G
-
el
+
el
calculation {el
B
X -
=
and {el
B
X +
=

G N} [] G {N}
G} [] G {
}
el
Calculation +
el
+
+
+
el
G, - *R (**U**, P, T*
and DF
(according to options) from:
I G
N
N
N)

N
*G I (**u+**, P+, T+*
N

N
N)
{el- el+ el- el+ el
,
,
,
, B
G} {G N} {G} {G N} [] G
*Calculation of (**U**, P, T*
by:
n+1 n+1 n+1)
*DF · (**U**, P, T*
+
+
+
+

,
,

1
1
+
+
*+1) = - R (**U** P T) + L (T*
I
N
N
N
I
N
N
N
E
)
Actualization:
*(**u+**, P+, T+*
+
+
+
,
,

,

,

1
1
1 =
+
+
+
+)
*(**U** P T) (**U** P T*
N
N
N
N
N
N
n+1
n+1
n+1)
IF test convergence OK
fine Newton: no next time
If not
N = n+1

4.3
Vectors and matrices according to options': routine EQUTHM

The framed central part of the algorithm presented Ci above is carried out by a generic routine EQUTHM. We give in appendix a chart of the call of this routine.

This routine is parameterized according to the equations present (mechanics, hydraulics with 1 or 2 pressures, thermics). The work carried out by this routine is parameterized by the option.

*The term - R (**U**, P, T will be calculated by the options*
I
N
N
N)
RAPH_MECA and FULL_MECA. This term includes them
following forces of volume: it will be considered that the following forces will be integrated into the options
RAPH_MECA, FULL_MECA and RIGI_MECA_TANG. If the user data
*do not comprise forces of volume, the **Fm+** vector will be simply null.*

The presentations made in the two following paragraphs are made in the case more general where there is an equation of mechanics, two equations of hydraulics and an equation of thermics.
routine EQUTHM will calculate or not the various terms according to description that one will make him
equations present.

The indices G and el from now on are omitted, but it is clear that what is described applies to each point of gauss of each element.

4.3.1 Residue or nodal force: options RAPH_MECA and FULL_MECA

One will distribute the terms of the variational formulation according to the following principle:

el
el T
If *g indicates a virtual field of deformation, *g = (v, (v),
calculated
1 ,
1 ,
2 ,
2 ,)
starting from a vector of displacement nodal virtual: {X el*}
****elT***
el
G. R

U
P T = R v

1
+ R v
2
+ R3 1 + R4 1 + R5 2 + R6 2 + R7 + R8
I G (
,
,
+
+
+)
( )

One has then:

Index R
associated

+
+
+

1
+
1
2
1
2
-
+
m
v
1
+ m1 + m2 + 2 m
m
F

2
+
+
+ I
(v)
p

3
- *1+* **-** *2+*
1
2

1
+ **-**
1
+
-
m
m
m
m
1
1
1

4
T (M1+ M2+
*1***-**
2
1
-
+
+ **-**
+

1)
() T (M
M

1
1
1)
1

5
- *1+* **-** *2+*
1
2
2
2
+ **-**
2
+
-
m
m
m
m
2
2

6
*T (**M1+** M2+*
1 **-**
2
1
-
+
+ **-**
+

2) (
*) T (**M***
M
2
2
2)
2

7

Q'+ - Q'-

(h1+
1-
1+
1-
2+
2-
2+
2-

1
1
1
+ - 1
1
- 1 + 1 + - 1
1
-
m
(
) H m) (m m) (H m () H m) (m m1)
(

h1+
1-
1+
1-
2+
2-
2+
2-

1
1
2
+ - 2
2
- 2 + 2 + - 2
2
-
m
(

) H m) (m m) (H m () H m) (m m2)
- T
(1 +
2 +
1 +
2 +
1
2
1
M
M
M
M
F
1
M
(
M
M
M2 -. Fm
2
)
1
+ 1 + 2 + 2) m
.
- T (-)
-
-
-
1
+ 1 + 2 +

8
- T (1+ 1+
H
2
2
1
1
2
2
M

M
M
M
Q

1
1 +
+
+
h1
1
+ +
+
H2
2
+ +
+
H2
2
+ +
m
m
m
m
)+
- (-) T (1 - 1
H
2
2
1
1
2
2
1
M
M
M
M
Q
1
1 +
-
-

h1
1
+ **-**
-
H2
2
+ **-**
-
H2
2
+ **-**
m
m
m
m
)

*From there one will define the vector nodal residue {**Vel** such as:*
***G**}*
{T
X el
}. {Vel** =*

***G**}*
el T
***
el
G. R
***U**,*
I G (
P T
+
+
+)

*{**Vel** will be calculated by:*
***G**}*
*{**Vel**} = [**Beautiful** T.R*
G
G] {}
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

4.3.2 Loading: options CHAR_MECA

This chapter is here only for memory because routine EQUTHM will not deal with these terms.

One will distribute the terms of the variational formulation according to the following principle:

****elT***
el
+
G. L
T
= L v
v

1
+ L2
+ L3 1 + L4 + L5 2 + L6 2 + L7 + L8
E G ()
( )

Index
L
element type associated with
1

F ext+
edge
v

3
T (M1 m2
+
edge

1
1ext
1 ext.)
5
T (M1 m2
+
edge

2
2 ext.
2 ext.)
7
tR
volume

edge
- T (

1

1
2
2
Q +
M
1
1
+

M
ext.
(H
H
m
ext.
1 m
1 ext.)
- T (H 1 1
M
2
2
2
2
+ H

M
m
ext.
2 m
2ext)
= - t~q
ext.

4.3.3 Tangent operator: options FULL_MECA, RIGI_MECA_TANG

Notice on the matric notations:

In what follows, if X indicates a vector of components X I and Y a vector of components
X
X I
Y J,
line: I, column: J is
.

Y will indicate a matrix of which the element (
) Yj

To calculate tangent operator DF, the following quantities will be calculated:
I

[DRDE] =
DR1U DR1E DR1P1 DR1GP1

DR1P2 DR1GP2
DR1T DR1GT
DR2U DR2E DR2P1 DR2GP1
DR2P2 DR2GP2
DR2T DR2GT
DR3U DR3E DR3P1 DR3GP1
DR3P2 DR3GP2
DR3T DR3GT
DR4U DR4E DR4P1 DR4GP1
DR4P2 DR4GP2
DR4T DR4GT
DR5U DR5E DR5P1 DR5GP1
DR5P2 DR5GP2
DR5T DR5GT
DR6U DR6E DR6P1 DR6GP1
DR6P2 DR6GP2
DR6T DR6GT
DR7U DR7E DR7P1 DR7GP1
DR7P2 DR7GP2
DR7T DR7GT
DR8U DR8E DR8P1 DR8GP1
DR8P2 DR8GP2
DR8T DR8GT

Where one noted:

F

DRiU

I

= U

F

DRiE

I

=

F

DRiP

I

1 = p

1

F

DRiP

I

2 = p

2

F

DRiGP

I

1 =

p1

F

DRiGP

I

2 =

p2

F

DRiT

I

= T

F

DRiGT

I

=

T

To make these calculations one considers that the laws of behavior will provide, for the options corresponding, all the derivative following:

U

p

1
p1 p2

p2
T
T

p
p
p

p
p
p
p
p

U

p

1
p1 p2

p2
T
T

m1
1
1
1
1
1
1
1

1
m1
m1
m1
m1
m1
m1
m1

U

p

1
p1 p2

p2
T
T

M1
1
M1
M1 M1 M1 M1 M1
1
M1 1
1
1
1
1
1

U

p
p
p
p
T
T

1
1
2
2

1
1
1
1
1
1
1
1

H
H
H

H
H
H
H
H

1 m
1 m
1 m
1 m
1 m
1 m
1 m
1 m

U

p
p
p
p
T
T

1
1
2
2

2
2
2
2
2
2
2

m
m
m
m
m

m
m
m2
1
1
1
1
1
1
1
1

U

p
p
p
p
T
T
1
1
2
2

2
2
2
2
2
2
2
2

M
M
M
M
M
M
M
M
1

1
1
1
1
1
1
1

U

p
p
p
p
T
T
1
1
2
2

2
2
2
2
2
2
2
2

H
H
H
H
H
H
H
H
1

m
1 m

1 m
1 m
1 m
1 m
1 m
1 m

U

p
p
p
p
T
T
[
1
1
2
2
DSDE] =
1
1
1
1
1
m
m
m
m
m
1
1
1
m
m
m

2
2
2
2
2

2
2
2

U

p
p
p
p
T
T

1
1
2
2

1
1
1
1
1
1
1
1

M
M
M
M
M
M
M
M

2
2
2
2
2
2

2
2

U

p
p
p
p
T
T

1
1
2
2

1
1
1
1
1
1
1
1

H
H
H
H
H
H
H
H

2m
2m
2m
2m
2m
2m
2m

2m

U

p
p
p
p
T
T

1
1
2
2

2
2
2

m
m
m
m2
2
2
2
2

2
m2
m2
m2
m
2
2
2
2

U

p

1
p1 p2
p2
T
T

2
2
2
2
2
2
2
2
M
M
M
M
M
M
M
M
2
2 2 2 2 2 2 2

U

p

1
p1 p2 p2
T
T

H2
2

2
2
2
2
2
2

2
H2
H2
H2
H2
H2
H2
H
m
m
m
m
m
m
m
2 m

U

p

T
1
p1 p2 p2
T

Q'
Q'
Q'
Q'
Q'
Q'
Q'
Q'

U

p1 p1 p2

p2
T

T

Q
Q
Q
Q
Q
Q
Q
Q

U

p

1
p1 p2

p2
T

T

Titrate:
New architecture THM. Integration of the equilibrium equations Dates
:

22/06/05
Author (S):
C. CHAVANT Key
:
D9.05.03-A Page
: 17/36

In fact, in these expressions, the derivative compared to U are all null, but we keep
el
the writing taking into account the definition of the matrices [B] which we adopted.
G

The call to the laws of behavior will provide the pieces of matrix [DSDE] according to equations' present:

[

p
p
p
p

1 1
2
DMECDE] =
;
[DMECP] 1 =

;

[DMECP2] =

2
[DMECDT] = T

T
p
p
p
p
p
p
p

p1

p1

p2 p2

T T
m1
1
1
1
1

m1
1
1
m1
1
m1
m
1
m1
m1

p

1

p1
p2 p

2

T
T
1

1
1
1
1

1
1
[
M

M
1 M
1
M1 M1
M
1 M
DP11DE] =
;
[
]
DP11P1 =
;
[DP
]
11P2
1

[DP11DT]
1

=

=
p
T
T
1

p1
p2 p

2

h1

1
1
1
1
h1
1
1
H
1m
h1
H
1
m
1 m
h1
H
m
1 m

m
m

p

T
T
1

p1
p2 p2

m2
2
2
2
2

m2
2
1
m1
1
m1
m
1
m1
m1

p

1

p1
p2 p

2

T
T
2

2
2
2
2

2
2
[
M

M1 M
1
M1 M1
M
1 M
DP12DE] =
;
[DP
]
12P1 =
;
[DP
]
12P2
1
[DP12DT]
1

=

=
p

T
T
1

p1
p2 p

2
H2

2
2

2
2
H2
2
1
H
1 m
h1
H
1
m
1 m
h1
H
m
1 m

m
m

p

T
T
1
p1
p2
p2
m1
1
1
1
1

m1
1

2
m2
2
m2
m
2
m2
m2

p

1

p1
p2
p

2

T
T
1

1
1
1
1

1
1
[
M

M2 M
2
M2 M2
M2 M

DP21DE] =
;
[DP
]
21P1 =
;
[DP
]
21P2 =
2

[DP21DT] =
2

p
T
T
1
p1
p2
p

2
h1

1
1

1
1
h1
1
2
H
2m
H2
H
m
2m
H2
H
m

2m

m
2m

p

T
T
1
p
1
p2
p2
m2
2
2
2
2

m2
2
2
m2
2
m2
m
2
m2
m2

p

1

p1
p2
p

2

T
T
2

2
2
2
2

2
2
[
M

M2 M
2
M2 M2
M2 M
DP22DE] =
;
[DP
]
22P1 =
;
[DP
]
22P2 =
2

[DP22DT] =
2

p
T
T
1
p1
p2
p

2
H2

2
2

2
2
H2
2
2
H
2 m
H2
H
m
2 m
H2
H
m
2 m

m
2 m

p

T
T
1
p
1
p2
p2
Q
'
Q
'
Q
'
Q

'

Q

'

Q

'

Q

'

[

DTDE]

=

p

1 p

p

1

2 p

;

[

]

DTDP1

=

;

[DTDP2]

=

2

[DTDT]

T

=

T

Q

Q

Q

Q
Q
Q
Q

p
1

p

1
p
2

p

2
T

T

In addition, by deriving the expression from the residue compared to the constraints, one defines:

