Procedure BEGINNING

1 Goal

To allocate the resources memory, disc and files.

The execution consists of a set of orders starting with BEGINNING and ending in END [U4.11.02], (see also the procedure CONTINUATION [U4.11.03]).

The order BEGINNING who is carried out, as of his reading by the Supervisor, carries out the following tasks:

- definition of the characteristics of the databases (managed by JEVEUX) and allowance of the associated files,
- reading of the catalogues of the elements and the orders.

The apparently complex syntax of this procedure should not worry the user; its call with the operands by default, sufficient in most case, is: BEGINNING ()

The operands are to be used studies in the case of requiring a more important size of the files “databases” or to divert the various files on numbers of logical unit different from the affected numbers by default.

Orders placed front BEGINNING, if they are syntactically correct, are ignored.

Contents

2 Syntax

```
BEGINNING
      (  ◊  PAR_LOT      =  /  'YES',                          [DEFECT]
                            /  'NOT',
         ◊  IMPR_MACRO   =  /  'NOT',                          [DEFECT]
                            /  'YES',

         ◊  LANG = Lang,                                       [TXM]

         ◊  BASE = _F (     ♦  FILE = /  'TOTAL',
                                      /  'VOLATILE',
                                      /  |LONG_ENRE = lenr,    [I]
                                         |NMAX_ENRE = nenr,    [I]
                                         |LONG_REPE = lrep,    [I]
                     ),

         ◊  CODE = _F  (  ♦  NIV_PUB_WEB     =  / 'Internet',
                                                / 'Intranet',
                        ),

         ◊  ERROR =   _F  (ERREUR_F=      /  'ABORT',          [DEFECT]
                                          /  'EXCEPTION',
                           ),
         ◊  IGNORE_ALARM =  l_vale,                            [l_KN]

         ◊  DEBUG =      _F  (  JXVERI =  /  'YES',
                                          /  'NOT',
                                   ENVIMA =    'TEST',         [l_KN]
                                ◊  JEVEUX =  /  'YES',
                                          /  'NOT',
                                ◊  SDVERI =  /  'YES',
                                             'NOT',
                                ◊  HIST_ETAPE = /  'NOT',
                                                   'YES',
                                   ),
         ◊  MESURE_TEMPS = _F (
                                ◊  NIVE_DETAIL =/ 0            [DEFECT]
                                                / 1
                                                / 2
                                                / 3
                                ◊  AVERAGE =    / 'NOT'        [DEFECT]
                                                / 'YES'
                                   ),

         ◊  MEMORY =  _F (  ◊  TAILLE_BLOC     =  /  800. ,
   [DEFECT]
                                                    /  tbloc,  [R]
                     ),

         ◊  CATALOGUE  = _F (  ♦ FILE    = nfic,               [l_Kn]
                               ◊ UNIT    = unit,               [I]
                        ),

         ◊  RESERVE_CPU =_F (  / VALE       =  vale            [R]
                               / PERCENTAGE = pcent            [R]

                                  ◊ LIMIT   =  /  bv,          [R]
                                               /  900.         [DEFECT]
                           ),
)
```

3 Postings

At the beginning of the execution of Code_Aster, a heading is displayed. One finds there:
- the identification specifies version used: number of version, date of the last modifications,
- the date and the hour of the beginning of the execution,
- the name, architecture, the operating system of the machine,
- the language used for the posting of the messages,
- the type of parallelism available (MPI/OpenMP), the number of allocated processors,
- the version of the libraries used (when it is available) for hdf5, med, mumps, Scotch tape,
- then several information on the distribution of the memory.
  For example:

```
Memory limits for the execution: 256.00 Mo
consumed by initialization: 148.68 Mo
by the objects of the command set: 17.48 Mo
remain for the dynamic allocation: 89.84 Mo
Size limits files of exchange: 48.00 Go
```

What means:
- `256 Mo` is the quantity of memory requested by the user, it is the total quantity which one should not exceed.
- `148.68 Mo` is consumed simply by starting the execution (loading of achievable, the associated dynamic libraries, etc).
- `17.48 Mo` is consumed by the reading of the command file (Note:: in mode PAR_LOT=' NON', the command set being read progressively, this value will be then worthless).
- `89.84 Mo` is the quantity of memory available (at this moment) for the objects of calculation (equalizes to 256-148.68-17.48). It is thus seen that calculation cannot begin if this value is too low.

