Comment lire la documentation des commandes

Résumé :

Cette note est un guide de lecture des fascicules U4 et U7 du Manuel d'Utilisation.

Elle explique notamment la signification des méta-caractères et des conventions typographiques utilisées pour la description de la syntaxe des commandes.

Tous les exemples donnés ici sont donnés à titre d'illustration et ne se substituent pas à la description complète des commandes figurant dans les fascicules U4 et U7.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Table des matières

1 Rappels sur la syntaxe des commandes de Code_Aster....3
2 Plan type des documents d'utilisation des commandes....3
3 Paragraphe But....4
4 Paragraphe Syntaxe....4
4.1 Méta-caractères de statut d'opérandes (♦ ◊ / |)....5
4.1.1 Opérandes obligatoires ou facultatives....5
4.1.2 Alternatives dans le choix des opérandes....5
4.1.3 Combinaisons des méta-caractères de choix des opérandes....7
4.2 Méta-caractères de type de concept ou d'argument....8
4.2.1 Types de concepts ou d'arguments [ ]....8
4.2.2 Type du concept produit [*]....8
4.3 Commentaires....9
4.4 Types des arguments attendus par les mots-clés....9
4.5 Types des concepts produits dans Aster....10
5 Paragraphe Opérandes....11
6 Phases de vérification / d'exécution....11
7 Typographie et indentations....12

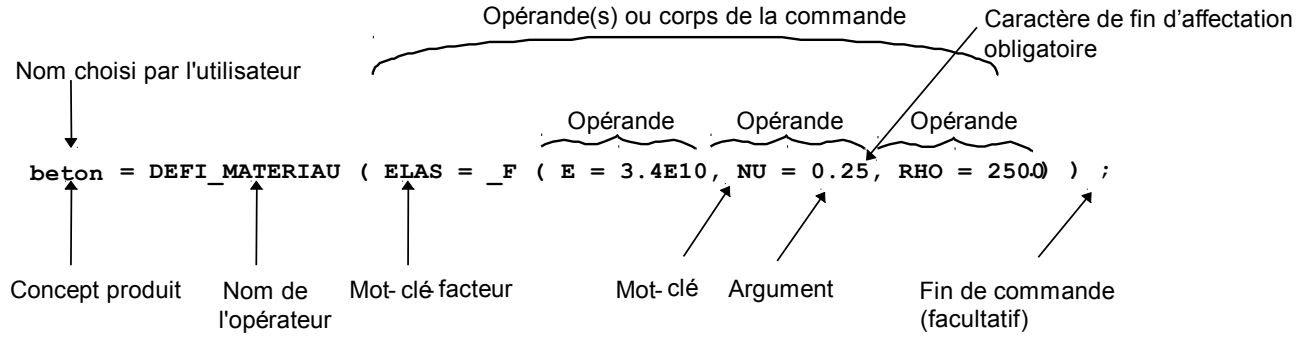
Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

1 Rappels sur la syntaxe des commandes de *Code_Aster*

Le langage de commande et son superviseur sont complètement décrits dans le document [U1.03.01]. On rappelle ici quelques notions sur la syntaxe des commandes de *Code_Aster*.

Dans *Code_Aster*, on entend par le terme générique de commandes à la fois les **opérateurs**, les **procédures** et les macro-commandes du langage de commande. Un opérateur fournit un **concept produit** typé (par l'opérateur) et nommé par l'utilisateur. Une procédure ne génère pas de concept produit, elle accomplit des **actions** telles que des impressions ou des allocations de ressources.

Dans l'exemple ci-dessous, on rappelle le vocabulaire qui est utilisé dans la description des commandes.



Terminologie *Aster*

Une opérande est donc l'ensemble constitué par un mot clé et son argument. Toutefois, dans la documentation des commandes, on désigne souvent les opérandes d'un opérateur ou d'une procédure par le nom de leur mot-clé. Par exemple : `RHO`, mot clé simple, ou `ELAS`, mot clé facteur.

Le terme de **concept produit** est générique pour tous les opérateurs, c'est le résultat du travail de l'opérateur.

