Manuel d'Utilisation
Fascicule U4.1- : Gestion
Document : U4.12.03

Procédure DEFI_FICHIER

1 But

Ajouter et/ou supprimer une association unité logique-nom symbolique de fichier. Cette association unité logique-nom symbolique de fichier vient en complément de celles définies par défaut dans *Code_Aster* et par les procédures DEBUT [U4.11.01] et POURSUITE [U4.11.03]. Cette action peut être effectuée à tout moment au cours du travail.

2 Syntaxe

```
DEFI_FICHIER

      (  ◊  ACTION =  / 'ASSOCIER' ,      [DEFAUT]
                      / 'LIBERER'  ,
         ◊  FICHIER = nomsymb      ,      [K16]

         ◊  UNITE = numul          ,      [I]

         ◊  NOM_SYSTEME  =  fic    ,      [K255]

         ◊  TYPE   =      /  'ASCII'   ,  [DEFAUT]
                          /  'BINARY'  ,
                          /  'LIBRE'   ,
         ◊  ACCES  =      /  'NEW'     ,
                          /  'OLD'     ,
                          /  'APPEND'  ,

         ◊  INFO   =      /  1         ,
                          /  2         ,
```

)

Le caractère obligatoire ou facultatif de certains opérandes dépend de la présence ou de la valeur associée des mots clés renseignés précédemment.

3 Fonctionnement de DEFI_FICHIER

Les entrées/sorties sur les fichiers ASCII depuis le *Code_Aster* sont réalisées à l'aide d'instructions FORTRAN utilisant un entier variant de 1 à 99 dénommé **unité logique**.

Dans le fichier de commandes *Aster* les fichiers en sortie sont désignés par des **noms symboliques**. Le code utilise par défaut les associations suivantes :

- 6 : MESSAGE
- 8 : RESULTAT
- 9 : ERREUR
- 80 : MED
- 95 : CODE
- 0 : VIGILE

Le mot clé IMPRESSION de la procédure DEBUT [U4.11.01] ou de la procédure POURSUITE [U4.11.03] en tête du fichier de commandes peut définir les associations entre des **noms symboliques imposés** de fichiers et leurs **unités logiques** FORTRAN.

L'utilisateur peut souhaiter **modifier** ces associations à l'aide de la procédure DEFI_FICHIER pour créer par exemple de nouveaux noms symboliques de fichiers, pour imprimer certains résultats ou pour les regrouper autrement dans des fichiers. La procédure DEFI_FICHIER permet en outre de désigner directement le fichier de type ASCII qui sera associé à l'unité logique spécifiée. Il peut être précisé soit par un nom en absolu (limité à 255 caractères) si le fichier est localisé sur la machine, soit par un nom relatif dans un répertoire convenu (./REPE_IN ou ./REPE_OUT) lorsque l'interface se charge du transfert distant et global de l'ensemble des fichiers situés sous le répertoire. Un OPEN Fortran nommé est alors réalisé sur les fichiers de type ASCII. La commande permet de plus de se positionner soit en tête de fichier, soit en fin de fichier.

4 Opérandes

La ou les modifications d'association unité logique-nom symbolique porte sur les fichiers de **sortie** et d'**entrée**.

4.1 Opérande ACTION

◊ ACTION = 'ASSOCIER'

Le numéro d'unité logique est associé, lorsque cela est permis, au nom symbolique défini derrière le mot clé NOM.
Il n'est pas possible de redéfinir les associations des numéros logiques 6,8,9 et 95.

◊ ACTION = 'LIBERER'

Le numéro d'unité logique est libéré, il n'est plus possible d'utiliser le nom symbolique, le fichier associé, lorsqu'il est de type ASCII, fait l'objet d'un ordre de fermeture à l'aide de l'instruction Fortran CLOSE. Il devient alors possible de réutiliser le numéro d'unité logique.

Dans ce cas, il est possible d'utiliser l'un ou l'autre des opérandes UNITE ou NOM_SYTEME.