During the execution, according to the dynamic allocations carried out, when this value varies of more than 10 % (upwards or downwards), a message of this type informs the user:

```
The memory currently consumed except JEVEUX is of 214.08 Mo.
The limit of dynamic allocation JEVEUX is fixed at 41.92 Mo.
```

At the end of the execution, an assessment indicates if same calculation can be started again with less memory:

```
The memory requested from launching is over-estimated, it is of 256 Mo.
The peak report used is of 216.02 Mo.
```

or so more memory is necessary (indeed according to the platforms, the maximum limit can be exceeded without the system stopping calculation):

```
The memory requested from launching is underestimated, it is of 256 Mo.
The peak report used is of 273.22 Mo.
```

4 Operands

4.1 Operand PAR_LOT

PAR_LOT =

Mode of treatment of the orders:

'YES': (option by default); the supervisor analyzes **all** orders before asking the execution of it.

'NOT': after having analyzed **one** order the supervisor requires his execution then passes to the analysis (and the execution) of the following order (treatment orders by order).

4.2 Keyword IMPR_MACRO

IMPR_MACRO =

Authorize or not the postings produced by the macros in the file of message. The reading of the files of message can be painful when it contains the totality of the echoes of the subcommands generated by macro itself. By default, only the echo of the orders explicitly called by the user in his command set will appear.

4.3 Keyword LANG

It makes it possible to choose the language of posting of the messages transmitted by the code.
If the keyword is not indicated, they are the variables of environment which determines the language of the messages (reference: http://www.gnu.org/software/gettext/manual/gettext.html#Users). One can for example define in the file ~/.bashrc : LANG=fr_FR.UTF-8 export.
The encoding (UTF-8 or ISO-8859-1) allows to correctly display the accentuated characters.
The keyword LANG a value in two letters waits, for example 'FR' (for French) or 'IN' (for English).
When a language is selected (that it is by the environment or LANG), still it is necessary that the file of the translated messages (file .mo) that is to say available. This file is expected under this name:

$ASTER_ROOT/share/locale/'Lang '/LC_MESSAGES/aster_ 'version '.mo

where $ASTER_ROOT is the principal repertoire of *Code_Aster* (e.g.: /aster or /opt/aster), Lang is the name in small letters of the language (e.g. in, Fr, of...) and version is the name of the version of *Code_Aster* used (e.g. stable, testing, unstable).
If the file of translation cannot be read, it is French who is used.

Notice

Even if the file of translation exists, when a message was not translated, there is displayed in French (language used of the messages in the source code).

4.4 Keyword BASE

BASE =

The functionality of this keyword is to redefine the values of the parameters of the files of direct access associated with the "databases" if one does not wish to use those fixed by default.

Values by default of the parameters associated with the databases.

TOTAL		
NMAX_ENRE	62914	
LONG_ENRE	100	Kmots
LONG_REPE	2000	
BIRD		
NMAX_ENRE	62914	
LONG_ENRE	100	Kmots
LONG_REPE	2000	

The word is worth 8 bytes out of platform 64 bits under LINUX 64, TRU64 and IRIX 64.4 bytes out of platform 32 bits under SOLARIS, HP-UX and WINDOWS - NT, LINUX.

Under Linux 64, with the values by default, procedure BEGINNING a file of direct access of at the maximum will allocate 62914 recordings of 100 Kmots (it K is worth 1024) for the base 'TOTAL'.

