How read the documentation of the Résumé

commands:

This note is a guide of reading of the U4 booklets and U7 of Manuel d' Utilisation.

She explains in particular the meaning of the méta-characters and the typographical conventions used for the description of the syntax of the commands.

All the examples given here are given as illustration and do not replace the full description of the commands appearing in the booklets U4 and U7.

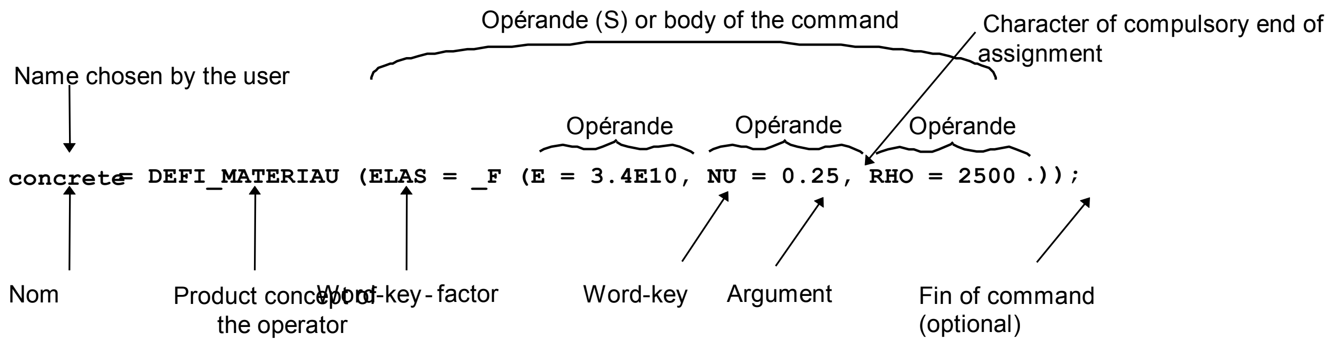
Contents

1 Rappels on the syntax of the commands of *Code_Aster*

the process control language and its supervisor are completely described in the document [U1.03.01]. One recalls here some notions on syntax of the commands of *Code_Aster*.

In *Code_Aster*, one understands by the generic term of commands at the same time **the operators**, **the procedures** and the macro-commands of the process control language. An operator provides **a product concept** typified (by the operator) and named by the user. A procedure does not generate a product concept, it achieves **actions** such as printings or resource allocations.

In the example below, one recalls the vocabulary which is used in the description of the commands.


Opérande (S) or body of the command
Character of compulsory end of assignment
Name chosen by the user
Opérande
Opérande
Opérande
concrete = DEFI_MATERIAU (ELAS = _F (E = 3.4E10, NU = 0.25, RHO = 2500 .));
Nom
Product concept of the operator
Word-key - factor
Word-key
Argument
Fin of command (optional)

Terminologie *Aster*

Une operand is thus the whole consisted a key word and its argument. However, in the documentation of the commands, one often indicates the operands of an operator or a procedure by the name of their key word. For example: `RHO`, simple key word, or `ELAS`, key word factor.

The term of **product concept** is generic for all the operators, it is the result of the work of the operator.

Here in example `DEFI_MATERIAU`, there was creation of data structure of the type `MATER` (material), named `concrete` by the user. It gathers the denominations (key word `E`, `NU`, `RHO`) and the values (arguments `3.4E10`, `0.25,2500 .`) of the mechanical elastic characteristics (key word factor `ELAS`) of the material.

The term of concept **of type result** applies to the outputs of the operators of computation, i.e. physical fields of variables (displacements, temperatures, stresses, forces, modes, etc…) on the nodes or the meshes at various times or for various frequencies.

The result concept comprises in general **under types.**

2 Standard plane of the documents of use of the commands

Chaque document of presentation of a command comprises the following chapters:

- Goal,
- Syntaxe,
- Opérandes,
- Exemples (possibly).

This presentation makes it possible to the user to find in only one document all knowledge necessary to the implementation of a command.

3 Paragraphe But

One states the functionality filled by the command (actions carried out). One also specifies the types of the concepts expected in input and the product concept, as well as characteristics of the command.

This paragraph is also displayed by the search engines; it thus contains only text without equations or formula.