Ici dans l'exemple `DEFI_MATERIAU`, il y a eu création de la structure de données de type `mater` (matériau), nommée `beton` par l'utilisateur. Elle regroupe les dénominations (mot-clé `E`, `NU`, `RHO`) et les valeurs (arguments `3.4E10`, `0.25`, `2500.`) des caractéristiques élastiques mécaniques (mot-clé facteur `ELAS`) du matériau.

Le terme de concept **de type résultat** s'applique aux sorties des opérateurs de calcul, c'est-à-dire des champs de grandeurs physiques (déplacements, températures, contraintes, efforts, modes, etc …) sur les nœuds ou sur les mailles à différents instants ou pour différentes fréquences.

Le concept résultat comporte en général des **sous types.**

2 Plan type des documents d'utilisation des commandes

Chaque document de présentation d'une commande comporte les chapitres suivants :

- But,
- Syntaxe,
- Opérandes,
- Exemples (éventuellement).

Cette présentation permet à l'utilisateur de trouver dans un seul document toutes les connaissances nécessaires à la mise en œuvre d'une commande.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

3 Paragraphe But

On énonce la fonctionnalité remplie par la commande (actions réalisées). On précise également les types des concepts attendus en entrée et du concept produit, ainsi que des particularités de la commande.

Ce paragraphe est aussi affiché par les moteurs de recherche ; il contient donc uniquement du texte sans équations ou formule.

Exemple : Opérateur STAT_NON_LINE [U4.51.03]

But :

Calculer l'évolution mécanique quasi-statique d'une structure en non linéaire.

La non linéarité est liée soit au comportement du matériau (par exemple plastique), soit à la géométrie (par exemple en grands déplacements). Pour avoir des détails sur la méthode de résolution employée, on se reportera à la documentation de référence [R5.03.01].

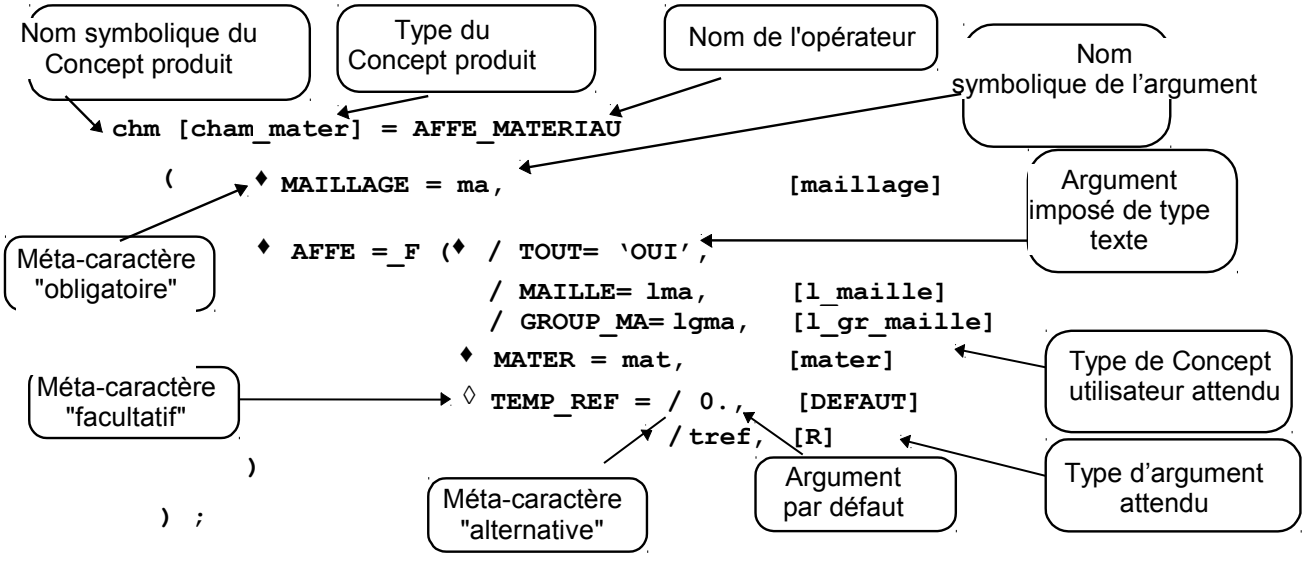
L'évolution peut être étudiée en plusieurs travaux successifs (concept réentrant), soit en poursuite (le dernier instant calculé est l'instant initial du calcul suivant), soit en reprise en partant d'un instant antérieur.