Note:

The real size of the file is dynamic; it depends on the volume of information to store indeed. But this size is limited by the conditions of operating and a parameter preset among the values characterizing the platform. On the platform of reference Linux 64 bits, the maximum size is fixed at 48 Go. This value can be modified while passing an argument on command line of achievable behind the keyword "- max_base size" where size is an actual value measured out of Mo.

On the platforms 32 bits, the maximum size is fixed at 2,047 Go (2 147,483,647), but the code manages several files to go beyond this limit when the parameter "- max_base" passed in argument.
For the Total base, which can be saved and re-used in data of a calculation, the maximum size in "CONTINUATION" is preserved such as it is if the parameter "- max_base" is not used, but perhaps redefined with the need for this manner.

4.4.1 Operand FILE

♦ FILE =

Reference symbol of the base considered.

4.4.2 Operands LONG_ENRE/NMAX_ENRE/LONG_REPE

Definition of the parameters of the database (files of direct access).

/ | LONG_ENRE = lenr

lenr is the length of the recordings in Kmots of the files of direct accesses used.

Note:

The manager of memory JEVEUX uses this parameter to determine two types of objects: the large objects which will be cut out in as many recordings as necessary, and the small objects which will be accumulated in a plug of the size of a recording before being discharged.

| NMAX_ENRE = nenr

nenr is the number of recordings by default, this value is given from LONG_ENRE and of an operating parameter on the platform of reference Linux 64 fixed at 48 Go (51 539 607 552 bytes) for the maximum size of the file associated with a database, if this value were not modified by the use of the keyword – max_base on the command line of the achievable one.

Note:

Two operands LONG_ENRE and NMAX_ENRE must be used with precaution, a bad use which can lead to the stop brutal of the program by saturation of the files of direct access. Coherence enters maximum size of the file and the value resulting from the product of the two parameters LONG_ENRE and NMAX _ENRE is checked at the beginning of execution.

| LONG_REPE = lrep

lrep is the initial length of the repertoire (maximum number of addressable objects by JEVEUX), it is managed dynamically by the manager of memory which extends the size of the repertoire and all the associated system objects as needs.

Note:

The choice by the user to modify these various parameters determines in a final way certain characteristics of the base TOTAL who cannot be modified any more in CONTINUATION.

4.5 Keyword CODE

CODE =

This keyword is intended only for the command files of the tests of nonregression managed with the source code.
The presence of this keyword positions the mode of deboggage automatically DEBUG (JXVERI=' OUI',) who implements checks on objects JEVEUX, which can bring a overcost to the execution. The behavior in the event of error can be modified.

4.5.1 Operand NIV_PUB_WEB

♦ NIV_PUB_WEB = 'INTRANET'

Level gauge of publication. Meaning that the test is only diffusable on the internal network.

NIV_PUB_WEB = 'INTERNET'

Indicate that the test is diffusable such as it is on the external network.

4.6 Keyword ERROR

Allows to modify the behavior of the code in the event of <F> error.

4.6.1 Operand ERREUR_F

In the event of error, the code stops the normal execution of the command set.
By default, an exception is then raised (for the detailed definition of an exception Python, one will refer to the documentation of Python or that of the supervisor, *cf.* [U1.03.01]). In this case, the code carries out the order END (*cf.* [U4.11.02]) which closes the base then in order to allow the possible continuation of calculation. It will be noticed that, although the initial error is known as "fatal" (<F>), the diagnosis is <S>_ERROR since the exception "is recovered" by END. This base will be then recopied by the manager of studies. This is the behavior when ERREUR_F=' EXCEPTION'.

If ERREUR_F=' ABORT', that means that one explicitly asks the code definitively to stop the execution of the command set in the event of fatal error (<F>). The order END is not carried out, the base is thus not closed correctly, it is not recopied and no resumption of calculation is possible.

Remarks

For the execution of the CAS-tests by the developers, the stop by ABORT is automatic and by default. This is activated by the presence of the keyword factor CODE (except if ERREUR_F specify another thing).

