Procédure TEST_RESU

1 But

Comparer une valeur extraite d'une structure de données à une valeur de référence fournie par l'utilisateur.

Cette commande permet de tester une valeur numérique : entier, réel ou complexe extraite d'un concept déjà calculé. Aujourd'hui, on peut tester une composante d'un cham_no ou d'un cham_elem, une composante d'un champ extrait d'un resultat, un paramètre d'un resultat, une valeur 'globale' extraite d'un champ ou bien le contenu d'un objet quelconque d'un concept utilisateur.

La procédure écrit alors un message conventionnel :

- "OK" (si c'est bon),
- "NOOK" (sinon).

suivi de la valeur trouvée, la valeur de référence et le pourcentage d'erreur.
Systématiquement, on vérifie une valeur de non régression, et quand cela est possible, une valeur de référence analytique, provenant d'une source externe ou d'un autre calcul avec Code_Aster.

Les commandes TEST_FONCTION [U4.92.02] et TEST_TABLE [U4.92.03] permettent de tester les valeurs extraites des fonctions et des tables.

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

2 Syntaxe

```
TEST_RESU (

  ♦ /  CHAM_NO=(_F  (  ♦  CHAM_GD = chno,                  [cham_no]
                       /  TYPE_TEST =  /  'SOMM_ABS' ,
                                       /  'SOMM' ,
                                       /  'MAX' ,
                                       /  'MIN' ,
                          ◊  NOM_CMP =    ncmp,            [K8]
                       /  ♦  GROUP _NO =  grno,         [group_no]
                          ♦  NOM_CMP  =   nomcmp ,         [K8]
                          ◊  LEGENDE =    legende,         [K16]
                       #  Voir définition de la valeur de référence
                    ),)
       /  CHAM_ELEM=(_F(  ♦  CHAM_GD = chel,                  [cham_elem]
                       /  TYPE_TEST =  /  'SOMM_ABS' ,
                                       /  'SOMM' ,
                                       /  'MAX' ,
                                       /  'MIN' ,
                          ◊  NOM_CMP =    ncmp,            [K8]
                       /  ♦  GROUP_MA  =  gma1   ,      [group_ma]
                          ♦  /  POINT    = nupoint,        [I]
                             /  GROUP_NO = grno,          [group_no]
                             ◊  SOUS_POINT = nusp,
                          ♦  NOM_CMP  = nomcmp,            [K8]
                          ◊  LEGENDE =    legende,         [K16]
                          #  Voir définition de la valeur de référence
                    ),),

       /  CARTE=(_F (  ♦  CHAM_GD = chel,                  [cham_elem]
                       ♦  NOM_CMP = nomcmp,               [K8]
                       ♦  GROUP_MA  =  gma1 ,          [group_ma]
                       ◊  LEGENDE =    legende,           [K16]
                       #  Voir définition de la valeur de référence
                    ),),

       /  MAILLAGE=(_F(♦  MAILLAGE =   ma,                [maillage]
                       ♦  CARA =
                             /  'NB_MAILLE'
                             /  'NB_NOEUD'
                             /  'NB_GROUP_MA'
                             /  'NB_NB_GROUP_NO'
                             /  'EXI_GROUP_MA'
                                ♦  NOM_GROUP_MA  = gma ,  [group_ma]
                             /  'EXI_GROUP_NO'
                                ♦  NOM_GROUP_NO  = gno ,  [group_no]
                       #  Voir définition de la valeur de référence
                    ),),

       /  RESU =(_F (  ♦  RESULTAT =   res ,           [resultat_sdaster]
                       ♦  /  NUME_ORDRE = nuor,             [I]
```


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

```
                        /   NUME_MODE  = numo,                [I]
                        /   INST       = inst,                [R]
                        /   FREQ       = freq,                [R]
                        /   NOEUD_CMP  = (noeud, cmp),        [l_Kn]
                        /   NOM_CAS    = nocas,               [Kn]
                        /   ANGLE       = α ,                 [R]

                    ♦   /   PARA     = para,                  [K16]
                        /   NOM_CHAM = nosymb,                [K16]

                            ◊ /NOM_CMP =    ncmp,             [K8]
                              /NOM_VARI =   nvari,            [K16]

                            /   TYPE_TEST =  /   'SOMM_ABS' ,
                                             /   'SOMM' ,
                                             /   'MAX' ,
                                             /   'MIN' ,

                            /   ♦   GROUP_NO  = grno,       [group_no]
                                    /   ♦   GROUP_MA= gma1, [group_ma]
                                        ♦   /   POINT = nupoint, [I]
                                            /   GROUP_NO = grno, [group_no]
                                            ◊   SOUS_POINT = nusp,
                                        ♦   NOM_CMP  = nomcmp,      [K8]
                        #   Voir définition de la valeur de référence
                    ),),
/   GENE =(_F (   ♦     RESU_GENE =  res,                  [VECT_ASSE_GENE]
                        NUME_CMP_GENE = ncmp,                 [I]

                        ~   RESU_GENE =  res,                 [MODE_GENE]
                        ♦   /   PARA     = para,              [K16]
                            /   NOM_CHAM = nosymb,            [K16]
                                NUME_CMP_GENE = ncmp,         [I]

                            ♦   /   NUME_ORDRE = nuor,        [I]
                                /   NUME_MODE = numo,         [I]
                                /   FREQ       = freq,        [R]

                            RESU_GENE =  res,                 [HARM _GENE]
                            NOM_CHAM = nosymb,                [K16]
                            NUME_CMP_GENE = ncmp,             [I]

                            ♦   /   NUME_ORDRE = nuor,        [I]
                                /   FREQ       = freq,        [R]

                            RESU_GENE =  res,                  [TRAN_GENE]
                            NOM_CHAM = nosymb,                [K16]
                            NUME_CMP_GENE = ncmp,             [I]

                            ♦   /   NUME_ORDRE = nuor,        [I]
                                /   INST       = inst,        [R]

                        #   Voir définition de la valeur de référence
                    ),),
```