Si le temps nécessaire pour effectuer le calcul n'est pas suffisant, le programme s'interrompt, mais les résultats déjà calculés sont sauvegardés si une base de données a été définie dans le profil d'étude de l'utilisateur. Produit une structure de données de type evol_noli.

4 Paragraphe Syntaxe

On donne, dans ce paragraphe, l'ensemble des opérandes de la commande. On précise, pour chaque opérande, à l'aide de méta-caractères et d'indentations propres à la présentation typographique des commandes (cf. exemple de l'opérateur AFFE_MATERIAU) :

- le nom de l'opérateur,
- le nom des mots-clés,
- les noms symboliques utilisateur du concept produit et des arguments des mots-clés,
- le caractère obligatoire ou facultatif des opérandes (statut),
- les alternatives dans les choix des opérandes,
- les types des arguments attendus par les mots-clés,
- les valeurs par défaut prises par les arguments dans le cas d'opérandes facultatives,
- le type du concept produit, lorsqu'il s'agit d'un opérateur.



Présentation de la syntaxe (partielle) de l'opérateur AFFE_MATERIAU

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

4.1 Méta-caractères de statut d'opérandes (♦ ◊ / |)

Quatre méta-caractères sont utilisés pour indiquer le statut des opérandes. Il faut entendre ici par statut des opérandes leur déclaration obligatoire ou facultative et la nature des alternatives dans les choix des opérandes.

Ces méta-caractères ne font pas partie du langage de commande. Ils n'ont qu'une fonction de présentation documentaire et ne doivent donc pas être utilisés pour la rédaction du fichier de commandes.

4.1.1 Opérandes obligatoires ou facultatives

Elles sont repérées par la présence en tête d'un losange noir ou blanc.

- ♦ losange noir : il est obligatoire de déclarer dans la commande le ou les opérandes qui suivent ce signe.
- ◊ losange blanc : la déclaration de la ou des opérandes qui suivent ce signe est facultative. En cas d'absence de l'opérande, la commande affectera éventuellement une ou des valeurs par défaut.

Exemple : opérateur DEFI_LIST_ENTI (définition d'une liste d'entiers strictement croissants dont les valeurs sont régulièrement espacées)

```
li =  DEFI_LIST_ENTI
                (    ♦ DEBUT = deb   ,
                     ◊  INTERVALLE = _F  (   ♦  JUSQU_A   =  if ,
                                             ♦  PAS       =  ipas,
                                         )
                 )
```

- Il est obligatoire de déclarer l'opérande identifiée par le mot-clé DEBUT et de fournir deb qui est le premier entier de la liste à construire.
- Il n'est pas obligatoire de déclarer l'opérande identifiée par le mot-clé facteur INTERVALLE. Dans ce cas la liste d'entiers se résumera à un seul entier de valeur deb (ceci est précisé dans la description des opérandes).
- Si l'opérande INTERVALLE est déclarée, alors il est obligatoire de déclarer l'opérande JUSQU_A qui précise l'extrémité entière if de l'intervalle à découper avec un pas constant et l'opérande PAS qui indique le pas ipas de découpage de l'intervalle.

4.1.2 Alternatives dans le choix des opérandes

Elles sont repérées par la présence en tête de chaque choix de l'alternative :

- d'un / (slash) : alternative exclusive, un seul choix parmi ceux proposés,
- d'un | (pipe, semi colon) : alternative non exclusive, un ou plusieurs choix parmi ceux proposés.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Exemple d'alternative exclusive : opérateur AFFE_MODELE (affectation du type d'éléments finis sur tout ou partie d'un maillage).

```
mo   = AFFE_MODELE (
       ♦      MAILLAGE = ma
       ♦      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]
                  ..............
                            )
                               )   ;
```

Dans l'opérande AFFE (obligatoire) il faut indiquer où sera affecté, sur le maillage, le type d'élément fini précisé dans les opérandes PHENOMENE et MODELISATION de la même commande :

- soit sur tout le maillage (TOUT),
- soit sur certaines mailles (MAILLE),
- soit sur certains nœuds (NOEUD),
- soit sur certains groupes de mailles (GROUP_MA),
- soit sur certains groupes de nœuds (GROUP_NO).