Example: Opérateur `STAT_NON_LINE` [U4.51.03]

Drank :

Compute quasi-static mechanical evolution of a structure into nonlinear.

Nonthe linearity is related either to the behavior of the material (for example plastic), or with the geometry (for example in large displacements). To have details on the method of resolution employed, one will refer to documentation of reference [R5.03.01].

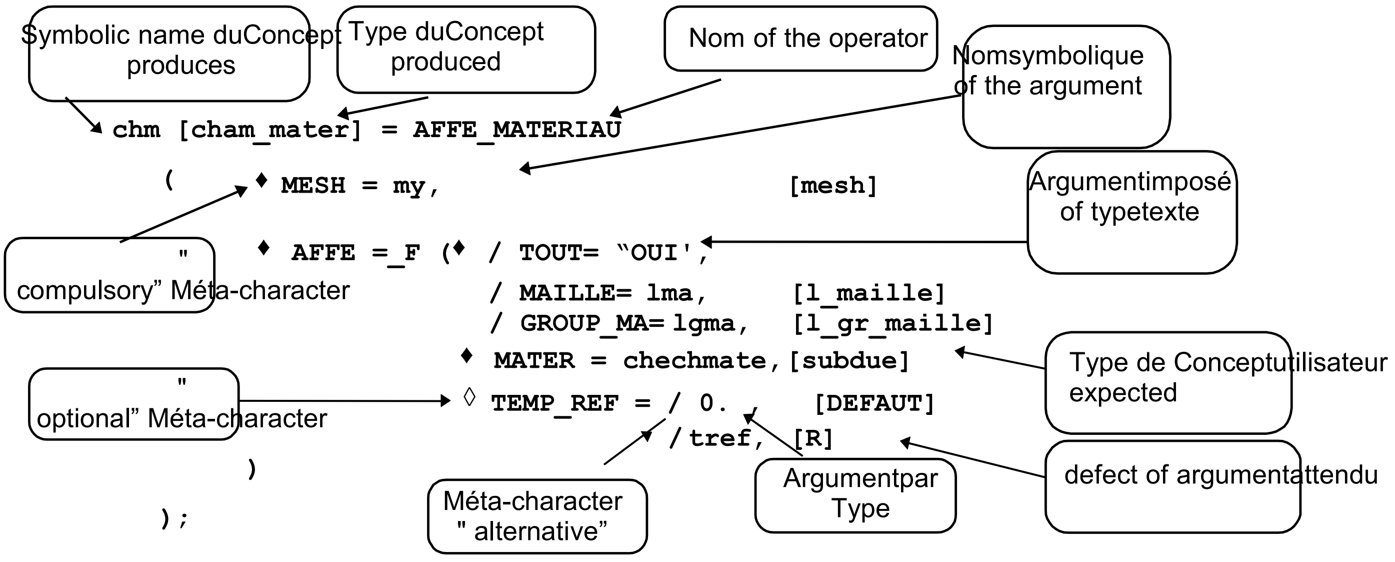
The evolution can be studied in several successive works (concept reentrant), either in continuation (the computed last moment is the initial time of following computation), or recovery some on the basis of one former time.

If the time necessary to carry out computation is not sufficient, the program stops, but the already computed results are saved if a data base were defined in the profile of study of the user. Product a data structure of the evol_noli `type`.

4 Paragraphe Syntaxe

One gives, in this paragraph, all the operands of the command. One specifies, for each operand, using méta-characters and of indentations suitable for the typographical presentation of the commands (cf example of operator `AFFE_MATERIAU`):

- the name of the operator,
- the name of the key words,
- symbolic names user of the product concept and the arguments of the key words,
- compulsory or optional character of the operands (statute),
- the alternatives in the choices of the operands,
- the standards of the arguments expected by the key words,
- the values by default taken by the arguments in the case of optional operands,
- the standard of the product concept, when it is about an operator.