1

R
1
R
1
R
1
R
1
R
1
R
1
R
1
R
1
R
1
R
1
R
1
R
1
R
1
R
1
R

1
1
1
2
2
2
1
1
1
2

2
2

p m
M
H
m
H
m
H
m
H
Q'
1
1 1

M
m
1
1 1

M
m
2
2 2

M
m
2
2

Q

2 m

R2 R2
R2 R2
R2
R2
R2
R2
R2 R2
R2

R2
R2
R2
R2 R2

1
1
1
2
2
2
1
1
1
2
2
2

p m
M
H
m
H
m
H
m
H
Q'
1
1 1

M
m
1
1 1

M
m
2
2 2

M

m
2
2

Q

2 m

3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R
3
R

1
1

1
2
2
2
1
1
1
2
2
2

p m
M
H
m
H
m
H
m
H
Q'
1
1 1

M
m
1
1 1

M
m
2
2 2

M
m
2
2

Q

2 m

R4
R4 R4 R4
R4
R4
R4
R4
R4 R4
R4
R4
R4
R4
R4 R4

1
1
1
2
2
2
1
1
1
2
2
2

[

p m
H
m
H
m
H
m
H
Q'
1
1 1

m
1

1 1

m
2
2 2

m
2
2

DRDS]

M
M
M
M
Q
2 m

= 5R 5R 5R 5R 5R 5R 5R
v
5
R
5
R
5
R
5
R
5
R
5
R
5
R

1
1
1
2

2
2
1
1
1
2
2
2

p m
M
H
m
H
m
H
m
H
Q'
1
1 1

M
m
1
1 1

M
m
2
2 2

M
m
2
2

Q

2 m

R6

6
R
6
R
6
R
6
R
6
R
6
R
6
R
6
R
R6
R6
6
R
6
R
6
R
R6 6
R

1
1
1
2
2
2
1
1
1
2
2
2

p m
M
H
m
H
m
H
m
H
Q'
1
1 1

M
m
1
1 1

M
m
2
2 2

M
m
2
2

Q

2 m

R7
R7 R7 R7
R7
7
R
7
R
7
R
7
R
R7

7
R
R7
R7
R7
7
R
7
R

1
1
1
2
2
2
1
1
1
2
2
2

p m
H
m
H
m
H
m
H
Q'
1
1 1

m
1
1 1

m
2
2 2

m
2
2

M
M
M
M
Q
2 m

8
R
8
R
8
R
8
R
8
R
8
R
8
R
8
R
8
R
8
R
8
R
8
R
8
R
8
R

8
R

1
1
1
2
2
2
1
1
1
2
2
2

p m
M
H
m
H
m
H
m
H
Q'
1
1 1

M
m
1
1 1

M
m
2
2 2

M
m
2

2

Q

2 m

All these quantities not being inevitably calculated, one will note:

[DR1DS]

=
1
R
1
R
R
R
R
R
R
;
[
]
DR1P11 =
1
1 or
1
1
1

+
+

+
+
1
1

+
+
1
1
p
m

M
m
M
+
1

1
1

1
1
1
m
R1
R
R
R
R
[
1
1
1

DR.
]
1P12 =
1

2+

2+ or
2+
2+
2+
m1
M1
m1
M

1
h1 m
R1
R
R
R
R
[
1
1
1

DR.
]
1P21 =
1

1+
1 + or
1+
1 +
1+
m2
M2
m2
M

2
h2m
R1
R

R
R
R
[
1
1
1

DR.
]
1P22 =
1

2+
2+ or
2+
2+
2+
m2
M2
m2
M

2
h2m
[
1
R
R
DR1DT]

=
1

+
+
Q'
Q

In the same way:

[DR8DS], [

]
DR8P11 [
,
]
DR8P12, [D
]
R8P21, [DR8P2]
2, [DR8DT]

It is then clear that:

[DRDE] = [DRDS]. [DSDE]

And the contribution of the point of gauss to tangent matrix DF el
is obtained by:
G I (u+, P+, T+
N
N
N)
DFel

el T
el
.
.

I

(u+, P+, T+
G
N
N
N)
[B] [DRDE] [B
G
]

=
G
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

5 Diagrams
general

STAT_NON_LINE

finite element

Routine YOU…

Routine EQUTHM
Comp méca

Connait unknown factors:
Arg In:
Comp hydrau

· T, P1, P 2, U
nature unknown factors
Comp ther

Call behavior

Buckle points of gauss

'

, ;
p

Calculate deformations:

1
1
1
Assemble contibution
m, M, H
;
1
1 1m

(U),

not gauss with residue
2
m, m2, 2
H
;
1
1
1m
p, p,
1
2 p,
1 p,
2
and/or M tgte
1
T, T
m, M1, 1
H
;
2
2
2m
2
m, m2, 2

H
;
2
2
2m
Calculate [B] elg
'
Q, Q
Call EQUTHM
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

6
Specifications of under generic program EQUTHM

6.1
Arguments of the routine

ARGUMENTS Of ENTRY: IN
COMPOR
Description of the behavior
OPTION
Option to be calculated

NDIM
dimension spaces
2 or 3
NDDL
Numbers total degrees of
freedom of the appealing element
DIMDEF
dimension of the table of
deformations generalized with
not gauss
DIMCON
dimension of the table of
constraints generalized with
not gauss
NVIMEC
A number of internal variables

“mechanical”
ADVIME
Address variables

mechanical interns in
table of the internal variables
at the point of gauss
NVIHY
A number of internal variables
“hydraulic”
ADVIHY
Address variables

hydraulic interns in
table of the internal variables
at the point of gauss
NVITM
A number of internal variables
“thermal”
ADVITM
Address variables

thermal interns in
table of the internal variables
at the point of gauss
B (1: dimdef, 1: nddl)
el

Stamp [B]
G
DEFGEP (1: dimdef)
Values of deformations

generalized at the point of
gauss time more
DEFGEM (1: dimdef)
Values of deformations

generalized at the point of
gauss time less
CONGEM (1: dimcon)
Values of constraints

generalized at the point of

gauss time less
VINTM (1: nvimec+nvihy+
Values of the internal variables
nvitm)
at the point of gauss time
less
MECA (1: 5)
YAMEC = MECA (1)
logic if 1 there is an equation of
mechanics

ADDEME = MECA (2)
Address in the tables of
deformations at the point of gauss
DEFGEP and DEFGEM of the deformations
corresponding to mechanics

ADCOME = MECA (3)
Address in the tables of
constraints at the point of gauss CONGEP
and
CONGEM of the constraints

---

corresponding to the equation ieq

NDEFME = MECA (4)

A number of mechanical deformations

NCONME = MECA (5)
A number of mechanical constraints
PRESS1 (1: 5)
YAP1 = PRESS1 (1)
logic if 1 there is an equation
constituting 1

NBPHA1 = PRESS1 (2)
a number of phases for the component

1

ADDEP1 = PRESS1 (3)
Address in the tables of
deformations at the point of gauss
DEFGEP and DEFGEM of the deformations
corresponding to the first pressure

ADCP11 = PRESS1 (4)
Address in the tables of
constraints at the point of gauss CONGEP
and
CONGEM of the constraints
corresponding to the first phase of
first component

ADCP12 = PRESS1 (5)
Address in the tables of
constraints at the point of gauss CONGEP
and
CONGEM of the constraints
corresponding to the second phase
first component

NDEFP1 = PRESS1 (6)
A number of deformations pressure 1

NCONP1 = PRESS1 (7)
A number of constraints for each
phase of component 1
PRESS2 (1: 5)
YAP2 = PRESS2 (1)

logic if 1 there is an equation constituting 2

NBPHA2 = PRESS2 (2)
a number of phases for the component 2

ADDEP2 = PRESS2 (3)
Address in the tables of deformations at the point of gauss DEFGEP and DEFGEM of the deformations correspondent with PRE2

ADCP21 = PRESS2 (4)
Address in the tables of constraints at the point of gauss CONGEP and CONGEM of the constraints corresponding to the first phase of second component

ADCP22 = PRESS2 (5)
Address in the tables of constraints at the point of gauss CONGEP and CONGEM of the constraints corresponding to the second phase second component

NDEFP2 = PRESS2 (6)
A number of deformations pressure 2

NCONP2 = PRESS2 (7)
A number of constraints for each phase of component 2
TEMPLE (1: 5)
YATE = TEMPLE (1)
logic if 1 there is an equation of thermics

ADDETE = TEMPLE (2)
Address in the tables of deformations at the point of gauss DEFGEP and DEFGEM of the deformations

corresponding to thermics

ADCOTE = TEMPLE (3)
Address in the tables of

constraints at the point of gauss CONGEP

and first CONGEM of the constraints

corresponding to thermics

NDEFT = TEMPLE (4)
A thermal number of deformations

NCONT = TEMPLE (5)
A number of thermal stresses

ARGUMENTS OF EXIT: OUT
CONGEP (1: dimcon)
Values of constraints

generalized at the point of
gauss time more
VINTP (1: nvimec+nvihy+
Values of the internal variables
nvitm)

at the point of gauss time more
V (1: nddl)
{

***Vel**} = [**Beautiful** T.R*
G
G] {}
CHECHMATE (1: nddl, 1: nddl)

DF

el

el T
el
.
.
I

(u+, P+, T+
G
N
N
***N**)*
*[**B**] [**DRDE**] [**B***
G
]

=
G
TABLES OF WORK
R (1: dimdef)

DRDS

(1: dimdef, 1: dimcon)
DSDE

(1: dimcon, 1: dimdef)

6.2

Addressing in the tables of deformation and constraint

6.2.1 Addressing in the deformations

6.2.1.1 Deformations time less

Part
Significance
Address in DEFGEM
(local name in routine
COMTHM)
DEMECM
U, (U)
ADDEME
DEP1M
p, p
ADDEP1
1
1
DEP2M
p, p
ADDEP2
2
2
DETM
T, T
ADDETE

6.2.1.2 Deformations time more

Part
Significance
Address in DEFGEP
(local name in routine
COMTHM)
DEMECP
U, (U)
ADDEME
DEP1P
p, p
ADDEP1
1
1
DEP2P

p, p
ADDEP2
2
2
DETP
T, T
ADDETE

6.2.2 Addressing in the constraints

6.2.2.1 Forced time less

Part
Significance
Address in CONGEM
(local name in routine
COMTHM)
COMECM
,
ADCOME
p
CP11M
m1
1

M
m1
1
or
M h1
,
,
,

ADCP11
1
1
1
1
1 m
CP12M
m2
2
M
m2
2
or
M H2
,
,
,

ADCP12
1
1
1
1
1 m
CP21M
m1
1
M
m1
1
or
M h1
,
,

,

ADCP21
2
2
2
2
2m
CP22M
m2
2
M
m2
2
or
M H2
,
,
,

ADCP22
2
2
2
2
2 m
COTM
Q', Q
ADCOTE

6.2.2.2 Forced time more

Part
Significance
Address in CONGEP
(local name in routine
COMTHM)
COMECP
,
ADCOME
p
CP11P
m1
1