In the event of lack of time CPU, of memory, for all errors of the type <S> and the exceptions, the behavior that is described when ERREUR_F=' EXCEPTION' .

4.7 Keyword IGNORE_ALARM

IGNORE_ALARM =

Allows the user to remove the posting of certain alarms (of which he knows the origin) in order to more easily identify other alarms which could to appear.
During the execution of the order END, one systematically displays a summary table of the alarms emitted during the execution (and the number of occurrences). The alarms ignored by the user are preceded by '*' them to distinguish (and they appear even if they were not emitted).
Alarms are indicated starting from the nomenclature appearing between the characters < and >, for example: IGNORE_ALARME = ('MED_2', 'SUPERVIS_40',…).

4.8 Keyword DEBUG

DEBUG =

Option of déboggage (reserved for the developers and the maintenance of the code).

4.8.1 Operand JXVERI

JXVERI =

Allows to control the integrity of the segments of the memory between two executions of consecutive orders. By default the execution is carried out without "DEBUG". This option is systematically activated in the presence of the keyword CODE.

4.8.2 Operand ENVIMA

ENVIMA = 'TEST'

Allows to print in the file RESULT values of the parameters preset in the software package ENVIMA characterizing the machine [D6.01.01].

4.8.3 Operand JEVEUX

◊JEVEUX =

Allows to activate the operating process in debug of the manager of memory JEVEUX: unloadings on disc not differed and assignment from the segments values to an indefinite value [D6.02.01].

4.8.4 Operand SDVERI

SDVERI = 'NOT'

The use of this keyword is bound for the developers. Attention, this functionality little to cause a considerable overcost during the execution.
This keyword starts the checking of the structures of data produced by the operators. It is used within the framework as of procedures of development of the code in the tests of nonregression. If the keyword CODE is present, this keyword takes the value by default 'YES'.

4.8.5 Operand HIST_ETAPE

HIST_ETAPE = 'NOT'

This keyword makes it possible to preserve all the history of the stages/orders used. This is greedy in memory and must be used only for quite particular cases (the orders which require it indicates it in their documentation).
By default, this history is not preserved.

4.9 Keyword MESURE_TEMPS

The keyword MESURE_TEMPS allows to choose the level of detail of the impressions of times CPU which will be displayed in the file of messages by the orders carrying out of elementary calculations, of the resolutions of systems linear, the unloading of objects on disc or communications MPI.

4.9.1 Operand NIVE_DETAIL

By default, at the end of each order, one will print a line of the type:

```
#1.Resolution.des.systemes.lineaires         CPU. (USER+SYST/SYST/ELAPS):  7.52 0.79 11.22
# 2.Calculs.elementaires.et.as semblages       CPU. (USER+SYST/SYST/ELAPS): 15.07 0.70 15.77
```

◊ NIVE_DETAIL = 0 no impressioN.
= 1 by default impressions.
= 2 more detailed impressions:

```
#1.Resolution.des.systemes.lineaires          CPU (USER+SYST/SYST/ELAPS):  7.72 0.82  8.72
#1.1.Numerotation, .connectivity .de.la.ma trice CPU (USER+SYST/SYST/ELAPS):  0.21 0.02  0.31
#1.2.Factorisation.symbolic                 CPU (USER+SYST/SYST/ELAPS):  0.58 0.05  1.28
#1.3.Factorisation.numeric. (ou.precond.)    CPU (USER+SYST/SYST/ELAPS):  6.78 0.73  7.71
#1.4.Resolution                               CPU (USER+SYST/SYST/ELAPS):  0.15 0.02  0.35
# 2.Calculs.elementaires.et.as semblages        CPU (USER+SYST/SYST/ELAPS): 28.87 0.64 29.47
#2.1.Routine.calcul                           CPU (USER+SYST/SYST/ELAPS): 26.61 0.56 26.61
#2.1.1.Routines.te00ij                        CPU (USER+SYST/SYST/ELAPS): 24.58 0.07 25.78
#2.2.Assemblages                              CPU (USER+SYST/SYST/ELAPS):  2.26 0.08  3.36
#2.2.1.Assemblage.matrices                    CPU (USER+SYST/SYST/ELAPS):  2.02 0.06  3.12
#2.2.2.Assemblage.seconds.membres             CPU (USER+SYST/SYST/ELAPS):  0.24 0.02  0.37
```

= 3 more detailed impressions and incremental impression for each step time.

