Procédure IMPR_CONCEPT

1 But

Imprimer sous une forme visualisable des champs de « données » à partir de concepts affectées par l'utilisateur avec les commandes AFFE_MATERIAU , AFFE_CARA_ELEM et AFFE_CHAR_MECA .

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

2 Syntaxe

```
IMPR_CONCEPT     (

◊  UNITE  =  unit,
◊  FORMAT =  /  'MED',                                          [DEFAUT]
             /  'RESULTAT',
◊  INFO =    / 1,                                               [DEFAUT]
             / 2,
Si format = 'MED' :
{
   ◊  VERSION_MED  =  /  '3.3.1',                               [DEFAUT]
                      /  '4.0.0',
                      /  '4.1.0',
}

♦  CONCEPT   =  _F(
   /  CHAM_MATER   = chmat,                                     [cham_mater]
   /  CARA_ELEM    = carele,                                    [cara_elem]
      ◊  REPERE_LOCAL           =  / 'NON' ,                    [DEFAUT]
                                   / 'ELEM',
                                   / 'ELNO',
      # si REPERE_LOCAL = 'ELEM' ou 'ELNO',
            ♦     MODELE    = mo                               [modele]
   /  CHARGE       = charg,                                     [charge]
      )
)
```


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

3 Typographie dans la documentation IMPR_CONCEPT

La procédure IMPR_CONCEPT permet d'imprimer sous une forme visualisable les quantités affectées par l'utilisateur avec les commandes AFFE_MATERIAU , AFFE_CARA_ELEM et AFFE_CHAR_MECA . Cela permet de vérifier que les règles de surcharge du code conduisent aux affectations souhaitées.

Par exemple :

```
IMPR_CONCEPT( FORMAT='MED',
              CONCEPT=(
                       _F(CHAM_MATER = CHAMPMAT),
                       _F(CARA_ELEM  = CARA_ELE),
                       _F(CHARGE     = CHARG1),),
            )
```

Les champs contenus dans ces structures de données sont imprimées sous deux formes très différentes :

- La forme "A" (la plus facile à interpréter graphiquement) : chaque composante est imprimée séparément comme un nombre réel. Par exemple, l'épaisseur des éléments de coque est visualisable comme un champ scalaire (scalar map dans Salomé).
- La forme "B" : on affecte à chaque maille un code entier : 1, 2, 3, ...n. Les mailles affectées par le même code ont alors TOUTES leurs composantes identiques. La "définition" des codes, c'est à dire les valeurs de ces composantes est imprimée dans le fichier message. On peut visualiser les "codes" comme un champ scalaire, ce qui permet de "voir" les zones où "tout est constant".

La forme "A" est utilisée systématiquement pour le format "MED", sauf pour le champ de matériaux car ce champ contient le nom du matériau affecté sur les mailles et ce nom n'est pas un nombre.
La forme "A" n'est pas programmée au format "RESULTAT".

La forme "B" est systématiquement utilisée pour le format "RESULTAT" et pour le champ de matériaux.

Pour la forme "B", la correspondance entre les quantités affectées et les code est donnée dans le fichier .mess.

Par exemple, pour le champ de matériaux :

```
IMPRESSION D'UN CHAMP DE CONCEPT : Champ de MATERIAUX
NOM DU CHAMP : CHAMPMAT_CHAMP_MAT
CORRESPONDANCE VALEUR <-> CONTENU :
VALEUR =          1.
    X1        = MAT_1
VALEUR =          2.
    X1        = MAT_2
VALEUR =          3.
    X1        = MAT_3
```

4 Opérandes FORMAT et UNITE et INFO

4.1 Opérande FORMAT

L'opérande FORMAT permet de spécifier le format du fichier où écrire le résultat.
Deux formats 'MED' et 'RESULTAT' sont disponibles. Le format 'MED' est le format d'écriture par défaut.

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

4.2 Opérande UNITE

Définit dans quelle unité on écrit le fichier med. Par défaut, UNITE = 80 et correspond à l'unité par défaut du type rmed dans astk.

4.3 Opérande INFO

Le mot-clé INFO lorsqu'il est égal à 2 permet d'obtenir des informations sur les impressions réalisées par la commande.