Exemple d'alternative non exclusive :

opérateur AFFE_CHAR_MECA opérande DDL_IMPO (affectation de déplacements imposés sur des degrés de libertés).

```
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  =          y  ,      [R]
                     |   DRZ  =          z  ,      [R]
                     |   GRX  =         g   ,      [R]
                     |   PRES=          p   ,      [R]
                     |   PHI  =          ,         [R]
                     |   TEMP=          T   ,      [R]
                     |   PRE1=          pr1 ,      [R]
                     |   PRE2=          pr2 ,      [R]
                  )
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Dans cet opérateur, il faut préciser obligatoirement :

- le domaine d'application sur le maillage : partout (TOUT), sur certains nœuds (NOEUD) ou sur certains groupes de nœuds (GROUP_NO),
- sur quels degrés de libertés avec quelles valeurs imposées par l'utilisateur.

Le méta-caractère | indique que l'utilisateur peut imposer une valeur de déplacement sur **un** (le symbole ♦ indique qu'il en faut au moins un) ou **plusieurs** des degrés de liberté (DX, DY, DZ, DRX, DRY, DRZ, GRX, PRES, PHI, TEMP, PRE1, PRE2) des nœuds préalablement désignés.

4.1.3 Combinaisons des méta-caractères de choix des opérandes

Ces méta-caractères peuvent être combinés pour illustrer la multiplicité des choix dans certaines commandes.

Exemple : commande DEFI_MATERIAU (définition d'un matériau par ses propriétés de comportement)

Pour une étude de thermomécanique, on a besoin de définir un matériau ayant **à la fois** des caractéristiques mécaniques (ELAS) et thermiques (THER) d'où l'emploi du pipe : |

Mais dans chaque choix, on est obligé de choisir si les propriétés du matériau sont dépendantes (_FO) ou non de la température d'où l'emploi du slash : / ; cf ci-dessous :

```
ma =   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,
                                                )
       ……
                            );
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

4.2 Méta-caractères de type de concept ou d'argument

Comme les méta-caractères de statuts d'opérandes, les crochets [ ] et l'étoile * ne font pas partie du langage de commande. Ils n'ont qu'une fonction de présentation documentaire.

4.2.1 Types de concepts ou d'arguments [ ]

Ils encadrent le type des concepts produits ainsi que le type des arguments.

Exemple : commande AFFE_MODELE (Affectation des éléments finis sur les mailles d'un maillage)

```
mo [modele] = AFFE_MODELE
                    (  ♦  MAILLAGE =   ma,                    [maillage]
                       ♦  AFFE =_F  (  ♦  /  TOUT   = 'OUI',
                                       /  MAILLE = mail,   [l_maille]
.....................
        )
```

Dans l'exemple ci-dessus, on précise donc que le concept produit par AFFE_MODELE est de type modele et que le concept attendu comme argument du mot-clé MAILLE doit être de type l_maille (i.e. liste de maille).

4.2.2 Type du concept produit [*]

Ce méta-caractère indique que le type du concept produit, ou le sous type du concept produit de type résultat, dépend des types des arguments de certaines opérandes. Dans ce cas les diverses possibilités sont inscrites après la syntaxe de la commande.

Exemple : commande CREA_CHAMP

Dans cet exemple, ch2 sera un champ aux nœuds, une carte ou un champ par élément selon la valeur de TYPE_CHAM.