M
m1
1
or
M h1
,
,
,

ADCP11
1
1
1
1
1 m
CP12P
m2
2
M
m2
2
or
M H2
,
,
,

ADCP12
1
1
1
1
1 m
CP21P
m1
1
M
m1
1
or
M h1
,
,

,

ADCP21
2
2
2
2
2m
CP22P
m2
2
M
m2
2
or
M H2
,
,
,

ADCP22
2
2
2
2
2 m
COTP
Q', Q
ADCOTE

6.2.3 Addressing in the variables intern (example)

6.2.3.1 Variables intern at time less

Part
Significance
Address in VINTM
(local name in routine
COMTHM)
VIMEM

ADVIME

VIHYM
Slq, vp
P, lq
P
ADVIHY

6.2.3.2 Variables intern at time more

Part
Significance
Address in VINTP
(local name in routine
COMTHM)
VIMEP

ADVIME

VIHYP
Slq, vp
P, lq
P
ADVIHY

6.3 Addressing

R, DRDS, DSDE

6.3.1 Addressing
in
R

Under part of R
Associated
Address in R
R1
v
ADDEME
R2
(v)
ADDEME+NDIM
R3

ADDEP1
1
R4

ADDEP1+1
1
R5

ADDEP2
2
R6

ADDEP2+1
2
R7

ADDETE
R8

ADDETE+1

6.3.2 Addressing
in
DRDS

Part of table DRDS Significance
Address in DRDS
DR1DS

ADDEME, ADCOME
R

R
1
1

+
+

p

DR2DS

ADDEME+NDIM-1, ADCOME
DR1P11
R

ADDEME, ADCP11
1
R1

or
m1+
1+
M
1
1

R

1
R1
R1

m1+
1+
1+

M
1

1
h1
m
DR2P11

ADDEME+NDIM-1, ADCP11
DR1P12
R

ADDEME, ADCP12
1
R1

or
m2+
2+
M
1
1

R

1
R1
R1

m2+
2+
2+
M
1
1
h1

m
DR2P12

ADDEME+NDIM-1, ADCP12
DR1P21
R

ADDEME, ADCP21
1

R1

or
m1+
1 +
M
2
2

R

1
R1
R1

m1+
1 +
1+
M
2
2
H2

m
DR2P21

ADDEME+NDIM-1, ADCP21
DR1P22
R

ADDEME, ADCP22
1
R1

or
m2+
2+
M
2
2

R

1
R1
R1

m2+
2+
2+
M
2
2
H2

m
DR2P22

ADDEME+NDIM-1, ADCP22

DR1DT
R
R
ADDEME, ADCOTE
1
1

Q'+ +
Q
DR2DT

ADDEME+NDIM-1, ADCOTE
DR3DS

ADDEP1, ADCOME
DR4DS

ADDEP1+1, ADCOME
DR3P11

ADDEP1, ADCP11
DR4P11

ADDEP1+1, ADCP11
DR3P21

ADDEP1, ADCP21
DR4P21

ADDEP1+ 1, ADCP21
DR3DT

ADDEP1, ADCOTE
DR4DT

ADDEP1+ 1, ADCOTE
DR5DS

ADDEP2, ADCOME
DR6DS

ADDEP2+ 1, ADCOME
DR5P11

ADDEP2, ADCP11
DR6P11

ADDEP2+ 1, ADCP11
DR5P21

ADDEP2, ADCP21
DR6P21

ADDEP2+1, ADCP21
DR5DT

ADDEP2, ADCOTE
DR6DT

ADDEP2+ 1, ADCOTE
DR7DS

ADDETE, ADCOME
DR8DS

ADDETE+ 1, ADCOME
DR7P11

ADDETE, ADCP11
DR8P11

ADDETE+ 1, ADCP11
DR7P21

ADDETE, ADCP21
DR8P21

ADDETE+ 1, ADCP21
DR7DT

ADDETE, ADCOTE
DR8DT

ADDETE+1, ADCOTE

6.3.3 Addressing
in
DSDE

Part
Significance
Address in DSDE
(local name with
COMTHM)
DMECDE

ADCOME, ADDEME

p

DMECP1

ADCOME, ADDEP1
p1

p1

p

p

p1 p1
DMECP2

ADCOME, ADDEP2
p2

p2

p

p

p2 p2
DMECDT

ADCOME, ADDETE
T

T

p

p

DP11DE
m1
ADCP11, ADDEME

1

1
M1

h1
1 m

DP11P1
m1
1
ADCP11, ADDEP1
1
m1

p

1
p1

1
1
M
M
1
1

p

1
p1

h1
1
1
H
m
1 m

p

1
p1
DP11P2
m1
1
ADCP11, ADDEP2
1
m1

p

2
p2

1

1
M
M
1
1

p

2
p2

h1
1
1
H
m
1 m

p

2
p2
DP11DT
m1
1
ADCP11, ADDETE
1
m1
T T
1
1
M
M
1
1

T
T
h1
1

1
H
m
1 m

T
T
DP12DE
m2
ADCP12, ADDEME

1

2
M1

H2
1 m

DP12P1
m2
2
ADCP12, ADDEP1
1
m1

p

1
p1

2
2
M
M
1
1

p

1
p1

H2
2
1
H
m
1 m

p

1
p1

DP12P2
m2
2
ADCP12, ADDEP2
1
m1

p

2
p2

2
2
M
M
1
1

p

2
p2

H2
2
1
H
m
1 m

p

2
p2
DP12DT
m2
2
ADCP12, ADDETE
1
m1
T T

2
2
M
M
1
1

T
T

H2
2

1
H
m
1 m

T
T

DP21DE
m1
ADCP21, ADDEME
2

1
M2

h1
2m

DP21P1
m1
1
ADCP21, ADDEP1
2
m2

p

1
p1

1
1
M
M
2
2

p

1
p1

h1
1
2
H
m
2m

p

1
p1
DP21P2
m1
1
ADCP21, ADDEP2
2
m2

p

2
p2

1
1
M
M
2

2

p

2
p2

h1
1
2
H
m
2m

p

2
p2
DP21DT
m1
1
ADCP21, ADDETE
2
m2
T T

1
1
M
M
2
2

T
T

h1
1

2

H
m
2m

T
T

DP22DE
m2
ADCP22, ADDEME
2

2
M2

H2
2 m

DP22P1
m2
2

ADCP22, ADDEP1
2
m2

p

1
p1

2
2
M
M
2
2

p

1
p1

H2
2
2
H
m
2 m

p

1
p1
DP22P2
m2
2
ADCP22, ADDEP2
2
m2

p

2
p2

2
2
M
M
2
2

p

2
p2

H2
2
2
H
m
2 m

p

2
p2
DP22DT
m2
2
ADCP22, ADDETE
2
m2
T T

2
2
M
M
2
2

T
T

H2
2

2
H
m
2 m

T
T

DTDE
Q'
ADCOTE, ADDEME

Q

DTDP1
Q'
Q'
ADCOTE, ADDEP1
p

1
p1
Q

Q

p

1
p1

DTDP2
Q'
Q'
ADCOTE, ADDEP2
p

2
p2
Q

Q
p

2
p2

DTDT
Q'
Q'
ADCOTE, ADDETE
T T
Q

Q
T
T

Data-processing handbook of Description

6.4 Algorithm routine EQUTHM

YAMEC = MECA (1)
ADDEME = MECA (2)
ADCOME = MECA (3)
NDEFME = MECA (4)
NCONME = MECA (5)

YAP1 = PRESS1 (1)
NBPHA1 = PRESS1 (2)
ADDEP1 = PRESS1 (3)
ADCP11 = PRESS1 (4)
ADCP12 = PRESS1 (5)
NDEFP1 = PRESS1 (6)
NCONP1 = PRESS1 (7)
YAP2 = PRESS2 (1)
NBPHA2 = PRESS2 (2)
ADDEP2 = PRESS2 (3)
ADCP21 = PRESS2 (4)
ADCP22 = PRESS2 (5)
NDEFP2 = PRESS2 (6)
NCONP2 = PRESS2 (7)

YATE = TEMPLE (1)
ADDETE = TEMPLE (2)
ADCOTE = TEMPLE (3)
NDEFT = TEMPLE (4)
NCONT = TEMPLE (5)

CALL COMTHM (

COMPOR OPTION NDIM
NDDL
DIMDEF
DIMCON
NVIMEC
NVIHY, NVITM
NDEFME
NDEFP1
NDEFP2
NDEFT
NCONME
NCONP1
NCONP2
NCONT
YAP1 NBPHA1
YAP2 NBPHA2
DEFGEM (ADDEME) DEFGEM (ADDEP1) DEFGEM (ADDEP2) DEFGEM (ADDETE)
DEFGEP (ADDEME) DEFGEP (ADDEP1) DEFGEP (ADDEP2) DEFGEP (ADDETE)
CONGEM (ADCOME) CONGEM (ADCOTE)

CONGEM (ADCP11) CONGEM (ADCP12) CONGEM (ADCP21) CONGEM (ADCP21)
VINTM (ADVIME) VINTM (ADVIHY) VINTM
(ADVITM)

CONGEP (ADCOME) CONGEP (ADCP11) CONGEP (ADCP21) CONGEP (ADCOTE)
VINTP (ADVIME) VINTP (ADVIHY) VINTP
(ADVITM)
DSDE
DSDE
DSDE
DSDE
(ADCOME, ADDEME) (ADCOME, ADDEP1) (ADCOME, ADDEP2) (ADCOME, ADDETE)
DSDE

DSDE
DSDE
DSDE
(ADCP11, ADDEP1) (ADCP11, ADDEME) (ADCP11, ADDEP2) (ADCP11, ADDETE)
DSDE
DSDE
DSDE
DSDE
(ADCP12, ADDEP1) (ADCP12, ADDEME) (ADCP12, ADDEP2) (ADCP12, ADDETE)
DSDE
DSDE
DSDE
DSDE
(ADCP21, ADDEP2) (ADCP21, ADDEME) (ADCP21, ADDEP1) (ADCP21, ADDETE)
DSDE
DSDE
DSDE
DSDE
(ADCP22, ADDEP2) (ADCP22, ADDEME) (ADCP22, ADDEP1) (ADCP22, ADDETE)
DSDE
DSDE
DSDE
DSDE
(ADCOTE, ADDETE) (ADCOTE, ADDEME) (ADCOTE, ADDEP1) (ADCOTE, ADDEP2)

)
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

If FULL_MECA or RAPH_MECA
If YAMEC
Injection of the terms +
+
+ I in R (ADDEME+NDIM-1)
p
Injection of the terms: -
+
R m
F
in R (ADDEME)
0

If YAP1
+
-
+
+
-
-
Injection of the terms - m1 + m1
1
2
1
2
or
in R (ADDEP1)
1

- m1 - m1 + m1 + m
1
1
Injection of the terms
tM1+
1
+ 1
-
-
or
1
() tM1

*T (**M1+** + M2+*
1-
2
+ 1
-
-
+
1
1)
*() T (**MR. M***
1
1)
in R (ADDEP1+1)
IF YAMEC
1+
+
m
1+
2+
+
*Injection of the terms: - m **F** or - (m*
***F** in R (ADDEME)*
1
+ m1) m
1
If YATE
Injection of the terms:
T (h1+
1-

1+
1-
1+
m
1-
m

+ 1-
-
-
MR. F - *1*
MR. F
1
1
1
1

1

m
(
) H m) (m m) T
(
) T1
or

T (H
1+
1-
1+
1-
2+
2-
2+
2
+ 1-
-
+
+ 1
-
-
-
1
1

1
1
1

m
(
) H m) (m m) T (H m () h1m) (m m
1
1)
-
t1+ m
1-
2 +
2
MR. F *- 1*
MR. F *-*
MR. F *- 1*
-
-
MR. F
1
() T
m
1

T
m
1
() T
m
1
in R (ADDETE)
Injection of the terms
-
1+
1+
HT
1
1
M
1
M *or*
1

1 - (-)
-
-
HT
m
1 m
1

- T (1+ 1+
H
2
2
1
1
2
2
M
M
1
M
M
1
1 +
+
+
H
m
1 m
1) - (-) T (-
-
h1
1 +
-
-
H
m
1 m
1)
in R (ADDETE+1)
If YAP2
1+
1-
1+

2 +
1-
2-
Injection of the terms + m - m or
in R (ADDEP2)
2
+ m2 + m2 - m2 - m
2
2
Injection of the terms
tM1+
1
+ 1
-
-
or
2
() tM2

*T (**M1+** +M2+*
1 -
2
+ 1
-
-
+
2
2) (
*) T (**MR. M***
2
2)
in R (ADDEP2+1)
IF YAMEC
1+
+
m
1+
2 +
+
*Injection of the terms: - m **F** or - (m*
***F** in R (ADDEME)*
2
+ m2) m

2
If YATE
Injection of the terms:

T (h1+
1-
1+
1-
1+
m
1-
m

+ 1-
-
-
MR. F *- 1*
MR. F
2
2
2
2

2

m
(
) H m) (m m) T
(
) T 2
or

T (H
1+
1-
1+
1-
2+
2-
2+
2
+ 1-
-
+

+ 1
-
-
-
2
2
2
2
2

m
(
) H m) (m m) T (H m () h2m) (m m
2
2)
-
t1+ m
1-
2 +
2
***MR. F** - 1*
***MR. F** -*
***MR. F** - 1*
-
-
MR. F
2
() T
m
2

T
m
2
() T
m
2
in R (ADDETE)
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

Code_Aster ®
Version

Injection of the terms
- +
+
1
1
T 2
H **M**
HT
or
m
2
- (1-) -
-
1
1
2 **M**
m
2
+
+
+
+

1
1
2
2
-
-
-

-
-
T 2
H ***M***
H
T H
H
m
2
+ 2 ***M***
m
2 - (1 -)
1
1
2
2
2 ***M***
m
2 + 2 ***M***
m
2

in R (ADDETE+1)
If YATE
Injection of the terms: Q'+ Q'-
-
in R (ADDETE)
Injection of the terms -
+
T ***Q*** *- (1 -)*
-

tq in R (ADDETE+1)
Accumulation in vector V:
{ }
V *= {}*
V *+ [****Beautiful*** *T.R*
G] {}
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