Symbolic name duConcept produces
Type duConcept produced
Nom of the operator
Nomsymbolique of the argument
Argumentimposé of typetexte
" compulsory" Méta-character
" optional" Méta-character
Type de Conceptutilisateur expected
defect of argumentattendu
Méta-character " alternative"
Argumentpar Type

```
chm [cham_mater] = AFFE_MATERIAU
  (   ♦ MESH = my,                        [mesh]
      ♦ AFFE =_F (♦ / TOUT= "OUI',
                    / MAILLE= lma,      [l_maille]
                    / GROUP_MA= lgma,   [l_gr_maille]
                  ♦ MATER = chechmate, [subdue]
                  ◊ TEMP_REF = / 0.      [DEFAUT]
                               / tref,  [R]
                  )
  );
```

Presentation of the syntax (partial) of operator `AFFE_MATERIAU`

4.1 Méta-characters of statute of operands (♦ ◊ / |)

Four méta-characters are used to indicate the statute of the operands. It is necessary to understand here by statute of the operands their compulsory or optional statement and the nature of the alternatives in the choices of the operands.

These méta-characters do not form part of the process control language. They have only one function of documentary presentation and do not have to thus be used for the drafting of the command file.

4.1.1 Compulsory or optional operands

Elles are identified by the presence at the top of a black or white rhombus.

♦ black rhombus: it is compulsory to declare in the command the operands which follow this sign.
◊ white rhombus: the statement of the operands which follow this sign is optional. In the event of absence of the operand, the command will affect possibly one or of the values by default.

Example : operator `DEFI_LIST_ENTI` (definition of a list of strictly increasing integers whose values are regularly spaced)

```
Li =DEFI_LISTE_ENTI
                (   ♦ DEBUT = deb. ,
                   ◊INTERVALLE    = _F  (   ♦JUSQU_A=if             ,
                                           ♦PAS=ipas               ,
                                        )
                )
```

- It is compulsory to declare the operand identified by key word `DEBUT` and to provide `deb.` which is the first integer of the list to be built.
- He is not compulsory to declare the operand identified by the key word factor `INTERVALLE`. In this case the list of integers will be summarized with only one integer of value `deb.` (this is specified in the description of the operands).
- If operand `INTERVALLE` is declared, then it is compulsory to declare the operand `JUSQU_A` which specifies the whole end `yew` of the interval to be cut out with a constant pitch and the operand `PAS` which indicates the pitch `ipas` interval division.

4.1.2 Alternatives in the choice of the Elles

operands are identified by the presence at the top of each choice of the alternative:

- of one / (slash): exclusive alternative, only one choice among those proposed,
- of one | (pipe, semi colonist): nonexclusive alternative, one or more choice among those proposed.

Example of exclusive alternative: operator `AFFE_MODELE` (assignment of the type of finite elements on whole or part of a mesh).

```
Mo   = AFFE_MODELE (
       ♦MAILLAGE      = my
       ♦AFFE       = _F   (  ♦   /  TOUT=          'OUI”,
                                /MAILLE          =mail     ,
[l_maille]
                                /   NOEUD=noeu            ,
[l_noeud]
                                /GROUP_MA     =g_mail      ,
[l_gr_maille]
                                /GROUP_NO     =g_noeu      ,
[l_gr_noeud]
               ...............
                        )
                            )   ;
```

In operand `AFFE` (compulsory) it should be indicated where will be affected, on the mesh, the type of finite element specified in operands `PHENOMENE` and `MODELISATION` of the same command:

- either on all mesh (`TOUT`),
- or on certain meshes (`MESH`),
- or on certain nodes (`NODE`),
- or on certain mesh groups (`GROUP_MA`),
- or on certain nodes groups (`GROUP_NO`).

Example of nonexclusive alternative:

operator `AFFE_CHAR_MECA` operand `DDL_IMPO` (assignment of displacements imposed on degrees of freedom).

```
DDL_IMPO = _F  (  ♦   /  TOUT    =     'OUI',
                       /NOEUD      =   lno ,       [l_noeud]
                       /GROUP_NO    =   lgno,       [l_gr_noeud]
                      /MAILLE     =lma        ,        [l_maille]
                      /GROUP_MA =lgma      ,       [l_gr_maille]
                  ♦   |   DX   =UX            ,        [R]
                      |   DY   =UY            ,        [R]
                      |   DZ   =UZ            ,        [R]
                      |   DRX =          □ X   ,    [R]
                      |   DRY =          □ there  ,    [R]
                      |   DRZ =          □ Z   ,    [R]
                      |   GRX =G             ,       [R]
                      |   PRES=p             ,       [R]
                      |   PHI =          □ ,       [R]
                      |   TEMP=T             ,       [R]
                      |   PRE1=pr1            ,        [R]
                      |   PRE2=pr2            ,        [R]
              )
```

Dans this operator, it is necessary to specify obligatorily:

- the scope of application on the mesh: everywhere (`TOUT`), on certain nodes (`NODE`) or certain nodes groups (`GROUP_NO`),
- on which degrees of freedom with which specified values by the user.