During parallel calculations (MPI), the time spent in the communications is also displayed:

```
#4 Communications MPI                      CPU (USER+SYST/SYST/ELAPS)    : 12.67  0.50 12.68
```

4.9.2 Operand AVERAGE

◊ AVERAGE = 'NOT' pas d' posting of the statistics
= 'YES' posting of the statistics

The keyword AVERAGE allows to exclusively control the posting of additional statistics for parallel calculations. It is the average of measurements on all the processors as well as the standard deviation of these measurements.
Each displayed time is then supplemented as follows:

```
#1 Résolution.des.systèmes.linéaires       CPU (USER+SYST/SYST/ELAPS):  0.29  0.00  0.35
       (average….diff. .procs)            CPU (USER+SYST/SYST/ELAPS):  0.30  0.00  0.47
       (variation-type.diff. .procs)            CPU (USER+SYST/SYST/ELAPS):  0.01  0.00  0.05
```

4.10 Keyword MEMORY

The allowance of the various structures of data is a dynamic allocation, the user indicates the limits of resource during launching of achievable in the interface of access.

4.10.1 Operand TAILLE_GROUP_ELEM

TAILLE_GROUP_ELEM = tgrel [defect: 1000]

This parameter gives the maximum number of finite elements of the same type which will be gathered in a group of elements.
This parameter influences the performances memory and CPU of elementary calculations and the assemblies.
When one increases tgrel, one must in general save time CPU. On the other hand, objects JEVEUX are larger, which can require more memory.

4.10.2 Operand TAILLE_BLOC

TAILLE_BLOC = tbloc [defect: 800.]

This parameter gives the size of the blocks of the factorized matrices for solvor LDLT. This size is given in kiloR8 (1 kiloR8 = 1024 realities). This parameter influences the number of operations of input/output and thus over the time of assembly and resolution. By default this value is fixed at 800 kiloR8, that is to say 8 recordings by default on the file of direct access associated with base JEVEUX.

4.11 Keyword CATALOGUE

This keyword is reserved for the developers, it is used at the time of the operation of compilation of the catalogues of elements to obtain the file in the form of base JEVEUX.

4.11.1 Operand FILE

♦ FILE = nfic

Can take only the value ' CATAELEM'

4.11.2 Operand UNIT

◊ UNIT = unit

Logical number of unit associated with the catalogues of elements. In the procedures of construction of the catalogue of elements one uses like value 4. The file fort.4 is obtained starting from the contents of the repertoire of sources **catalo** using a procedure python.

4.12 Keyword RESERVE_CPU

Allows to reserve a share of the time CPU allotted to the job to properly finish the execution in the event of stop by lack of time CPU detected by an order Aster. This mechanism is useful only in the case of an execution batch of *Code_Aster*. The value of this reserve can be indicated in absolute value or in the form of a percentage of total time CPU. This value is limited by the value of the keyword LIMIT.
When the keyword CODE is present, i.e. for the whole of the tests of nonregression, one imposes systematically a reserve of time 10 second old CPU if the keyword RESERVE_CPU is absent.
After the execution of *Code_Aster,* it perhaps necessary to carry out operations of compression before transfer of the files of results or the Total base which are sometimes very expensive.

4.12.1 Operand VALE

Value expressed in seconds withdrawn from the total time CPU, over which certain total orders is based to stop the execution properly.

4.12.2 Operand PERCENTAGE

Percentage withdrawn from the total time CPU, over which certain total orders is based to stop the execution properly.

4.12.3 Operand LIMIT

Maximum value of the reserve of time, being worth by default 900 seconds.