4.4 Opérande VERSION_MED

```
◊  VERSION_MED  =  /'3.3.1',                    [DEFAUT]
                   /'4.0.0',
                   /'4.1.0',
```

Lors de la création d'un nouveau fichier au format med, l'impression est faite au format med 3.3.1. Si le fichier existe déjà, le niveau de format med est conservé.
On peut changer la version de fichier MED avec le mot-clef VERSION_MED .

5 Mot-clé facteur CONCEPT

5.1 Opérande REPERE_LOCAL

Si REPERE_LOCAL a la valeur 'ELEM' ou 'ELNO', le mot-clé MODELE est obligatoire.

Si REPERE_LOCAL a la valeur 'ELEM', les 3 vecteurs constituant le repère local de chaque élément sont imprimés, pour pouvoir être visualisés à l'aide de ParaVis.

Si REPERE_LOCAL a la valeur 'ELNO', les 3 vecteurs constituant le repère local de chaque élément sont enregistrés sous forme de champ aux nœuds, pour pouvoir être ensuite utilisés dans la calculatrice de ParaVis en combinaison avec d'autres champs aux nœuds. Dans ce cas aucune autre information venant du concept CARA_ELEM n'est enregistré dans le fichier.

Lorsque REPERE_LOCAL='ELNO' il est donc possible de combiner les vecteurs locaux avec les composantes des champs d'efforts internes. Cela permet de visualiser les vecteurs efforts en 3D, ainsi que de réaliser une animation de leur évolution au cours des instants de calcul. Pour réaliser cette action plusieurs opérations élémentaires sont à réaliser dans ParaVis :

- Ouvrir le fichier MED contenant les repères :
  sur les REPE -> Filtre "ELNO Points" -> Filtre "Merge blocs"
- Ouvrir le fichier MED contenant les champs :
  sur les EFGE_ELNO -> Filtre "ELNO Points" -> Filtre "Merge blocs"
- sélection des 2 "Merge blocs" puis "Append Attributes"

Dans "Calculator" on a accès aux vecteurs des REPE et aux composantes des champs.
Le calcul du vecteur effort (N, Vy, Vz) est fait de la façon suivante :

$Fint$ = CAREL___.REPLC_1∗xxxxxxxxEFGE_ELNO_N +
CAREL___.REPLC_2∗xxxxxxxxEFGE_ELNO_VY +
CAREL___.REPLC_3∗xxxxxxxxEFGE_ELNO_VZ

où 'xxxxxxxx' désigne le nom du concept résultat, produit par *Code_Aster*.

Le calcul du vecteur Moment (Mt, My, Mz) est fait de la façon suivante :

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

$Mint$ = CAREL___.REPLC_1∗xxxxxxxxEFGE_ELNO_MT +
CAREL___.REPLC_2∗xxxxxxxxEFGE_ELNO_MFY +
CAREL___.REPLC_3∗xxxxxxxxEFGE_ELNO_MFZ

où 'xxxxxxxx' désigne le nom du concept résultat, produit par *Code_Aster*.

5.2 Champs pouvant être visualisés

CHAM_MATER :
Champ de matériaux

CARA_ELEM :
Caractéristiques générales des barres
Caractéristiques géom. des barres
Caractéristiques générales des poutres
Caractéristiques géom. des poutres
Caractéristiques des cables
Caractéristiques des poutres courbes
Caractéristiques des poutres "fluides"
Caractéristiques des éléments discrets K_*
Caractéristiques des éléments discrets M_*
Caractéristiques des éléments discrets A_*
Caractéristiques géom. des coques
Orientation des éléments 2D et 3D
Orientation des coques et des poutres

CHARGE :
Chargement de PESANTEUR
Chargement de ROTATION
Chargement de PRES_REP
Chargement de forces volumiques en 3D
Chargement de forces surfaciques en 3D
Chargement de forces linéiques en 3D
Chargement de forces surfaciques en 2D
Chargement de forces linéiques en 2D
Chargement de forces réparties pour les coques
Chargement de PRE_EPSI
Chargement de FORCE_ELEC
Chargement de FLUX_THM_REP
Chargement d'IMPE_FACE
Chargement d'ONDE_FLUI

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