Méta-character **|** indicate that the user can impose a value of displacement on **one** (the symbol ♦ indicates that one needs at least one of them) or **more** of degrees of freedom (`DX`, `DY`, `DZ`, `DRX`, `DRY`, `DRZ`, `GRX`, `PRES`, `PHI`, `TEMP`, `PRE1`, `PRE2`) of the beforehand indicated nodes.

4.1.3 Combinations of the méta-characters of choice of the operands

Ces méta-characters can be combined to illustrate the multiplicity of the choices in certain commands.

Example : order `DEFI_MATERIAU` (definition of a material by its properties of behavior)

Pour a study of thermomechanics, one needs to define a material having **at the same time** mechanical characteristics (`ELAS`) and thermals (`THER`) from where use of the pipe: **|**

But in each choice, one is obliged to choose if the properties of the material are dependent (`_FO`) or not on the temperature from where use of the slash: **/** ; cf below:

```
my =   DEFI_MATERIAU     (      |   / ELAS = _F   (  ♦   E =        yg,
                                                     ♦  NU =nu    ,
                                                     ◊  RHO =rho      ,
                                                     ◊  ALPHA =dil    ,
                                              )
       ......
                                / ELAS_FO =_F (  ♦   E =       f1,
                                                 ♦  NU =f2     ,
                                                 ◊  RHO =f3        ,
                                                 ◊  ALPHA =f4 ,
                                              )

                            |   / THER = _F   ( ♦    RHO_CP =      CP,
                                                ♦    LAMBDA =la   ,
                                              )
       ......
                                / THER_FO =_F (    ♦    RHO_CP =g1   ,
                                                ♦LAMBDA    =g2   ,
                                              )
       ......
                         );
```

4.2 Méta-characters of the type of concept or argument

Comme the méta-characters of statutes of operands, the hooks [] and the star * do not form part of the process control language. They have only one function of documentary presentation.

4.2.1 Types of concepts or arguments []

Ils frame the type of the product concepts as well as the type of the arguments.

Example : order `AFFE_MODELE` (Affectation of the finite elements on the meshes of a mesh)

```
Mo   [model] = AFFE_MODELE
                     (  ♦MAILLAGE  =ma   ,                        [mesh]
                        ♦AFFE   =_F    ( ♦   /  TOUT=   "OUI",
                                             /MAILLE       = mall,    [l_maille]
....................
       )
```

Dans the example above, one thus specifies that the product concept by `AFFE_MODELE` is of model `type` and that the expected concept as argument of the key word `MESH` must be of l_maille `type` (i.e list of mesh).

4.2.2 Type of the product concept [*]

This méta-character indicates that the type of the product concept, or under type of the product concept of type result, depends on the types of the arguments of certain operands. In this case the various possibilities are registered after the syntax of the command.

Example: order `CREA_CHAMP`

Dans this example, `ch2` will be a field with the nodes, a card or a field by element according to the value of `TYPE_CHAM`.

```
ch2  [*] = CREA_CHAMP
    (  ♦  TYPE_CHAM =   / ` NOEU_xxxx',                   [kN]
                        / ` CART_xxxx',
                        / ` ELGA_xxxx',
                        / ` ELNO_xxxx',
                        / ` ELEM_xxxx',
    )

          SiTYPE_CHAM=     'NOEU_TEMP_R'        then [*]    =CHAM_NO_TEMP_R

                         'NOEU_DEPL_R'
CHAM_NO_DEPL_R
                        …
                        'CART_TEMP_R'                    CARTE_TEMP_R
                        'CART_DEPL_R'                    CARTE_DEPL_R
                        …
```

4.3 Commentaires

Pour certain commands complex such as `AFFE_CARA_ELEM` or `DEFI_MATERIAU` for example, the character of comment is employed to comment on the alternatives of the operands. It has the same meaning that in the process control language and is interpreted like such by the supervisor.