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

Si TYPE_CHAM = 'NOEU_TEMP_R' alors [*] = CHAM_NO_TEMP_R
'NOEU_DEPL_R'
CHAM_NO_DEPL_R

```
                       ...
                       'CART_TEMP_R'                  CARTE_TEMP_R
                       'CART_DEPL_R'                  CARTE_DEPL_R
                       ...
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

4.3 Commentaires

Pour certaines commandes complexes telle que AFFE_CARA_ELEM ou DEFI_MATERIAU par exemple, le caractère de commentaire est employé pour commenter les alternatives des opérandes. Il a le même sens que dans le langage de commande et est interprété comme tel par le superviseur.

Exemple pour AFFE_CARA_ELEM :

```
POUTRE=_F(♦   / MAILLE =       lma,                          [l_maille]
              / GROUP_MA =     lgma,                         [l_gr_maille]
          ♦   /  SECTION = 'GENERALE',
                 /   # section constante
                     ◊  CARA=  | 'A'
                               | 'IY' | 'IZ'
liste des choix                | 'AY' | 'AZ'
possibles pour une             | 'EY' | 'EZ'
section constante              | 'JX'
                               | 'RY' | 'RZ' | 'RT',
                 /   # section variable
                     ◊  CARA=  | 'A1'  | 'A2'
                               | 'IY1' | 'IY2' | 'IZ1' | 'IZ2'
liste des choix                | 'AY1' | 'AY2' | 'AZ1' | 'AZ2'
possibles pour une             | 'EY1' | 'EY2' | 'EZ1' | 'EZ2'
section variable               | 'JX1' | 'JX2'
                               | 'RY1' | 'RY2' | 'RZ1' | 'RZ2' | 'RT1' | 'RT2',
.....
       )
```

4.4 Types des arguments attendus par les mots-clés

Les mots-clés des opérandes attendent des arguments qui correspondent, en général, à quatre classes :

- des valeurs, on précise alors par un nom symbolique le type informatique accepté (réel, entier, chaîne de caractères, etc …),
- des textes imposés, alors le ou les textes ('OUI', 'HY1') sont indiqués entre quotes,
- des noms d'entités topologiques simples (nom de nœud, de mailles, ou listes de noms), déclarés dans le fichier de maillage, ou des noms de groupes de nœuds ou de mailles, ou listes de noms de groupes de nœuds ou de mailles,
- des noms et des listes de noms de concepts produits par les opérateurs.

Le tableau ci-dessous rassemble tous les principaux types des arguments attendus par les mots clés :

1) [R]	réel	3.
[l_R]	liste de réels	(1., 3., 7.)
[I]	entier	7
[l_I]	liste d'entiers	(9, 6, 1, 9)
[C]	complexe	RI 1.1, 7.8 ou MP 10.,1.57
[l_C]	liste de complexes	(RI 1.1, 7.), (RI 4.7, 9.)
[TXM]	texte sans contrainte (nom de TITRE...)	**'mon titre'**
[Kn]	texte inférieur ou égal à n caractères	'INST'
[l_Kn]	liste de textes inférieurs ou égaux à n caractères	('SIXX', 'SIYY', 'SIXY')
[noeud]	nom de nœud	N23
[l_noeud]	liste de noms de nœuds	(N23, N24, N25)
[gr_noeud]	nom de groupe de nœuds	NBORD6

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

[l_gr_noeud]	liste de noms de groupes de nœuds	(NBORD, NBASE, NBORD)
[maille]	nom de maille	M34
[l_maille]	liste de nom de maille	(M34, M35)
[gr_maille]	nom de groupe de mailles	MPIQUAGE
[l_gr_maille]	liste de noms de groupes de mailles	(MSOM, MDROI, MGA)
[type_concept]	type de concept (ou de champ) produit préalablement avec généralement vérification automatique du type	monresu
[l_type_concept]	liste de type de concept utilisateur	(resu1, resu2)

4.5 Types des concepts produits dans *Aster*

On utilise le méta-caractère de choix d'alternative exclusive / pour signifier la pluralité de concept attendu derrière un mot-clé.

Exemple : opérateur ASSE_MATRICE (assemblage des matrices élémentaires contenues dans une liste de concepts de type matr_elem_*.)

