

Procédure DEFI_FICHER

1 But

Ouvrir ou fermer un fichier associé à un numéro d'unité logique. Ce numéro peut être indiqué dans la procédure ou bien obtenu en retour de cette dernière. Cette action peut être effectuée à tout moment au cours du travail. Cette procédure permet d'associer un numéro d'unité logique avec un fichier FORTRAN de type ASCII, mais cette notion peut aussi s'appliquer avec certaines restrictions à des fichiers de type binaire.

2 Syntaxe

```
[nfic [entier] = ] DEFI_FICHIER (  
  
    ◇ ACTION      = / 'ASSOCIER',          [DEFAULT]  
                   / 'LIBERER',  
                   / 'RESERVER',  
  
    ◇ FICHIER      = nomfic,                [K255]  
  
    ◇ UNITE        = numul,                 [I]  
  
    ◇ TYPE         = / 'ASCII',            [DEFAULT]  
                   / 'LIBRE',  
                   / 'BINARY',  
  
    ◇ ACCES        = / 'NEW',              [DEFAULT]  
                   / '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.

Cet opérateur possède la particularité de fonctionner à la fois comme une commande en fournissant un résultat de type entier réutilisable derrière un mot clé d'une commande ultérieure, soit comme une procédure.

3 Fonctionnement de DEFI_FICHIER

Les entrées/sorties sur les fichiers ASCII depuis le *Code_Aster* sont réalisées pour la plupart, à l'aide d'instructions FORTRAN utilisant la notion d'**unité logique**, c'est un entier variant de 1 à 99 associé à l'aide d'une instruction de type « open » à un fichier. Par défaut ce numéro xx est associé au nom fort.xx, mais il est possible de l'associer à un nom de fichier quelconque. Le numéro d'unité logique est utilisé dans l'interface d'accès au code astk pour recopier les fichiers en entrée et en sortie.

Dans le fichier de commandes *Aster* les différents opérateurs effectuant des lectures ou des écritures admettent le mot clé `UNITE` comme argument.

L'unité logique 6 est réservée par le code lors de la phase d'initialisation. Il n'est pas autorisé de modifier ce numéro d'unité logique 6 qui permet de tracer le déroulement d'une exécution.

unité logique	nom du fichier	modification autorisée
6	MESSAGE	non

L'utilisateur peut souhaiter ajouter ou **modifier** des associations à l'aide de la procédure `DEFI_FICHIER` pour utiliser par exemple de nouveaux noms 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 (type `repe` dans `astk`). 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.

Le code gère en interne une structure de données regroupant l'ensemble des associations unité logique – nom de fichier – type de fichier - type d'accès.

4 Opérandes

La ou les modifications d'association unité logique-nom de fichier 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 défini derrière le mot clé `FICHIER` s'il est renseigné, au nom `fort.xx` sinon.

Il n'est pas possible de redéfinir les associations des numéros logiques 6 et 9.

◇ ACTION = 'LIBERER'

Le numéro d'unité logique est libéré, il n'est plus licite d'utiliser ce numéro d'unité, 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 avec une action de type 'LIBERER'. Ce mécanisme est indispensable si l'on veut pouvoir ré-exploiter immédiatement le contenu du fichier associé dans le fichier de commandes en cours, en effet les buffers doivent avoir été complètement vidés et le fichier doit pouvoir être accessible, éventuellement lors d'un appel à un logiciel depuis une commande python de type `os.system`.

◇ ACTION = 'RESERVER'

Ce type d'action est utilisée dans les macros commandes et permet d'éviter les conflits de numéro d'unité logique entre Fortran et python..

Le numéro d'unité logique est associé, lorsque cela est permis, au nom défini derrière le mot clé `FICHIER` s'il est renseigné, au nom `fort.xx` sinon. Il n'y a pas d'instruction de type `OPEN` exécutée, à charge pour la macro commande d'effectuer les différentes actions nécessaires.

4.2 Opérande FICHIER

◇ `FICHIER = nomfic`

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 fichiers de type ASCII, une instruction `OPEN` Fortran est exécutée sur le nom associé à l'unité logique.

4.3 Opérande UNITE

◇ `UNITE = numul`

Numéro d'unité logique associé, sa valeur pour les fichier ASCII ouverts par des instructions est comprise entre 1 et 99 inclus.

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 et 9 qui sont respectivement alloués aux fichiers `MESSAGE` et `ERREUR`.

Ce numéro peut être utilisé ensuite dans toutes les commandes Aster qui possèdent l'opérande `UNITE` (`IMPR_RESU`, `IMPR_TABLE`, `IMPR_FONCTION`, etc).

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 `FICHIER`, le code se charge ensuite en interne d'associer le numéro d'unité logique et le fichier associé. Il est alors possible de récupérer une valeur entière en sortie de l'opérateur.

4.4 Opérande TYPE

◇ `TYPE = 'ASCII'`

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

◇ `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.

◇ `TYPE = 'BINARY'`

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

4.5 Opérande ACCES

◇ ACCES = 'NEW'

Le fichier de type « ASCII » est ouvert et on se positionne en tête, une instruction Fortan de type **REWIND** est exécutée. S'il existe, le fichier est écrasé. À utiliser pour un fichier en résultat.

◇ ACCES = 'OLD'

Le fichier de type « ASCII » est ouvert et on se positionne tel quel, c'est à dire éventuellement à la position à laquelle on se situait lors du dernier accès.
À utiliser pour un fichier en donnée.

◇ ACCES = 'APPEND'

Le fichier est ouvert et on se positionne en fin de fichier. N'est utilisé que dans le cas des fichiers de type ASCII.
À utiliser pour un fichier en donnée que l'on veut compléter (qui sera donc a priori en résultat)..

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

Généralement l'utilisateur appellera **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.

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. Il faut ajouter les fichiers dans le profil d'étude en sélectionnant le type **libr** et leur associer le numéro d'unité logique choisi.

Le type **repe** est utilisé dans l'interface **astk** pour transmettre ou recevoir tout le contenu d'un répertoire de fichier, à charge pour l'utilisateur de faire appel dans le fichier de commande à **DEFI_FICHIER** pour effectuer l'association avec le numéro d'unité logique.

Pour convention les fichiers en données sont transmis dans le répertoire de nom local **REPE_IN**, les fichiers en résultat sont transmis dans le répertoire de nom local **REPE_OUT**.

Dans la commande **DEFI_FICHIER**, le nom passé derrière **FICHIER** est de la forme :
./REPE_IN/mon_fichier.

Pour un fichier en donnée, on fera par exemple :

```
DEFI_FICHIER(FICHIER='./REPE_IN/file.data',  
            ACCES='OLD',  
            UNITE=11)  
LIRE_FONCTION(UNITE=11, ...)
```

Remarque

INCLUDE ne peut pas être utilisé avec une unité logique associée à un nom de fichier particulier.