IF RAPH_MECA or RIGI_MECA_TANG
IF YAMEC
calculation of DR1DS and injection in DRDS (ADDEME, ADCOME)
calculation of DR2DS and injection in DRDS (ADDEME+NDIM-1, ADCOME)
IF YAP1
calculation of DR1P11 and injection in DRDS (ADDEME, ADCP11)
calculation of DR2P11 and injection in DRDS (ADDEME+NDIM-1, ADCP11)
IF NBPHA1 > 1
calculation of DR1P12 and injection in DRDS (ADDEME, ADCP12)
calculation of DR2P12 and injection in DRDS (ADDEME+NDIM-1, ADCP12)
IF YAP2
calculation of DR1P21 and injection in DRDS (ADDEME, ADCP21)
calculation of DR2P21 and injection in DRDS (ADDEME+NDIM-1, ADCP21)
IF NBPHA2 > 1
calculation of DR1P22 and injection in DRDS (ADDEME, ADCP22)
calculation of DR2P22 and injection in DRDS (ADDEME+NDIM-1, ADCP22)
IF YATE
calculation of DR1DT and injection in DRDS (ADDEME, ADCOTE)
calculation of DR2DT and injection in DRDS (ADDEME+NDIM-1, ADCOTE)
IF YAP1
calculation of DR3P11 and injection in DRDS (ADDEP1, ADCP11)
calculation of DR4P11 and injection in DRDS (ADDEP1+1, ADCP11)
IF NBPHA1 > 1
calculation of DR3P12 and injection in DRDS (ADDEP1, ADCP12)
calculation of DR4P12 and injection in DRDS (ADDEP1+1, ADCP12)
IF YAMEC
calculation of DR3DS and injection in DRDS (ADDEP1, ADCOME)
calculation of DR4DS and injection in DRDS (ADDEP1+1, ADCOME)

IF YAP2
calculation of DR3P21 and injection in DRDS (ADDEP1, ADCP21)
calculation of DR4P21 and injection in DRDS (ADDEP1+ 1, ADCP21)
IF NBPHA2 > 1
calculation of DR3P22 and injection in DRDS (ADDEP1, ADCP22)
calculation of DR4P21 and injection in DRDS (ADDEP1+ 1, ADCP22)
IF YATE
calculation of DR3DT and injection in DRDS (ADDEP1, ADCOTE)
calculation of DR4DT and injection in DRDS (ADDEP1+ 1, ADCOTE)
IF YAP2
calculation of DR5P21 and injection in DRDS (ADDEP2, ADCP21)
calculation of DR6P21 and injection in DRDS (ADDEP2+1, ADCP21)
IF NBPHA2 > 1
calculation of DR5P22 and injection in DRDS (ADDEP2, ADCP22)
calculation of DR6P22 and injection in DRDS (ADDEP2+1, ADCP22)
IF YAMEC
calculation of DR5DS and injection in DRDS (ADDEP2, ADCOME)
calculation of DR6DS and injection in DRDS (ADDEP2+ 1, ADCOME)
YAP1 thus:
calculation of DR5P11 and injection in DRDS (ADDEP2, ADCP11)
calculation of DR6P11 and injection in DRDS (ADDEP2+ 1, ADCP11)
IF NBPHA1 > 1
calculation of DR5P12 and injection in DRDS (ADDEP2, ADCP12)
calculation of DR6P12 and injection in DRDS (ADDEP2+ 1, ADCP12)
IF YATE
calculation of DR5DT and injection in DRDS (ADDEP2, ADCOTE)
calculation of DR6DT and injection in DRDS (ADDEP2+ 1, ADCOTE)
IF YATE
calculation of DR7DT and injection in DRDS (ADDETE, ADCOTE)
calculation of DR8DT and injection in DRDS (ADDETE+1, ADCOTE)
IF YAMEC
calculation of DR7DS and injection in DRDS (ADDETE, ADCOME)
calculation of DR8DS and injection in DRDS (ADDETE+ 1, ADCOME)

:

IF YAP1
calculation of DR7P11 and injection in DRDS (ADDETE, ADCP11)
calculation of DR8P11 and injection in DRDS (ADDETE+ 1, ADCP11)
IF NBPHA1 > 1
calculation of DR7P12 and injection in DRDS (ADDETE, ADCP12)
calculation of DR8P12 and injection in DRDS (ADDETE+ 1, ADCP12)
IF YAP2
calculation of DR7P21 and injection in DRDS (ADDETE, ADCP21)
calculation of DR8P21 and injection in DRDS (ADDETE+ 1, ADCP21)
IF NBPHA1 > 1
calculation of DR7P22 and injection in DRDS (ADDETE, ADCP22)
calculation of DR8P22 and injection in DRDS (ADDETE+ 1, ADCP22)

*[**DRDE**] = [**DRDS**]. [**DSDE**]*

DFel

el T
el
.
.
accumulated in CHECHMATE
I

(u+, P+, T+
G
N
N
N)
*[**B**] [**DRDE**] [**B***
G
]

=

G

6.5
Arguments of the routine of call of the laws of behavior

SUBROUTINE COMTHM (

ARGUMENTS Of ENTRY: IN
COMPOR OPTION NDIM
NDDL
DIMDEF
DIMCON
NVIMEC
NVIHY, NVITM
NDEFME
NDEFP1
NDEFP2
NDEFT
NCONME
NCONP1
NCONP2
NCONT
YAP1 NBPHA1
YAP2 NBPHA2
DEMECM
DEP1M
DEP2M
DETM
***U**, (**U**)*
p, p
p, p
T, T
1
1
2
2
time less
time less
time less
time less
DEMECP
DEP1P
DEP2P
DETP

***U**, (**U**)*
p, p
p, p
T, T
1
1
2
2
time more
time more
time more
time more
COMECM
COTM

,
*Q', **Q***
p
time less
time less
CP11M
CP12M
CP21M
CP21M
m1
1
*, **M** or*
m2
2
*, **M** or*
m1
1
*, **M** or*
m2
2
*, **M** or*
1
1
1
1
2
2

2
2
m1
1
***M** h1*
,
,

m2
2
***M** H2*
,
,

m1
1
***M** h1*
,
,

m2
2
***M** H2*
,
,

1
1
1 m
1
1
1 m
2
2
2m
2
2
2 m
time less
time less
time less
time less
VIMEM
VIHYM

VITMM

internal variables
internal variables
internal variables
méca
hydro
therm
time less
time less
time less
ARGUMENTS OF EXIT: OUT
COMECP
COTP

,
Q', ***Q***
p
time more
time more
CP11P
CP12P
CP21P
CP21P
m1
1
, ***M*** *or*
m2
2
, ***M*** *or*
m1
1
, ***M*** *or*
m2
2
, ***M*** *or*
1
1
1
1
2
2
2

2
m1
1
***M** h1*
,
,

m2
2
***M** H2*
,
,

m1
1
***M** h1*
,
,

m2
2
***M** H2*
,
,

1
1
1 m
1
1
1 m
2
2
2m
2
2
2 m
time more
time more
time more
time more
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

VIMEP
VIHYP
VITMP

internal variables
internal variables
internal variables
méca
hydro
therm
time more
time more
time more
DMECDE
DMECP1
DMECP2
DMECDT

p

p
T
T
2
p
1
p

1

2

p
p

p
p

p

p

p

T
T

p

p

2

p
1
p1
2
DP11DE
DP11P1
DP11P2

DP11DT
m1
m1
1
m1
1
m1
1
1
m
1
m
1

1
m

1
1
1

p

p

T
T
2
p
1
p
1
1

2

M
1
1
1
1
1
1
M
M
1

1

M
M
M
M
1

1

1

1

1

T
T

p

p

2
p
1

p
h1
1

2

h1
1
1
H
1 m

h1
1

h1
1

m
1 m
1
H
1
h1
1

m
m
m
m
T
T

p

p

2

p
1
p1
2
DP12DE
DP12P1
DP12P2
DP12DT
m2
m2
2
m2
2
m2
2
1
m
1
m
1

1
m

1
1
1

p

p

T
T

2
p
1

p
2
1

2

M
2
2
2
2
2
2
M
M
1

1

M
M
M
M
1

1

1

1

1

T
T

p

p

2
p
1

p
H2
1

2

H2
2

1
H
1 m

H2
2
H2
2
m
1 m
1
H
1
h1
1

m
m
m
m

T
T

p

p

2
p
1
p1
2
DP21DE
DP21P1
DP21P2
DP21DT
m1
m1
1
m1
1
m1
1
2
m
2
m
2

2
m

2
2
2

p

p

T
T

2
p
1
p
1
1

2

M
1
1
1
1
1
1
M
M
2

2

M
M
M
M
2

2

2

2

2

T
T

p

p

2
p
1

p
h1
1

2

h1
1

2
H
2m

h1
1
h1
1
m
2m
2
H
2
H2

2

m
m
m
m
T
T

p

p

2
p
1
p1
2

DP22DE
DP22P1

DP22P2
DP22DT
m2
m2
2
m2
2
m2
2
2
m
2
m
2

2
m

2
2
2

p

p

T
T

2
p
1
p
2
1

2

M
2
2
2
2
2
2
M
M
2

2

M
M
M
M
2

2

2

2

2

T
T

p

p

2
p

1

p
H2
1

2

H2
2

2
H
2 m

H2
2
H2
2
m
2 m
2
H
2
H2
2

m
m
m
m
T
T

p

p

2
p
1
p1
2
DTDE
DTDP1
DTDP2
DTDT
Q'
Q'
Q'
Q'
Q'
Q'
Q'

p

p

T T
2
p
1

p

1
Q

2

Q

Q
Q
Q
Q

Q

p

p

T
T
2
p
1
p

1

2

)

*REAL*8*
DEMECM (NDEFME), DEP1M (NDEFP1), DEP2M (NDEFP2), DETM (NDEFT)
DEMECP (NDEFME), DEP1P (NDEFP1), DEP2P (NDEFP2), DETP (NDEFT)
COMECM (NCONME), CP11M (NCONP1), CP21M (NCONP2), COTM (NCONT)

VIMEM (NVIMEC), VIHYM (NVIHY), VITMM (NVITM)

COMECP (NCONME), CP11P (NCONP1), CP21P (NCONP2), COTP (NCONT)
VIMEP (NVIMEC), VIHYP (NVIHY), VITMP (NVITM)

DMECDE (NCONME, NDEFME), DMECP1 (NCONME, NDEFP1),
DMECP2 (NCONME, NDEFP2), DMECDT (NCONME, NDEFT)
DP11DE (NCONP1, NDEFME), DP11P1 (NCONP1, NDEFP1),
DP11P2 (NCONP1, NDEFP2), DP11DT (NCONP1, NDEFT)
DP21DE (NCONP2, NDEFME), DP21P1 (NCONP2, NDEFP1,
DP21P2 (NCONP2, NDEFP2, DP21DT (NCONP2, NDEFT)

DP12DE (NCONP1, NDEFME), DP12P1 (NCONP1, NDEFP1),
DP12P2 (NCONP1, NDEFP2), DP12DT (NCONP1, NDEFT)
DP22DE (NCONP2, NDEFME), DP22P1 (NCONP2, NDEFP1,
DP22P2 (NCONP2, NDEFP2, DP22DT (NCONP2, NDEFT)

DTDE (NCONT2, NDEFME), DTDP1 (NCONT2, NDEFP1),
DTDP2 (NCONT2, NDEFP2), DTDT (NCONT2, NDEFT)

: 36/36

Intentionally white left page
Data-processing handbook of Description
D9.05 booklet: -
HT-66/05/003/A

Code_Aster ® *Version*
3

Titrate:
Data-processing description of IMPR_RESU

Date: 22/04/97
Author (S):
A.M. DONORE

Key: D9.07.01
Page:
1/24

Mechanical department and Digital Models

Index: With
Diffusion:
Users - Developers

Handbook of Reference

D9.07 booklet:

D9.07.01 document

Data-processing description of IMPR_RESU

***Summary**:*

This document is a data-processing description of the order IMPR_RESU, whose role is to print results of Code_Aster to the format "RESULT", "IDEAS", "CASTEM" or "ENSIGHT".

One finds the list of the principal routines used by the order, as well as a short summary of theirs functionalities.

One describes the characteristics of the impression of the results to format I-DEAS, as well as the format of
"datasets" constituting universal file IDEAS.

For format CASTEM, one describes the format of the file used by code CASTEM 2000.

Code_Aster ® *Version*
3

Titrate:
Data-processing description of IMPR_RESU

Date: 19/09/2003
Author (S):
A.M. DONORE

Key: D9.07.01
Page:
2/24

Tables of content

Handbook of Reference
D9.07 booklet:
Index A

Code_Aster ® *Version*
3

Titrate:
Data-processing description of IMPR_RESU

Date: 19/09/2003
Author (S):
A.M. DONORE

Key: D9.07.01
Page:
3/24

1
Tree of call of the principal routines of IMPR_RESU

1.1
Description of the routines appearing in the tree of call

OP0039.f
Program main associate with order IMPR_RESU. It is in this
routine which are read all the operands of order IMPR_RESU.
IRECRI.f
Writing of a field of size or a concept result. This routine can
to be called by other routines for the impression of a concept Aster. It
“hat” for the impression of results is to some extent a routine
Aster, deprived of any adherence to order IMPR_RESU (not
calls to routines supervisor). It is in this routine that one finds

loops on the sequence numbers of a concept result and on the list of reference symbols.
IRCH19.f
Routine of impression of a cham_no or a cham_elem.
IRCHML.f
Impression of a cham_elem with real or complex components with the format RESULT, IDEAS or CASTEM.
IRDEPL.f
Impression of a cham_no with real or complex components with the format RESULT, IDEAS, CASTEM or ENSIGHT.
IRCNRL.f
Impression of a cham_no with actual values with the format RESULT (on listing). This routine also allows the research and the note printing minimal and of the maximum value of the field to the nodes. These treatments can be carried out on all the field or only on values belonging to an interval defined by the user.