Example for `AFFE_CARA_ELEM` :

```
POUTRE=_F (♦ / NET =lma       ,                          [l_maille]
             / GROUP_MA =lgma    ,                          [l_gr_maille]
          ♦  /  SECTION = "GENERALE",
                /  # constant section
                   ◊CARA=    | "A"
                             | "IY" | "IZ"
lists choices             | "AY" | Possible "AZ
" for one             | "EY" | "EZ"
constant section          | "JX"
                             | "RY" | "RZ" | "RT",
                /  # section variable
                   ◊CARA=    | "A1"  | "A2"
                             | "IY1" | "IY2" | "IZ1" | "IZ2"
lists choices             | "AY1" | "AY2" | "AZ1" | Possible "AZ2
" for one             | "EY1" | "EY2" | "EZ1" | "EZ2"
variable section             | "JX1" | "JX2"
                             | "RY1" | "RY2" | "RZ1" | "RZ2" | "RT1" | "RT2",
.....
         )
```

4.4 Types of the arguments expected by the key words

Les key words of the operands expect arguments which correspond, in general, with four classes:

- values, one then specifies by a symbolic name the accepted data-processing type (real, whole, character string, etc…),
- imposed texts, then the texts (`"OUI"`, `"HY1"`) are indicated between quotes,
- of the names of topological entities simple (name of node, meshes, or lists of names), declared in the mesh file, or of the names of nodes groups or meshes, or lists of names of nodes groups or meshes,
- the names and the lists of names of product concepts by the operators.

The table below gathers all the main types of the arguments expected by the key words:

1) `[R]`	real	`3.`
`[l_R]`	list of realities	`(1. , 3. , 7.)`
`[I]`	whole	`7`
`[l_I]`	list of integers	`(9, 6,1,9)`
`[C]`	complex	`IH 1.1,7.8 or MP 10. , 1.57`
`[l_C]`	list of complexes	`(IH 1.1,7.), (IH 4.7,9.)`
`[TXM]`	unconstrained text (name of TITLE…)	**` my title'**
`[KN]`	text lower or equal to `N` characters	`'INST'`
`[l_Kn]`	list of texts lower or equal to `N` characters	``(` SIXX', ` SIYY', ` SIXY')``
`[node]`	name of node	`N23`
`[l_noeud]`	lists names of nodes	`(N23, N24, N25)`
`[gr_noeud]`	name of nodes group	`NBORD6`

`[l_gr_noeud]`	lists names of nodes groups	`(NBORD, NBASE, NBORD)`
`[mesh]`	name of mesh	`M34`
`[l_maille]`	lists name of mesh	`(M34, M35)`
`[gr_maille]`	name of mesh group	`MPIQUAGE`
`[l_gr_maille]`	lists names of mesh groups	`(MSOM, MDROI, MGA)`
`[type_concept]`	standard of concept (or field) produced beforehand with generally automatic checking of the type	`monresu`
`[l_type_concept]`	list of type of concept user	`(resu1, resu2)`

4.5 Types of the product concepts in *Aster*

One uses the méta-character of choice of exclusive alternative / to mean the multiplicity of concept expected behind a key word.

Example: operator `ASSE_MATRICE` (assembly of the elementary matrixes contained in a list of concepts of the matr_elem_* `type`.)

```
my   [matr_asse_*] = ASSE_MATRICE
               (  ♦  MATR_ELEM =lmel     ,  /        [l_matr_elem_DEPL_R]
                                            /        [l_matr_elem_DEPL_C]
                                            /        [l_matr_elem_TEMP_R]
                                            /        [l_matr_elem_TEMP_C]
                                            /        [l_matr_elem_PRES_R]
                                            /        [l_matr_elem_PRES_C]
…
               );
```

```
if MATR_ELEM        [matr_elem_DEPL_R]      then      [*]     	     DEPL_R
                   [matr_elem_DEPL_C]                        	     DEPL_C
                   [matr_elem_TEMP_R]                        	     TEMP_R
                   [matr_elem_TEMP_C]                        	     TEMP_C
                   [matr_elem_PRES_R]                        	     PRES_R
                   [matr_elem_PRES_C]                        	     PRES_C
```

Dans the example above the concept expected in argument of `MATR_ELEM` can be various types and type of the last concept in argument by the user will depend (according to stated rules Ci - above) typing on the product concept by operator `ASSE_MATRICE`.