```
ma [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]
...
              );
```

si MATR_ELEM	[matr_elem_DEPL_R]	alors	[*]	□	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 l'exemple ci-dessus le concept attendu en argument de MATR_ELEM peut être de différents types et du type du concept passé en argument par l'utilisateur dépendra (selon les règles énoncées ci-dessus) le typage du concept produit par l'opérateur ASSE_MATRICE.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

5 Paragraphe Opérandes

On décrit, pour chaque opérande le sens de l'opérande pour cette commande, la nature et le type des arguments attendus par les mots-clés, et les restrictions et difficultés d'emploi.

Par exemple, dans la documentation de l'opérateur AFFE_MATERIAU, pour l'opérande AFFE, opérande destinée à préciser sur quelle(s) entité(s) topologique(s) du maillage de nom ma va être affecté le matériau de nom mat produit par l'opérateur DEFI_MATERIAU, on lira :

♦ AFFE

Mot-clé facteur qui permet d'affecter différents matériaux sur des «morceaux» du maillage.

/ TOUT = 'OUI',

Ce mot-clé permet d'affecter sur toutes les mailles du maillage.

/ GROUP_MA = lgma,

Ce mot-clé permet d'affecter sur une liste de groupes de mailles du maillage.

/ MAILLE = lma,

Ce mot-clé permet d'affecter sur une liste de mailles du maillage.

A chaque groupe de mailles, (mot-clé GROUP_MA) ou chaque liste de mailles (mot-clé MAILLE), ou encore à tout le maillage (mot-clé TOUT) est affecté un matériau mat, qui est un concept produit par l'un des opérateurs DEFI_MATERIAU [U4.43.01] ou DEFI_COMPOSITE [U4.42.03].

Si une maille apparaît explicitement (ou implicitement) dans plusieurs occurrences du mot-clé facteur AFFE, la règle de surcharge est appliquée : c'est la dernière affectation qui prime [U2.01.08].

6 Phases de vérification / d'exécution

Le paragraphe Syntaxe de la documentation d'utilisation est le reflet exact du catalogue de la commande. Ce catalogue est un fichier qui comprend, écrites dans le langage du superviseur, toutes les règles sur les mots clés : présence, exclusion, implication, contenu …

L'éditeur EFICAS exploite ce catalogue de commande et permet à l'utilisateur, si au final le fichier composé est valide, d'obtenir un jeu de commandes correct.

A l'exécution de l'étude, le superviseur de *Code_Aster* reproduit la même tâche de vérification syntaxique : soit globalement pour tout le fichier, soit en alternant avec l'exécution, commande par commande.

De plus, lors de l'exécution proprement dite des commandes (entrées dans la partie FORTRAN du code source), des vérifications supplémentaires peuvent être faites. Il s'agit de contraintes impossibles à gérer au niveau du langage de commande (égalité de cardinaux de listes différentes …).

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

7 Typographie et indentations

Pour la lisibilité des documents relatifs aux commandes, tout ce qui se rapporte à la syntaxe est imprimé en `police Courier 10 points`. On différencie les différents types d'éléments fonctionnels (concept produit, mot-clé, mot-clé facteur, argument) par l'emploi de majuscules et de minuscules.

En majuscules :
- noms des opérateurs, des procédures
- noms des mots-clés et des mots-clés facteurs,
- arguments imposés de type texte (ceux-ci sont entre 'quotes' comme dans la syntaxe des commandes).

En minuscules :
- noms des concepts produits,
- noms symboliques des arguments,
- types des concepts produits et des arguments.

En mixte minuscule - majuscule lorsque le concept produit admet un sous type. Celui-ci apparaît en majuscules ainsi que le type FORTRAN de la grandeur du sous type.

On renforce la lisibilité de la syntaxe par l'utilisation d'indentations. Elles servent au repérage des blocs d'opérandes et au dégagement d'un groupe d'opérandes sous un mot-clé facteur. On les utilise aussi pour disposer les parenthèses d'un même bloc sous le même aplomb.

Exemple :

```
ma  [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,

    );


si MATR_ELEM     [matr_elem_DEPL_R]         alors     [*]          DEPL_R
                 [matr_elem_DEPL_C]                                DEPL_C
                 [matr_elem_TEMP_R]                                TEMP_R
                 [matr_elem_PRES_C]                                PRES_C
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)