Handbook of Reference
D9.07 booklet:
Index A

IRCRRL.f
Impression of a cham_no with constant representation and actual values with format RESULT.
Seek and impression of the values minimal and maximum.
Treatment carried out on all the field or only on one definite interval by the user.
IRCNC8.f
Impression of a cham_no with complex values with the format RESULT.
Seek and impression of the values minimal and maximum.
Treatment carried out on all the field or only on one definite interval by the user.
IRDECA.f
Impression of a cham_no with actual values with format CASTEM (writing of crush number 2, to see description of the structure of file CASTEM 2000 [§2]).
IRDRCA.f
Impression of a cham_no with constant representation and actual values with format CASTEM (writing crushes number 2, to see description of the structure of file CASTEM 2000 [§2]).
IRDESR.f
Impression of a cham_no with actual values with the format of the universal file IDEAS (dataset 55, to see description of the structure of universal file IDEAS [§3]).
IRDRSR.f
Impression of a cham_no with constant representation and actual values with format of universal file IDEAS (dataset 55, to see description of the structure universal file IDEAS [§3]).

IRDESC.f
Impression of a cham_no with complex values with the format of the universal file IDEAS (dataset 55, to see description of the structure of universal file IDEAS [§3]).
IRDEER.f
Impression of a cham_no with actual values with format ENSIGHT.
IRDRER.f
Impression of a cham_no with constant representation and actual values with format ENSIGHT.
IRDEEC.f
Impression of a cham_no with complex values with format ENSIGHT.
IRDREC.f
Impression of a cham_no with constant representation and complex values with format ENSIGHT.
IRCERL.f
Impression of a cham_elem to the nodes or the points of Gauss and values real with the format RESULT.
Seek and impression of the values minimal and maximum.
Treatment carried out on all the field or only on one definite interval by the user.
IRCECL.f
Impression of a cham_elem to the nodes or the points of Gauss and values complexes with the format RESULT.
Seek and impression of the values minimal and maximum.
Treatment carried out on all the field or only on one definite interval by the user.
IRCERS.f
Impression of a cham_elem to the nodes or the points of Gauss and values real with the format of universal file IDEAS (datasets 56 and 57, to see description of the structure of universal file IDEAS).
Note: the cham_elem at the points of Gauss are written as constant fields by element, while realising on the points of Gauss).

IRCECS.f
Impression of a cham_elem to the nodes or the points of Gauss and values complexes with the format of universal file IDEAS (datasets 56 and 57, to see description of the structure of universal file IDEAS).
Note: the cham_elem at the points of Gauss are written as constant fields by element, while realising on the points of Gauss).
IRCECA.f
Impression of a cham_elem to the nodes and actual values with the format CASTEM (writing of the structure of file CASTEM 2000).
IRMARE.f
Impression of the grid to the format *Aster* (which can be read again by the order LIRE_MAILLAGE).
IRMACA.f
Impression of the grid to format CASTEM (impression of the piles number 0 (co-ordinates of the nodes) and number 1 (connectivities of the elements), to see description of the structure of file CASTEM 2000).
IRMASU.f
Impression of the grid to the format universal file IDEAS (datasets 15 (co-ordinates of the nodes single precision), 781 (co-ordinates of the nodes double precision double precision), 71 or 780 (connectivities of the meshs) and 752 (groups of nodes or of meshs), to see description of universal file IDEAS).
IRMAEN.f
Impression of the grid on file geometry ENSIGHT.

1.2 Others routines

ECRTES.f
Writing of the heading of the datasets 55, 56 and 57 at the time of the writing of a field of size *Aster* with format IDEAS.
This routine is called by the routines IRDESR.f, IRDRSR.f, IRDESC.f, IRCERS.f and IRCECS.f.

IRGAGS.f
Seek datasets IDEAS necessary to the writing of a field of size *Aster*.
This routine is called by the routines IRDESR.f, IRDRSR.f, IRDESC.f, IRCERS.f and IRCECS.f.
IRADHS.f
Treatment of "adherences IDEAS" at the time of the writing of a grid to the format IDEAS.
This routine is called by the routines IRMAIL.f and IRCHML.f.
INISTB.f
Initializations necessary to the writing of a grid to format IDEAS.
This routine is called by the IRADHS.f routine.
IRPACA.f
Impression of the values of the variables of access *Aster* to the format of the files CASTEM 2000 (piles 25, 26 and 27).
This routine is called by the IRECRI.f routine.
IRPARA.f
Impression of the values of the parameters and the variables of access to the format RESULT.
This routine is called by the IRECRI.f routine.
RSINFO.f
Impression of the list of the reference symbols and the sequence numbers available for a concept RESULT (key word INFO_RESU in IMPR_RESU).
This routine is called by OP0039.f.

Page:
6/24

2
Structure of file CASTEM 2000

This structure of file corresponds to level 3 of CASTEM 2000.

Each information contained in the file is preceded by a descriptor RECORDING BY TYPE followed by an entirety (I4). This one indicates the type of information which follows:

4 general Information on the grid

7 general Information CASTEM 2000

2 Description of a pile

5 End of file

2.1
Recording of the type 4: general information grid

General information on the grid. Two lines follow the descriptor.

LEVEL niv LEVEL ERROR ierr DIMENSION ndim DENSITY dens

niv: number of version CASTEM 2000 (niv = 3, I4)

ierr
:
level
of error
(ierr
=
0,
I4)

ndim: dimension of the problem

(ndim = 2 or 3, I4)

dens: density of the grid

(dens = 0.D0, E12.5)

2.2
Recording of the type 7: general information CASTEM 2000

General information CASTEM 2000. Two lines follow the descriptor.

INFORMATION CASTEM 2000 ninfo NUMBERS

IFOUR ifour NIFOUR nifour IFOMOD ifomod IECHO iecho IIMPI iimpi

IOSPI iospi ISOTYP isotyp

ninfo:
numbers information being reproduced on the following line (ninfo=7, I4)

ifour:
options of calculation
(- 1: plane deformations (if dimension 2), I4

2: three-dimensional (if dimension 3), I4)

nifour: harmonic of Fourier (0: no the harmonic, I4)

ifomod: type of mode

(- 1: plan (if dimension 2), I4

2:
three-dimensional
(if
dimension
3),
I4)

iecho:
echo of the data to the screen (iecho=1, I4)

iimpi:
level of impression

(iimpi=0, I4)

iospi:
trace operators

(iospi=0 not of trace, I4)

isotyp: type of layout of isovaleurs (isotyp=1, I4)

7/24

2.3
Recording of the type 2: description of a pile

Description of a pile. This line is always followed following line:

CRUSH NUMBER npile NO. OBJECTS NAME iobno NO. iob OBJECTS

npile:
described the type of elements contained in the pile (I4)

(
npile
=0
co-ordinates

nodes

=1
connectivity

elements

=2
fields
with
nodes

=10

description of a table

=25

crush realities

=26
crush
entireties

=27

crush of word type

=39
fields
by
elements
)

iobno:
a number of named objects
(I5)

iob:
a total number of objects
(I5)

2.3.1
Crush 0: co-ordinates of the nodes

CRUSH NUMBER 0 NO. OBJECTS NAME iobno NO. iob OBJECTS

The named objects are the groups of *Aster* nodes to only one node.

If the number of named objects is not null, there is two information;

.
the list of the names of groups of nodes *Aster* to only one node (8 (1X, A8)),
.
the position of each node in the pile (16I5).

Then the co-ordinates and the density of the points follow: X (I), I = 1, iob* (ndim+1)
(3 (1X, D21.14)).

2.3.2
Crush 1: connectivities of the elements

CRUSH NUMBER 1 NO. OBJECTS NAME iobno NO. iob OBJECTS

If iobno is not null, there are two lines, one describing the names of the objects (8 (1X, A8) and the other their position (16I5).

The named objects are the groups of nodes *Aster* to more than one node, the groups of *Aster* meshs and meshs of the *Aster* model (if requested by the user by the MODEL key word).

In the objects, one finds:

· the whole of the nodes of the grid in the form of meshs of the type POI1. This object is essential during the reading of fields to the nodes. This object is not named and is always the first object written in this pile,
· one
object
CASTEM 2000 for each group of nodes *Aster* to more than one node. It is
grids containing of the meshs of the type POI1,
· one
object
CASTEM 2000 for each group of meshs *Aster*. The object bears the name of the group
meshs in *Aster*,
· it
grid
Aster: object named with same name as the grid *Aster*,
· it
model
Aster (if it were stipulated in order IMPR_RESU by the MODEL key word):
object named with same name as the model *Aster*.

Note:

In Aster, a group of nodes and a group of meshs can have even name.
The interface writes the two objects with format CASTEM 2000 pennies the same name but under two forms distinct (meshs of the type POI1 for the group of nodes and meshs of the group of meshs).
Taking into account about writing in file CASTEM 2000, it is the object associated with group meshs which is accessible.

Then, for each grid object, a line describes this object:

--> itype nbsd nbref nbnoe nbelem

itype: number of the type of element grid (I5)

1:
POI1

2:
SEG2

3:
SEG3

4:
TRI3

6:
TRI6

8:
QUA4
10:
QUA8
14:
CUB8

15:
CU20
16:
PRI6

17:
PR15
23:
TET4

24:
TE10
25:
PYR5
26:
PY13

0: if the object is composed of under objects

nbsd: numbers of under objects (I5)
For the interface, the number of under objects is the number of the types of meshs composing the grid object.
(A group of meshs made up of triangles and quadrangles will be written in the form of three objects CASTEM 2000:
·
under object containing the triangles,
·
under object containing the quadrangles,
·
a named object pointing on the two other objects)

nbref: a number of references (=0, I5)

nbnoe: a number of nodes per element (I5) (=0 if object pointing on under objects)

nbelem:
a number of elements (I5) (=0 if object pointing on under objects)

Note:

Under object of a given object is the standard whole of the elements in the same way being able to be extracts. If the object consists of only one type of elements, there is not under objects.

If nbsd0, the list of the rows of under objects is registered with the line according to (16I5).

If itype=0, then necessarily nbsd0. That means that object is with a grid with elements of different types.

If nbsd=0,

One finds then the list of the numbers of the colors of each element

ICOLOR (I), I=1, nbelem (16I5)

Then the list of the numbers of the nodes “tops” of the elements

(NUM (I, J) I=1, nbnoe), j=1, nbelem (16I5)

2.3.3
Crush 2: field with the nodes

CRUSH NUMBER 2 NO. OBJECTS NAME iobno NO. iob OBJECTS

This pile makes it possible to write cham_no *Aster*. The cham_no can be fields of size named or of the fields of size resulting from a concept result (in which case they are not expressly named).

iobno = 1 if cham_no named Aster

0 if associated a reference symbol of a concept result

iob = 1

--> If iobno0, the two following lines appear in the file with format CASTEM 2000:

· name

CHAM_NO *Aster* (1X, A8),
·
number of occurrence of the field to the nodes in file CASTEM 2000 (I5).

--> nbsch nbcmp ifour (3I5)

nbsch: of "under fields to the nodes" = 1 (one numbers considers that all the nodes have it an even number of degrees of freedom; the envelope of the degrees of freedom present).

nbcmp: component count present at the nodes: wrap components.

ifour: 2 = standard of three-dimensional calculation

--> ipgeo nbpoin nbcmp (3I5)

ipgeo: pointer in the pile of connectivities of the elements = 1 (the object made up of elements of the type POI1 associated with all the nodes with the grid is always written at the beginning of
the pile of connectivities).

nbpoin: a number of points of the grid.

nbcmp: component count (idem that on the preceding line).

--> NOMCMP (I) (1: 4), I=1, nbcmp 16 (1X, A4)

list names of the components. They are the first four characters of the name of components *Aster*, except for displacements (DX becomes UX, DY becomes UY, DZ becomes

UZ, DRX becomes X-ray, DRY becomes RY and DRZ becomes RZ).

--> NOHARM (I), I=1, nbcmp (16I5)

corresponds to the numbers of the harmonics of Fourier. These parameters are identically
null in the case of the interface *Aster* - CASTEM 2000.

--> Name of the size (1X, A71).

--> 1 white line

--> (Value of the field (I, J), I=1, nbpoin), j=1, nbcmp) (3 (1X, E21.14)).

2.3.4
Crush 39: field by element

CRUSH NUMBER 39 NO. OBJECTS NAME iobno NO. iob OBJECTS

This pile makes it possible to write cham_elem with the *Aster* nodes. The cham_elem with the nodes can
to be named fields of sizes or fields of size resulting from a concept result (in

which case, they is not expressly named).

iobno = 1 if cham_elem with the nodes named in Aster

0 if associated a reference symbol of a concept result

iob = 1

--> If iobno0, the two following lines appear in the file with format CASTEM 2000:

· name

CHAM_ELEM *Aster* (1X, A8),
·
number of occurrence of the cham_elem to the nodes in file CASTEM 2000 (I5).