4.2 Opérande FICHIER

◊ FICHIER = nomsymb

Nom symbolique (≤ 16 caractères) que l'on désire **donner** à un fichier. Ce nom doit être placé entre quotes. Ce nom peut être utilisé ensuite dans les commandes Aster qui possèdent l'opérande FICHIER (IMPR_RESU, IMPR_TABLE, IMPR_COURBE, etc).
Cet opérande est obligatoire dans le cas où le fichier est de type ASCII et si NOM_SYSTEME est présent.

4.3 Opérande UNITE

◊ UNITE = numul

Numéro d'unité logique associé.

Il est possible de réutiliser un numéro déjà affecté mais dans ce cas il faut prendre la précaution de libérer ce dernier auparavant. Certains numéros d'unités logiques ne peuvent pas être redéfinis depuis les commandes Aster, il s'agit des numéros 6, 8, 9 et 95 qui sont respectivement alloués aux fichiers MESSAGE, RESULTAT, ERREUR et CODE.

L'opérande UNITE peut parfois être omis, c'est alors le code qui choisira d'affecter un numéro, suivant les disponibilités, il faut alors impérativement préciser l'opérande NOM_SYSTEM, le code se charge ensuite en interne d'associer le numéro d'unité logique et le fichier associé.

4.4 Opérande NOM_SYSTEME

◊ NOM_SYTEME = fic

Nom physique du fichier (≤ 255 caractères) que l'on désire associer à une unité logique. Ce fichier sera créé sous le répertoire d'exécution du code, mais on peut indiquer directement un nom de fichier (respectant les conventions UNIX) dans le répertoire de l'utilisateur. Sous le répertoire d'exécution, il est possible d'utiliser un niveau supplémentaire d'arborescence de nom conventionnel REPE_IN (fichiers de données) ou REPE_OUT (fichiers de résultats) reconnus par l'interface d'accès au code astk. Ce nom doit être placé entre quotes. Bien qu'ils ne soient pas associés à une unité logique par un ordre Fortran OPEN, les fichiers binaires (par exemple MED) peuvent être traités avec ce mécanisme, il faut néanmoins préciser le type d'accès NEW ou OLD pour activer la recopie (par un appel système depuis le code) depuis le répertoire en données ou vers le répertoire en résultat.
Lorsque l'opérande est absent, c'est par défaut le nom de fichier fort.ul où ul est le numéro d'unité qui est associé à l'unité logique défini derrière UNITE.
Pour les fichier de type ASCII, une instruction OPEN Fortran est exécutée sur le nom associé à l'unité logique.

4.5 Opérande TYPE

◊ TYPE = 'ASCII'

Le fichier associé à l'unité logique est de type ASCII.

◊ TYPE = 'BINARY'

Le fichier associé à l'unité logique est de type binaire (non utilisé encore dans le code).

◊ TYPE = 'LIBRE'

Le fichier associé à l'unité logique est de type indéterminé au sens du Fortran, cela permet de gérer de façon plus souple l'accès au fichier, ce type est essentiellement utilisé pour l'accès aux fichiers MED. L'unité logique n'est pas réellement utilisée dans ce cas, mais cela permet d'avoir la convention de nom fort.ul sur le fichier et de pouvoir le transmettre facilement à travers l'interface d'accès au code.

4.6 Opérande ACCES

N'est utilisé que dans le cas des fichiers de type ASCII.

◊ ACCES = 'NEW'

Le fichier est ouvert et on se positionne en tête, une instruction Fortan de type REWIND est exécutée.

◊ ACCES = 'OLD'

Le fichier est ouvert et on se positionne tel quel.

◊ ACCES = 'APPEND'

Le fichier est ouvert et on se positionne en fin de fichier.

4.7 Opérande INFO

◊ INFO = inf

Permet d'imprimer dans le fichier MESSAGE la liste des unités logiques ouvertes avec la commande DEFI_FICHIER ainsi que les paramètres associés. Si INFO = 1, il n'y a pas d'impression.