5 Paragraphe Opérandes

One describes, for each operand the meaning of the operand for this command, the nature and the type of the arguments expected by the key words, and the restrictions and difficulties of employment.

For example, in the documentation of operator `AFFE_MATERIAU`, for the operand `AFFE,` operand intended to specify on which (S) entity (S) topological (S) of the mesh of name `my` will be affected the material of name `chechmate` produced by operator `DEFI_MATERIAU`, one will read:

♦`AFFE`

Word-key factor which makes it possible to affect various materials on “pieces” of the mesh.

`/TOUT   = ` OUI',`

This key word makes it possible to affect on all the meshes of the mesh.

`/GROUP_MA  = lgma,`

This key word makes it possible to affect on a list of mesh groups of the mesh.

`/MAILLE    =lma  ,`

This key word makes it possible to affect on a list of meshes of the mesh.

A each mesh group, (key word `GROUP_MA`) or each list of meshes (key word `NETS`), or with all the mesh (key word `TOUT`) is affected a material `chechmate`, which is a product concept by one of operators `DEFI_MATERIAU` [U4.43.01] or `DEFI_COMPOSITE` [U4.42.03].

If a mesh appears explicitly (or implicitly) in several occurrences of the key word factor `AFFE`, the rule of overload is observed: it is the last assignment which precedes [U2.01.08].

6 Phases of checking / of execution

the Syntaxe paragraph of the documentation of use is the exact reflection of the catalogue of the command. This catalogue is a file which understands, written in the language of the supervisor, all the rules on the key words: presence, exclusion, implication, contained…

editor EFICAS exploit this catalogue of command and allow so the user, with final the made up file is valid, to obtain a correct command set.

With the execution of the study, the supervisor of *Code_Aster* reproduces the same task of syntactic checking: either overall for all the file, or while alternating with the execution, orders by command.

Moreover, during the execution itself of the commands (entered part FORTRAN of the source code), of the additional checks can be made. They are stresses impossible to manage with the language level of command (equality of cardinals of different lists…).

7 Print and Pour

indentations the legibility of the document concerning to the commands, all that refers to syntax is printed in `font Courier 10 points`. One differentiates the various types of functional elements (product concept, key word, key word factor, argument) by the use of uppercases and tiny.

In capital letters:

- names of the operators, of the procedures
- names of the key words and the key words factors,
- imposed arguments of standard text (those are between "quotes" as in the syntax of the commands).

In small letters:

- names of the product concepts,
- symbolic names of the arguments,
- types of the product concepts and the arguments.

Into mixed tiny - uppercase when the product concept admits under type. The aforementioned appears in capital letters as well as type FORTRAN of the quantity of under type.

One reinforces the legibility of syntax by the use of indentations. They are used with the identification of the blocks of operands and the release of a group as operands under a key word factor. Are also used they to lay out the brackets of the same block under the same balance.

Example:

```
my   [matr_asse_*] = ASSE_MATRICE

     (  ♦   MATR_ELEM =lmel     ,   /                  [l_matr_elem_DEPL_R]
                                    /                  [l_matr_elem_DEPL_C]
                                    /                  [l_matr_elem_TEMP_R]
                                    /                  [l_matr_elem_PRES_C]

        ♦   NUME_DDL  =nu       ,                          [nume_ddl]

       ◊CHAR_CINE     =lcha     ,   /                  [l_char_cine_meca]
                                     /                 [l_char_cine_ther]
                                    /                  [l_char_cine_acou]

       ◊INFO   =                /1  ,                [DEFAUT]
                                    /2  ,

    );


if MATR_ELEM      [matr_elem_DEPL_R]        then      [*]
DEPL_R
                 [matr_elem_DEPL_C]                                    DEPL_C
                 [matr_elem_TEMP_R]                                    TEMP_R
                 [matr_elem_PRES_C]                                    PRES_C
```