--> nzone ifour lcomp nbcar (4I5))

nzone: numbers of under zones. It is equal to the number of the type of elements present, in model.
A model made up of triangles and quadrangles are composed of two pennies zones.

ifour: type of three-dimensional calculation = 2

lcomp: 4

nbcar: a number of characters composing the type of the cham_elem = 16

--> title of the cham_elem: name of the size *Aster* (1X, A71)

(except
CONSTRAINTS for sizes SIEF* or SIGM*

DEFORMATIONS
for

sizes
EPSI*
)

--> (n1 N2 n3 n4 n5 n6 n7) X nzone (16 I5) n1: pointer in the pile of connectivities of the elements. Number of occurrence of the model *Aster* (or of its under objects if nzone 0) in the pile of connectivities

N2
: 0
n3
: component count of under zone
n4: 0
n5
: 0
n6: 0
n7
: 0
(cham_elem defined in the nodes. Documentation available does not indicate value for cham_elem at the points of Gauss).

--> For each component of under zone, number of the harmonic (16I5).
n3 X (0.) X nzone

--> Name of components (NOMCMP (I) (1: 8), I=1, n3) X nzone (8 (1X, A8))

The names of components are composed of the first 2 characters concaténés with

the last 2 characters of the name of the component *Aster*.
For the internal variables, there are VAR1, VAR2,…
For the components of constraints, SiXX becomes SMXX,…, SiYZ becomes SMYZ.

--> Standard of the components (“REAL*8”) X n3 X nzone 4 (1X, A16)

--> Value of the field by element to the nodes in each under zone

--> n1 N2 0 0 (16I5)

-->
(VALLEY
(I, J), i=1, n1)
(j=1, N2)
(1P, 3E22.14)

n1: a number of nodes defining the meshs of under zone considered.
N2: a number of elements of under zone considered.

2.4
Writing of a concept *Aster* result with format CASTEM 2000

The concepts *Aster* results are written with format CASTEM 2000, in the shape of tables.

The piles necessary to the writing of a table are (in the order and for each sequence number):

.
Crush 27: crush of word type, allows to have the list of the “concepts” available for the table:
variables of access, sequence number, fields to the nodes, fields by element. That gives them
indices “character” of the table, and makes it possible to extract from the table, a reality, an entirety, a field
with the nodes, or a field by element.

U
=
Count
.
1
.
DEPL
It is the word DEPL which appears in pile 27.

· Pile 26: crush entireties; allows to write the value of the sequence number and the value of variables of access of the whole type. Pile 26 appears for each variable of access and for sequence number, and this for all the sequence numbers.

·

Crush 25: crush realities, allows to write the value of the variables of access of the real type. This pile appears for each variable of access and each sequence number.

·

Crush 2: field with the nodes (format describes previously),

·

Crush 39: field by elements (format describes previously),

·

Crush 10: description of the table. Give the occurrence of each object constituting the table in the file with format CASTEM 2000. For example, the field with the nodes with the sequence number 10 is the 50ème object of the field type to the nodes stored in the file.

2.4.1
Crush 27: Crush word type

CRUSH NUMBER 27 NO. OBJECTS NAME 0 NO. iob OBJECTS

iob = a number "of objects" constituting table CASTEM 2000 for a given sequence number. These objects
are identified by a character string of 4 characters.

The objects present are:

·

ORDR, whole value of the sequence number,

·

the first 4 characters of the variables of access (for example: INST,…),

·

the first 4 characters of the name of the size associated with the fields with the nodes and/or with fields by element (for example: DEPL, SIGM,…).

It is thus supposed that numbers it variables of access and fields is identical for all them sequence numbers, and which they are in the same way standard.

Handbook of Reference
D9.07 booklet:
Index A

The following recordings are then present in the file:

--> ncar nmot (2I5)

ncar: total characters = iob*4 numbers
nmot: number of words = iob.

--> character string (A72)

for example ORDRINSTDEPLSIGM for a concept result which has the moment for variable of access, a field with the nodes of the displacement type and a field by element of type forced.

--> (npos) X iob (12I5)

pointers on the preceding character string.

2.4.2
Crush 26: Crush entireties

CRUSH NUMBER 26 NO. OBJECTS NAME 0 NO. iob OBJECTS

.
If writing of the value of the sequence number:

iob
=
2
*
NBORDR

where NBORDR = a number of sequence numbers present in the concept result

Then 2 recordings follow; the first is an entirety which is equal to 2*NBORDR (I5), it
second is a list of entireties (I, value of the ième sequence number), I = 1, NBORDR (7I11).

.
If writing of a variable of access of the whole type:

iob
=
1

Then 2 recordings follow; the first is 1 (I5) and the second the value of the variable
access (2I11).

2.4.3
Crush 25: Crush realities

CRUSH NUMBER 25 NO. OBJECTS NAME 0 NO. iob OBJECTS

This pile makes it possible to write the actual values of the variables of access of the concepts results, and is written
for each variable of access and each sequence number:

iob
=
1

Then 2 recordings follow; the first is 1 (I5) and the second the value of the variable of access
(1X, 1P21.14).

2.4.4
Crush 10: descriptor of the table

CRUSH NUMBER 10 NO. OBJECTS NAME 1 NO. OBJECTS 2

This pile makes it possible to define the structure of the table (which is in fact a table of tables) while giving
position of the piles constituting the table (order of appearance in the file with format CASTEM 2000).

The recordings necessary to the description of the table are:

--> name of the table which is not other than the name of the concept result *Aster* (1X, A8)

--> nbobj (I5)
It is the number of words constituting the pile of the word type.

--> ncar (I5)
It is the total number of characters indicated in the pile of the word type.

--> (27 posmot numpile position) X nbobj (16I5)

27: crush of word type.
posmot: entirety indicating the occurrence in the character string appearing in the pile of word type.
numpile: number of the pile containing information relating to the occurrence posmot (26, 25, 2 or 39).
position: occurrence of the pile in the file with format CASTEM 2000.

For example: Let us suppose that in the pile of the word type, one has ORDRINSTDEPLSIGM.

The table consists of four objects:

.
an object of the whole type (associate with chain ORDR), which is stored in a pile of number 26, which contains the value of the sequence number, and which for the sequence number considered is the ième table of this type,
.
an object of the real type (associate with chain INST) which is stored in a pile of number 25, which contains the value of the moment and which for the sequence number considered is jème table of this type,
.
an object of the field type to the nodes (associate with chain DEPL) which is stored in one crush number 2, which contains the values of the field of displacement in all the nodes grid, and which for the sequence number considered are the kème table of this type,
.
an object of the field type by elements (associate with chain SIGM) which is stored in one crush number 39, which contains the values of the stress field, and which for sequence number considered is the lème table of this type.

27 1 26 I 27 2 25 J 27 3 2 K 27 4 39 L

--> 4 X number order (I5)

--> 26 numéroordre 10 numérotable

Handbook of Reference
D9.07 booklet:
Index A

3
Structure of the universal file

IDEAS

The total structure of a universal file IDEAS is presented in the form of datasets characterized by their number.

Each dataset is framed by the line "- 1":

-
1

n°
dataset

-

-

-

-
1

-
1

n°
dataset

-

-

-

-
1

Each dataset contains a particular type of information (coordinated nodes, connectivities of meshs, results with the nodes,…), and is characterized by a number and a data-processing structure clean.

However, the numbers of dataset and their data-processing structure are not solidified and can vary of a version to another.

Order IMPR_RESU makes it possible to print grids, and results with the nodes or by elements, and this, with the choices of the user, in version 4 or 5 of IDEAS.

So the number of datasets written by order IMPR_RESU is reduced:

Version 4
Version 5

151.151 Titrate
775
775
Properties of the beams
15

Co-ordinates of the nodes in single precision

781
Co-ordinates of the nodes in double precision double precision
71
780
Connectivity of the meshs
752
752
Groups of nodes and meshs
55
55
Results with the nodes (cham_no)
56
56
Results by elements (cham_elem at the point
of Gauss)
57
57
Results with the nodes by elements
(cham_elem with the nodes)

3.1
Dataset 151: Titrate

-
1

151% Title

Vxx.xx.xx aster of the date result of the date-stamp
(A80)

1st
line

titrate
Aster
(A80)

2nd
line

titrate
Aster
(A80)

line
white
(A80)

4th
line

titrate
Aster
(A80)

5th
line

titrate
Aster
(A80)

6th

line

titrate
Aster
(A80)

-
1

This dataset is always the first dataset written in universal files IDEAS generated by order IMPR_RESU.

3.2
Dataset 775: Properties of the beams

This dataset is obligatory when the grid comprises elements of the beam type, and appears then just after the dataset 151.

-
1

775% Properties can section beams

1
0
0
(3I10)

BEAM1

0.
0.
0.
0.
0.
0.
(6 (1PE13.6))

0.
0.
0.
0.
(4 (1PE13.6))

0.
0.
0.
0.
0.
0.
(6 (1PE13.6))

0.
0.
0.
0.
0.
0.
"

0.
0.
0.
0.
0.
0.
"

0.
0.
0.
0.
0.
0.
"

0.
0.
0.
0.
0.
0.
"

0.
0.
0.
0.

0.
0.
"

0.
0.
0.
0.
0.
0.
"

11
7
8
14
1
10
(6I10)

0
45
1
11
1.
(4I10,1PE13.6)

-
1

3.3
Dataset 15: Co-ordinates of the nodes in single precision

If the user requests the writing of a grid from the format universal file IDEAS version 4, them coordinated nodes are written in single precision, in the form of this dataset.

-
1

15% Nodes

N
0
0

11
X
Y
Z
(4I10,3E13.6)

.
.
.
.
.
.
.

.
.
.
.
.
.
.

-
1

N: number of the node (it is the *Aster* number except if the grid were generated by IDEAS, in which case it is number IDEAS).

Following information relates to the definition of the frame of reference which in *Aster* is always
the Cartesian reference mark, from where value 0.

Author (S):
A.M. DONORE

Key: D9.07.01
Page:
16/24

The fourth information indicates the color affected during the posting of the node.

X, Y, Z are the three co-ordinates of the node.

With each node of the grid a line in the dataset 15 corresponds.

3.4
Dataset 781: Co-ordinates of the nodes in double precision double precision

If the user requests the writing of a grid from the format universal file IDEAS version 5, them coordinated nodes are written double precision, in the form of this dataset.

-
1

781% Real*8 Nodes

N
0
0
11
(4I10)

for
each
node

X
Y
Z
(3E25.17)

.
.
.

.

.
.
.

-
1

N: number of the node (it is the *Aster* number except if the grid were generated by IDEAS, in which case it is number IDEAS).

Following information relates to the definition of the frame of reference which in *Aster* is always the Cartesian reference mark, from where value 0.

The fourth information indicates the color affected during the posting of the node.

X, Y, Z are the three co-ordinates of the node.

With each node of the grid two lines in the dataset 781 correspond.

3.5
Dataset 71: Connectivities of the meshs

If the user requests the writing of a grid from the format universal file IDEAS version 4, them connectivities of the meshs are written in the form of this dataset.

-
1

71% Elements

IMAS ICOD1 ICOD2 IPHY IMAT 7 NNOE

(7I10)

NODSUP
(J),
J=1,
NNOE

(8I10)

.

.

.

.

.

.

.

-
1

The two lines of information indicated are written for each element of the grid.

IMAS: Number of the mesh. It is the *Aster* number except if the grid were generated by IDEAS, in which case it is number IDEAS.

ICOD1: Code graphic element.

Code graphic

Net
Nodes
Type
1 Linear
2
Linear
2 Triangle
3
Linear
3 Triangle
6
Quadratic
4 Triangle
9
Cubic
5 Quadrilateral
4
Linear
6 Quadrilateral
8
Quadratic
7 Quadrilateral
12
Cubic
14 Tetrahedron
4
Linear
15 Tetrahedron
10
Quadratic
16 Pentahedron
6
Linear
17 Pentahedron
15
Quadratic
18 Pentahedron
24
Cubic
19 Hexahedron
8
Linear
20 Hexahedron

20
Quadratic
21 Hexahedron
32
Cubic

ICOD2: Descriptor of the finite element

By defect, a value of the descriptor is assigned to each type of mesh. That is done when the user did not specify of *Aster* model and that one thus does not have the knowledge of the type of the finite element.