5 Exemple d'utilisation

Généralement l'utilisateur utilisera DEFI_FICHIER en vue d'effectuer des post-traitements lorsqu'il éprouve le besoin de créer physiquement plusieurs fichiers de résultats en fonction des cas de charges, des grandeurs, des pas ou instants d'évolution du calcul.

5.1 Créer plusieurs fichiers de résultats

Supposons un utilisateur venant d'effectuer un calcul mécanique (calcul de **déplacements** et de **contraintes**) évolutif sur un **certain nombre** de pas de temps. Pour les besoins du dépouillement cet utilisateur désire :

- un fichier des déplacements de la structure à certains instants : déplacements aux instants visualisables par I-DEAS™,
- un fichier des contraintes sur un élément pour visualiser l'évolution des contraintes en fonction du temps (pour tous les instants du calcul, visualisation avec agraf).

Association nom symbolique-unité logique

Par appel à la procédure DEFI_FICHIER l'utilisateur va commencer par créer 2 noms symboliques de fichiers auxquels il va associer 2 unités logiques libres :

- un fichier de nom symbolique 'DEPL_IDEA', d'unité logique 38 pour y déposer les résultats au format IDEAS, le fichier aura pour nom système fort.38 dans le répertoire d 'exécution de *Code_Aster*.

```
DEFI_FICHIER ( FICHIER = 'DEPL_IDEA' , UNITE = 38 )
```

- un fichier de nom symbolique 'CONT_AGRAF', d'unité logique 39 pour y déposer les résultats au format agraf, le fichier aura pour nom système fort.39 dans le répertoire d 'exécution de *Code_Aster*.

```
DEFI_FICHIER ( FICHIER  = 'CONT_AGRAF' , UNITE = 39 )
```

Ecriture dans les fichiers

Ecriture des déplacements au format I-DEAS™ dans le fichier 'DEPL_IDEA' aux instants choisis par l'utilisateur :

```
IMPR_RESU (  MODELE= MONMOD,
             RESU= _F( FORMAT    = 'IDEAS' ,
                       FICHIER   = 'DEPL_IDEA' ,
                       RESULTAT  = MES_RESU,
                       MAILLAGE  = MONMAIL,
                       NOM_CHAM  = 'DEPL' ,
                       LIST_INST = MES_INST ) )
```

La liste des instants choisis par l'utilisateur aura au préalable été créée (sous forme d'un concept de type listr8) par la commande DEFI_LIST_REEL [U4.34.01].

Ecriture des contraintes au format agraf dans le fichier 'CONT_AGRAF' à tous les instants du calcul :

- d'abord récupération, sous forme d'une fonction, dans le résultat MES_RESU, des contraintes calculées sur la composante N de l'élément supporté par la maille MA13, au point de GAUSS n°1 à tous les pas de temps (voir RECU_FONCTION [U4.32.03]) :

```
F = RECU_FONCTION (    RESULTAT   = MES_RESU,
                       NOM_CHAM   = 'SIEF_ELGA' ,
                       NOM_CMP    = 'N' ,
                       GROUP_MA   = MA13,
                       POINT      = 1,
                       TOUT_ORDRE = 'OUI' )
```

- [U4.33.01]) :

```
IMPR_COURBE (FORMAT   = 'AGRAF' ,
                 FICHIER  = 'CONT_AGRAF' ,
                 COURBE   = _F (FONCTION = F ) )
```

L'utilisateur pourra visualiser ce fichier par agraf.

5.2 Déclaration dans l'interface d'accès au code de l'unité logique mise en œuvre dans DEFI_FICHIER

L'utilisateur doit déclarer les noms physiques des fichiers et les unités logiques associées. Cette déclaration s'effectue, dans l'interface astk préalablement au lancement de l'exécution du travail.

Pour l'exemple cité plus haut, l'utilisateur a besoin de :

- créer 2 fichiers (avec leurs noms physiques),
- sélectionner le type **libr** et leur associer le numéro d'unité logique choisi.