Type of mesh
Descriptor

POI1
161 (lumped mass)
SEG2
21 (linear beam)
SEG3
24 (parabolic beam)
TRIA3
74
(membrane linear triangle)
TRIA6
72
(membrane parabolic triangle)
TRIA9
73
(membrane cubic triangle)
QUAD4
71
(membrane linear quadrilateral)
QUAD8
75
(membrane parabolic quadrilateral)
QUAD12
76
(membrane cubic quadrilateral)
TETRA4
111

(solid linear tetrahedron)
TETRA10
118
(solid parabolic tetrahedron)
PENTA6
112
(solid linear wedge)
PENTA15
113
(solid parabolic wedge)
HEXA8
115
(solid linear brig)
HEXA20
116
(solid parabolic brig)

When the user provided a name of model, one refines these default values by holding account type of the finite element. The elements concerned are:

MEAXQU4/THAXQU4
--> 84
(Quadrilateral Axi linear)
MEAXQU8/THAXQU8
--> 85
(Quadrilateral Axi parabolic)
MEAXTR3/THAXTR3
--> 81
(Axi linear triangle)
MEAXTR6/THAXTR6
--> 82
(Axi parabolic triangle)
MEDPQU4/THDPQU4
--> 54
(Strain linear Planes quadrilateral)
MEDPQU8/THDPQU8
--> 55
(Strain parabolic Planes quadrilateral)
MEDPTR3/THDPTR3
--> 51
(Strain linear triangle Planes)
MEDPTR6/THDPTR6
--> 52

(Strain parabolic triangle Planes)

MECPQU4/THCPQU4
--> 44
(Planes quadrilateral stress linear)
MEPLQU4/THPLQU4

MECPQU8/THCPQU8
--> 45
(Planes quadrilateral stress parabolic)
MEPLQU8/THPLQU8

MECPTR3/THCPTR3
--> 41
(Stress linear triangle Planes)
MEPLTR3/THPLTR3

MECPTR6/THCPTR6
--> 42
(Stress parabolic triangle Planes)
MEPLTR6/THPLTR6

MEAXSE2/MECPSE2
--> 21
(Linear beam)
MEDPSE2/MEPLSE2

THAXSE2/THCPSE2

THDPSE2/THPLSE2

MEDKQU4/MEDSQU4
--> 94
(thin hull: Quadrilateral TN linear)
MEQ4QU4
MEDKTR3/MEDSTR3
--> 91
(thin hull: TN linear triangle)

IPHY: number of the table of the physical properties = number *Aster* associated with the type of mesh or it
aster number associated with the type with the finite element if a model were specified by the user.

IMAT: number of the table of the characteristics material = 1 except for the meshs reduced to a point in which case IMAT is worth 2.

The following recording indicates the color of the element during its posting in IDEAS (by defect 7 green color).

NNOE: a number of nodes defining the mesh.

NODSUP (J), J=1, NNOE: list numbers of nodes composing the mesh.

Note:

The Aster meshs not existing in IDEAS are ignored by interface (QUAD9, HEXA27).

3.6
Dataset 780: Connectivities of the meshs

If the user requests the writing of a grid from the format universal file IDEAS version 5, them connectivities of the meshs are written in the form of this dataset.

-
1

780% Elements

IMAS
ICOD2
1
IPHY
1
IMAT
7
NNOE
(8I10)

NODSUP (J),
J=1,
NNOE (8I10)

% if meshs linear

IMAS
ICOD2
1
IPHY
1
IMAT
7
NNOE
(8I10)

0
1
1
1
1
(5I10)

NODSUP (J),
J=1,
NNOE (8I10)

IMAS: number of the mesh. It is the *Aster* number except if the grid were generated by IDEAS, in which case it is number IDEAS.

ICOD2: descriptor of the finite element (see description dataset 71).

IPHY: number of the table of the physical properties (see description dataset 71).
IMAT: number of the table of the characteristics material (see description dataset 71).

The following recording indicates the color of the element during its posting in IDEAS (by defect 7 green color).

NNOE: a number of nodes defining the mesh.

Note:

To write an element of beam, there is an additional line in the dataset 780. This line defines the characteristics of the beam, inter alia, the number of the node being useful for orientation of the principal directions of the beam. Values written by the order IMPR_RESU are values can.

The Aster meshs not existing in IDEAS are ignored (QUAD9, HEXA27).

3.7
Dataset 752: Groups of nodes and meshs

-
1

752% Groups

NUM
0
0
0
0
NO.
(6I10)

NAME
(20A2)

(ICOD,
MUMENT)
I=1,
NO.
(8I10)

.
.
.
.
.
.

.

.
.

-
1

For each group of nodes or meshs *Aster*, one writes the instructions indicated previously.

NUM: Number of the group. It is sequential; one starts with the groups of nodes in the order of their appearance in *Aster*.

NO.: A number of nodes or meshs composing the group.

NAME: Name of the group. It is the name *Aster*.

ICOD: It is a code indicating the type of the entity the following. 7 indicates that the number which follows it is
that of a node, 8 indicates that the number which follows it is that of a mesh.

NUMENT: Number of the entity (number of a node or a mesh).

Note:

The third line is as many repeated once as necessary to write all the numbers of nodes or of meshs composing the group.

3.8 Writing of the *Aster* results: CHAM_GD or Concept RESULT

The three datasets being used to write the *Aster* results are the datasets 55 (for fields with nodes), 56 (for fields by elements at the points of Gauss) and 57 (for fields by elements with the nodes).

3.8.1
Dataset 55: Values with the nodes

MODTYP: type of the model
MODTYP = 1 Structural

MODTYP = 2 Heat transfer

Order IMPR_RESU takes MODTYP = 1 except when the size associated with the field with to print is TEMP or FLOW, in which case MODTYP = 2.

ANATYP: type of analysis

ANATYP =0
UNKNOW
Value taken by defect by the order
IMPR_RESU
=1
STATIC Value taken during the impression of fields of
size named in *Aster*
=2
NORMAL
MODE
Value taken during the impression of a concept
result having for variable of access NUME_MODE
=4
TRANSIENT
Value taken during the impression of a concept
result having for variable of access INST
=5
FREQUENCY
Value taken during the impression of a concept
RESPONSE
result having for variable of access FREQ and
not NUME_MODE

Handbook of Reference
D9.07 booklet:
Index A

DATCAR = characteristic of the data

=1
scalar
=3
vector with 6 degrees of freedom (3 translations and 3 rotations)
=4
symmetrical tensor

DATTYP = standard of the data

=0
unknown
=2
constraints
=3
deformations
=5
temperature
=6
flow
=8
displacement
=11
speed
=12
acceleration

=15
pressure

TYPE = 2 for actual values, 5 for complex values.

NO. = a number of values to be printed by node.

NUMOR = sequence number *Aster* for the concepts results.

IMODE = value of the variable of access NUME_MODE for the sequence number considered (for concepts result having this variable of access).

FREQ = value of the variable of access FREQ for the sequence number considered (for the concepts result having FREQ for variable of access).

MASGEN = value of parameter MASSE_GENE for the sequence number considered (for the concepts results having NUME_MODE for variable of access).

AMOR1 = value of parameter AMOR_REDUIT for the sequence number considered (for the concepts result having NUME_MODE for variable of access).

AMOR2 = 0.D0

INST = value of the variable of access INST for the sequence number considered (for the concepts result having INST for variable of access).

NO. = a number of values of the result by node.
NUMNOE = number of the node.
VALE = values of the result (cham_no) to the node considered.

3.8.2 Dataset 56: Values by elements

The heading of this dataset is identical to that of the dataset 55.

NUMMAI = number of the mesh.
NBVAL = a number of values on the mesh.
VALE = values of the result (cham_elem at the points of Gauss) on the mesh considered.
cham_elem at the points of Gauss *Aster* is written with format IDEAS in the form of constant fields by element (average of the values at the points of Gauss).

3.8.3 Dataset 57: Values with the nodes by element

The heading of this dataset is identical to that of the dataset 55 and 56.

NUMMAI = number of the mesh.
NBNOE = a number of nodes of the mesh.
NBVAL = a number of values per node.
VALEJ = values of the result (cham_elem to the nodes) on the node J of the mesh considered.

3.9
Writing rule of a dataset of results

In *Code_Aster*, the fields of size are fields with the nodes or fields by
element at the points of Gauss or the nodes.
The fields with the nodes are written in the form of datasets number 55, the fields by elements with points of Gauss in the form of datasets number 56 and fields by element with the nodes under
form datasets 57.

For the concepts result, one treats sequence number per sequence number and reference symbol by name symbolic system, various fields of size composing the concept result.

The only difference in the writing of the datasets 55, 56 and 57 is the value of the code corresponds to the type
from analysis (ANATYP) which are worth 1 for the fields of size named in *Aster*, and 2, 4 or 5 for concepts results (see description dataset 55).

Handbook of Reference
D9.07 booklet:
Index A

In *Code_Aster*, a field of size is associated a size of which the list of the components
possible is defined in a catalogue. The various components can be vectors,
tensors or of the scalars.

In IDEAS, a dataset includes/understands to the maximum 6 components, and these components are typified:

·
vectors with 6 components of the displacement type, speed, acceleration, flow or unknown,
·
symmetrical tensors of type forced or deformations,
·
scalars of the temperature type or pressure.

Also, one associates an *Aster* size a finished number of datasets IDEAS likely to be written (if
the components really exist on the model). It is the IRGAGS.f routine which carries out this work.

Let us consider for example the size DEPL_R whose components are DX, DY, DZ, DRX,
DRY, DRZ, GRX, DDZDN, NEAR, PHI.

With a field with the nodes associated with this size to the maximum three datasets IDEAS (all corresponds
three of number 55):

·
a dataset of the vector type to 6 components for the writing of DX, DY,… DRZ,
·

a dataset of the scalar type for the writing of CLOSE,
·
a dataset of the unknown type for the writing of components GRX, DDZDN, PHI.

These datasets can then exist or not according to the presence or not components. The dataset exist as soon as one of the components the component is defined on a node of the grid (in which case the value of the components absent is 0.).

With regard to the internal variables, one generates an one or more (dataset includes/understands with maximum 6 components) datasets of number 55 and type “UNKNOWN”.

4
Structure of the file

ENSIGHT

The description of format ENSIGHT will be integrated into a forthcoming version of this document.

5 Bibliography

[1]
R. GALON, H. BUNG. Report/ratio DMT/91.127. Program Plexus: exits on files for postprocessing by CASTEM 2000. 04/03/1991.

[2]
Mr. FARVACQUE. Report/ratio DEMT/86/411. CASTEM 2000: Data-processing note of operators SAUVER-RESTITUER. 28/10/1986.

[3]
D. RICHARD. Structure of the backup file of CASTEM 2000. Report/ratio CSA/92.022. 31/08/1992.

[4]
User' S Guides “Core utilities”. I-DEAS. SDRC.

Titrate:
Data-processing description of LIRE_RESU

Date:
31/08/01
Author (S):
***J. PELLET**, F. LEBOUVIER Key*
:
D9.07.02-A Page
: 1/6

Organization (S): EDF/MTI/MMN, DeltaCAD

Data-processing handbook of Description
D9.07 booklet:
D9.07.02 document

Data-processing description of LIRE_RESU

1 Goal

This document is a data-processing description of the order LIRE_RESU, whose role is lira of the fields
with the nodes or fields by elements on a file with format "IDEAS", "ENSIGHT".

One finds:

· the list of the principal routines used by the order, as well as a short summary of theirs functionalities.
· routines to modify to allow the addition of a new format of reading,
· routines to be modified for the addition of new types of results as well as new fields
within the framework of format IDEAS.
Data-processing handbook of Description
D9.07 booklet:
HI-75/01/005/A

Code_Aster ®
Version
6.0

Titrate:
Data-processing description of LIRE_RESU

Date:
31/08/01
Author (S):
J. PELLET, F. LEBOUVIER Key
:
D9.07.02-A Page
: 2/6

Count

matters

2
Tree of call of the principal routines of LIRE_RESU

OP0150.f
IDEAS
ENSIGHT
MED
LRIDEA.f
CRSDFI.f
RSUTC2.f
DECOD1.f
RSUTC3.f
DECOD2.f
NUMEOK.f
TRFMOT.f
STOCK.f

Appear 2-a: OP0150.f

3 Description of the routines observers in the tree of call

OP0150.f
Main program of order LIRE_RESU. This routine treats in-house the format ENSIGHT. Format MED is not available.

LRIDEA.f
Main program of reading of the files of results to the universal format "unv"

RSUTC2.f
This routine makes it possible starting from the name of the field to determine the name of the size ("DEPL_R") associated and the type of the SD results ("NOEU", "ELNO", "ELGA")

RSUTC3.f
This routine makes it possible to determine the access mode "FREQ" or "INST" according to type of results

CRSDFI.f
Creation and initialization of SD FORMAT_IDEAS, it allows identified in the file universal "unv" the dataset which will be read. This SD is described in detail in the document [D4.02.xx].

NUMEOK.f
This routine checks if the sequence number, the moment or the frequency read in the dataset corresponds to that or that required.

DECOD1.f
From the information contained in the SD FORMAT_IDEAS, this routine checks if the heading of the dataset read, corresponds to that required.

DECOD2.f
From the information contained in the SD FORMAT_IDEAS, this routine extracts it sequence number, the moment or the frequency

STOCK.f
This routine stores the results contained in the simple field (cham_no, cham_elem)
in the SD results.
Data-processing handbook of Description
D9.07 booklet:
HI-75/01/005/A

4
Flow chart of lridea

BEGINNING
Creation of the SD
CRSDFI.f
&&LIRE_RESU_IDEA
Reading of the heading of the dataset
Buckle on
names
With
of field
Values
yes
DECOD1.f
waited
Buckle on
recordings
not

B
Presence:
frequency
not
DECOD2.f
moment
order
yes
C
Nume_Ordre,
One passes to
not
moment,
NUMEOK.f
next dataset
fréqence
yes
One stores the values in
the simple field
One stores the values in
STOCK.f
the SD results
not
End of
file unv
yes
END

Appear 4-a: Flow chart of lridea.f
Note:

· A: one checks if the values read in the heading of the dataset correspond to those awaited.
· B: one checks if there exists at the place indicated by SD FORMAT_IDEAS of information of the types
whole or real to identify the sequence number, or the moment or the frequency.
· C: it is checked if the sequence number, or the moment or the frequency corresponds has that or that waited.
Data-processing handbook of Description
D9.07 booklet:
HI-75/01/005/A

Code_Aster ®
Version
6.0

Titrate:
Data-processing description of LIRE_RESU

Date:
31/08/01
Author (S):
J. PELLET, F. LEBOUVIER Key
:
D9.07.02-A Page
: 5/6

5 Evolutions

In this paragraph one presents work to be undertaken for the addition of new functionalities.
One specifies for each evolution the files with modified and the nature of the modifications.

5.1
New format results

The addition of a new format results (ex: "MED") implies the modification of the following files:

· op0150.f:
a structure "IF THEN ELSE" allows the introduction of new
format,
· lire_resu.cata:
addition of the new type of format.

5.2 Format
IDEAS

5.2.1 New type of results

The addition of a new type of results ("DYNA_TRANS", "EVOL_THME") implies the modification
of
following files:

· lire_resu.cata: addition of the new type of results,
· rsutc3.f:
definition of the access mode ("INST", "FREQ") associated new
type of results.

5.2.2 New field

The addition of a new type of results ("DYNA_TRANS", "EVOL_THME") implies the modification of following files:

· lire_resu.cata:
addition of the new field,
· crsdfi.f:
definition of the characteristics of the new field, the SD FORMAT_IDEAS created and initialized in this routine is defined in the document [D4.02.06],
· rsutc2.f:
definition of the name of the size ("DEPL_R", "TEMP_R") and of the type SD ("NOEU", "ELNO", "ELGA") associated the new field.

Intentionally white left page.
Data-processing handbook of Description
D9.07 booklet:
HI-75/01/005/A

Code_Aster ®
Version
7.4

Titrate:
Description of the format of files GIBI

Date
:

01/12/04
Author (S):
Key COURTEOUS Mr.
:
D9.07.03-B Page
: 1/6

Organization (S): EDF-R & D /AMA

Data-processing handbook of Description
D9.07 booklet: -
Document: D9.07.03

Description of the format of files GIBI

Summary:

The first part of the document describes the way in which the file of grid produced by GIBI is indicated by
the operator TO SAVE, option FORMAT. It corresponds to version 2000 of GIBI (located by level 11 in
output file).
On a simple example of a grid, one describes the lines of the file one by one.

The second part describes the contents of the pile of the fields by elements such as must write it IMPR_RESU with
format "CASTEM".

Data-processing handbook of Description
D9.07 booklet: - HT-66/04/003/A

Code_Aster ®
Version
7.4

Titrate:
Description of the format of files GIBI

Date
:

01/12/04
Author (S):
Key COURTEOUS Mr.
:
D9.07.03-B Page
: 2/6

1 File
grid

1.1 Example
used

The description of the file of grid produced by GIBI is made starting from the following example:

opti dime 2 elem qua4;
Pa = 0 0; Pb = 1 0; liab = Pa droi 3 Pb;
known = trans liab 2 (0 1);
ens= liab and known;
opti sauv format "mon.fic";
sauv format liab ens;
end;

In the file result, one inserts comments to explain what the lines contain of
file right afterwards.

1.2
Description of each line

Beginning of the file "mon.fic"
First package of which the number of lines does not vary. One finds there indications general.
RECORDING OF THE TYPE 4
LEVEL 11 LEVEL ERROR 0 DIMENSION 2
DENSITY .00000E+00
RECORDING OF THE TYPE 7
INFORMATION CASTEM2000 8 NUMBERS
IFOUR -1 NIFOUR 0 IFOMOD -1 IECHO 1 IIMPI 0 IOSPI 0 ISOTYP 1
NSDPGE 0

Second package which defines all the piles (a pile by type of object and certain piles in more).
A recording of the type 2 prevents writing of a new pile, that of type 5 prevents end.
RECORDING OF THE TYPE 2

CRUSH NUMBER 1NBRE OBJECTS NAME 3NBRE OBJECTS 6
The pile number 1 is that of the objects of the grid type. The following line gives the name of the objects
saved grids.
LIAB KNOWN ENS
The following line gives the sequence numbers, in the pile, of the named objects quoted previously.
In our case LIAB is the first, KNOWN is the third and ENS is the second.
(valid for all the lines which follows to the next pile)
1 3 2

Passage to the description of the objects ones after the others.
Description of the first object:
The first recording of each object is composed of 5 numbers representing:
ITYPEL: type of the element 1=point, 2=segment with two nodes?
NBSOUS: numbers of under parts in this object, under part by type of elements the component.
NBREF: numbers of under references. A reference is for example contour
NBNOEL: a number of nodes per element
NBEL: a number of elements
If ITYPEL=0 then NBSOUS different from zero. In this case one will read the list of the positions, in the pile
objects, under parts the component.
If NBSOUS=0, NBNOEL and NBEL are different from zero, if need be, the list of the references is found,
numbers of the colors then connectivities.
Here 3 elements with 2 nodes of segment to 2 nodes
2 0 0 2 3
As NBREF=0 one passes to the recording giving the number of the color of the elements.
0 0 0

:
D9.07.03-B Page
: 3/6

Table of connectivities. Description of the first element then second…
ATTENTION it is not a question of true classification, it is necessary to make it pass by the filter of the last
table of the pile number 32. Thus element 1 is formed of nodes 1 and, 3 element 2 is made of 3 and 4 and element 3 is formed of nodes 4 and 2.
1 2 2 3 3 4
Description of the second grid object
ITYPEL=0 thus grid complexes made up of 2 pennies left
0 2 0 0 0
These under-parts are represented by the objects grid 1 and 3 of this pile
1 3
Description of the third object of the pile
ITYPEL=8 NBSOUS=0 thus grid of 6 element with 4 nodes. There are 4 references (here the sides of KNOWN rectangle).
8 0 4 4 6
The list of the grids objects follows representing the references
1 4 5 6
The list of the numbers of the colors follows
0 0 0 0 0 0
Follows the list of connectivities (numbers of nodes per element) not to forget to pass it to the filter of the last list of pile 32.
1 2 5 6 2 3 7 5 3 4
8 7 6 5 9 10 5 7 11 9
7 8 12 11
Description of the fourth object: It is the second reference of the third object of the pile
2 0 0 2 2
0 0
4 8 8 12
Fifth object
2 0 0 2 3
0 0 0
12 11 11 9 9 10
Sixth object
2 0 0 2 2
0 0
10 6 6 1
End of the pile number 1 and beginning of pile 32 (that of the points)
RECORDING OF THE TYPE 2

CRUSH NUMBER 32NBRE OBJECTS NAME 2NBRE OBJECTS 12
List names of points
PA PB
The list of the numbers of the named points Pa follows = 1 PB = 4
1 4
the number of nodes follows
12
The following table gives the filter to have the true number of the nodes belonging to the elements described. For example, if an element, described in pile 1, refers to a number of node equal to 5 it should be put equal to 12
1 3 4 2 12 10 13 11 7 6
8 9
End of pile 32, beginning of pile 33 (that of the configurations (coordinated))
RECORDING OF THE TYPE 2
CRUSH NUMBER 33NBRE OBJECTS NAME 0NBRE OBJECTS 1
The number of points follows which one gives the co-ordinates
39
Data-processing handbook of Description
D9.07 booklet: - HT-66/04/003/A

The co-ordinates are given by nodes. Initially the first then the second…
For each nodes, one gives the 2 or 3 co-ordinates plus the current density to the moment of its creation thus here 3 values per node.
0.00000000000000E+00 0.00000000000000E+00 0.00000000000000E+00
1.00000000000000E+00 0.00000000000000E+00 0.00000000000000E+00
3.33333333333333E-01 0.00000000000000E+00 3.33333333333333E-01
6.66666666666667E-01 0.00000000000000E+00 3.33333333333333E-01
0.00000000000000E+00 1.00000000000000E+00 0.00000000000000E+00
0.00000000000000E+00 1.00000000000000E+00 0.00000000000000E+00
3.33333333333333E-01 1.00000000000000E+00 3.33333333333333E-01
6.66666666666667E-01 1.00000000000000E+00 3.33333333333333E-01
1.00000000000000E+00 1.00000000000000E+00 0.00000000000000E+00
0.00000000000000E+00 5.00000000000000E-01 5.00000000000000E-01
1.00000000000000E+00 5.00000000000000E-01 5.00000000000000E-01
3.33333333333333E-01 5.00000000000000E-01 5.00000000000000E-01
6.66666666666667E-01 5.00000000000000E-01 5.00000000000000E-01
RECORDING OF THE TYPE 5
AUTOMATIC LABEL: 1
End of the file.

2
Crush fields by elements

One describes here the pile of the fields by elements (pile number 39) such as must produce it the order IMPR_RESU.

The pile number 39 corresponds to the fields by elements (MCHAML in Castem).
CRUSH NUMBER 39NBRE OBJECTS NAME 0NBRE OBJECT 1

This line specifies the number of subfields (one by elementary grid, noted N1), the mode of calculation (- 2 plane constraints, -1 plane deformations, 0 axisymmetric, 1 Fourier series, 2 three-dimensional), a number of information further provided (noted N3), length of the title
5 2 4 16

Titrate field by element
CONSTRAINTS

Block of N1x (3+N3) whole (here 5x7): pointer towards the grid support of the subfield, not used, component count of the field in the subfield, N3 information (dependence with respect to locate, not used, number of the harmonic of Fourier, values defined in the nodes or elsewhere…).
215 0 6 0 0 0 0 218 0 6
0 0 0 0 219 0 6 0 0 0
0 220 0 6 0 0 0 0 221 0
6 0 0 0 0

Names of the components of each subfield (in general, Aster does not write anything, like the format of reading is 8 (1X, A8) and that for the reading Castem two values to read a name of component, it is necessary
*thus [(N1*2-1) /8 + 1] blank lines!).*
[blank line]
[blank line]

There is then N1 blocks, one by subfield:

Values not used: as many 0 as components in this zone.
0 0 0 0 0 0
Data-processing handbook of Description
D9.07 booklet: - HT-66/04/003/A

Code_Aster ®
Version
7.4

Titrate:
Description of the format of files GIBI

Date
:

01/12/04
Author (S):
*Key **COURTEOUS Mr.***
:
D9.07.03-B Page
: 5/6

Name of the components (K8).
NR VY VZ MT MFY MFZ

Types of the components (two K8 by type separated by a white).
REAL*8 REAL*8 REAL*8 REAL*8
REAL*8 REAL*8

For the component n°1: a number of points per element, a number of element, used, used.
2 1 0 0

Value with node 1 of element 1, value with node 2 of element 1, etc
3.6379788070917E-012 3.6379788070917E-012

Idem for component 2.
2 1 0 0
-2.1684043449710E-018 -2.1684043449710E-018

Idem for component 3.
2 1 0 0
-1.0658141036402E-014 -1.0658141036402E-014

Idem for component 4.
2 1 0 0
0.0000000000000E+000 0.0000000000000E+000

Idem for component 5.
2 1 0 0
1.6653345369377E-015 -2.2204460492503E-015

Idem for component 6.
2 1 0 0
-2.7105054312138E-020 -3.2526065174565E-019

Block for subfield 2: the names of the components are different, one has 4 values for each one of 59274 elements…
0 0 0 0 0 0
SMXX SMYY SMZZ SMXY SMXZ SMYZ

REAL*8 REAL*8 REAL*8 REAL*8
REAL*8 REAL*8
4 59274 0 0
-5.0599133238301E+006 -5.0599133238300E+006 -5.0599133238300E+006
-5.0599133238300E+006 -2.6477329870372E+007 -2.6477329870372E+007
-2.6477329870372E+007 -2.6477329870372E+007 1.2433887828476E+005
1.2433887828476E+005 1.2433887828476E+005 1.2433887828476E+005
...

Intentionally white left page.
Data-processing handbook of Description
D9.07 booklet: - HT-66/04/003/A