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

```
/   OBJET =(_F(
                  ♦  NOM    =  nomobj ,                    [K24]

                  #  Voir définition de la valeur de référence
               ),),

/   TEST_NAN =          /   'NON' ,                        [DEFAUT]
                        /   'OUI' ,
)

# Définition de la valeur de référence

    ♦  /  VALE_CALC   = val   ,                            [R]
       /  VALE_CALC_C = val   ,                            [C]
       /  VALE_CALC_I = val   ,                            [I]
       /  VALE_CALC_K = val   ,                            [K*]
     # si VALE_CALC = 0.
    ◊  ORDRE_GRANDEUR = ordgrd ,                           [R]

    ◊  LEGENDE =    legende,                               [K16]

    ◊  VALE_ABS =   /   'NON' ,                            [DEFAUT]
                    /   'OUI' ,
    ◊  /  |  TOLE_MACHINE =  /  1.0D-6 ,                   [DEFAUT]
                             /  prec   ,                   [R]
          |  CRITERE =       /  'RELATIF' ,                [DEFAUT]
                             /  'ABSOLU'  ,
       # uniquement pour RESU ou GENE
       /  |  TOLE_MACHINE =  /  (prec1, prec2),            [l_R]
          |  CRITERE      =  /  (crit1, crit2),            [l_Kn]

    # sauf dans TEST_FICHIER
    ◊  REFERENCE =  /  'ANALYTIQUE' ,
                    /  'SOURCE_EXTERNE' ,
                    /  'AUTRE_ASTER' ,

    Si REFERENCE est renseigné
       ♦  /  VALE_REFE   = val   ,                         [R]
          /  VALE_REFE_C = val   ,                         [C]
          /  VALE_REFE_I = val   ,                         [I]
          /  VALE_REFE_K = val   ,                         [K*]
       ◊     PRECISION =  /  1.0D-3 ,                      [DEFAUT]
                          /  prec   ,                      [R]
```

Remarque 1

La définition de la valeur de référence est commune aux commandes TEST_RESU, TEST_TABLE, TEST_FONCTION *et* TEST_FICHIER. Tous les mots-clés ne sont pas disponibles dans toutes les commandes, voir les commentaires.
Le type entier (_I) n'existe pas dans TEST_FONCTION et TEST_FICHIER. Le type complexe (_C) n'existe pas dans TEST_FICHIER et pour OBJET. Le type chaine (_K) n'existe que dans TEST_FICHIER et TEST_TABLE.
VALE_ABS n'existe pas pour CARTE et TEST_FICHIER.

Remarque 2

Les mots clés CHAM_NO, CHAM_ELEM et CARTE *permettent de tester les* cham_no , *les* cham_elem *et les* carte. *Le mot clé* RESU est réservé aux *concepts de type* resultat. Le mot clé MAILLAGE permet de tester quelques caractéristiques (entières) d'un maillage.

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

3 Généralités

Cette commande permet de tester une valeur numérique scalaire récupérée dans un concept de type cham_no, cham_elem, carte ou resultat, par rapport à une valeur de non régression et, quand c'est possible, par rapport à une valeur de référence analytique, provenant d'une source externe ou d'un autre calcul avec Code_Aster.

Trois types de valeurs numériques peuvent être testées :

- une composante d'un champ (cham_no, cham_elem, carte ou champ faisant partie d'un resultat),
- un paramètre contenu dans un concept de resultat ,
- une valeur globale d'un champ [§4.4].

Pour tester une composante de champ, il faut choisir un champ [§4.1] puis choisir une composante [§4.3].

Pour tester un paramètre, il faut choisir un numéro d'ordre [§4.1.4] et choisir le nom du paramètre.

La valeur numérique attendue (réelle, complexe ou entière) est fournie conformément à [§4.7].

Remarque concernant les tests de non régression

On fait systématiquement un test de non régression par rapport à une valeur calculée précédemment. La tolérance associée à ce test (TOLE_MACHINE) doit être très faible et ne devrait pas être supérieure à la valeur par défaut. En particulier, la valeur devrait être la même (sur au moins 8 décimales) sur toutes les plates-formes. De plus, cette valeur calculée ne doit changer que lorsque de l'algorithme est modifié, corrigé. Tout autre variation doit alertée le développeur sur la fiabilité de la programmation.

Remarques concernant les tests dans les structures de données "généralisées" :

On peut tester les composantes généralisées (déplacements, vitesses ou accélérations d'un transitoire dans l'espace modal). Il convient néanmoins d'être circonspect avec ce type de test. En effet la valeur d'une composante généralisée dépend entièrement de la norme du mode. Or celle-ci est déterminée de manière arbitraire. Donc sans normalisation préalable des normes, la valeur d'une grandeur généralisée est arbitraire. Enfin, il n'y pas de possibilité dans Code_Aster de fixer la direction d'un mode. Pour un mode multiple, cela veut dire que, même une fois les modes normés, une grandeur généralisée peut prendre une valeur quelconque. Dans le cas d'un mode simple, il peut être orienté dans une direction ou dans la direction opposée. On obtient alors une valeur généralisée ou son opposée.

4 Opérandes

4.1 Sélection d'un champ

Afin de tester un champ qui peut être un champ isolé (cham_no ou cham_elem ou carte), ou un champ extrait d'un résultat, ou utilisera les mot-clés facteurs : CHAM_NO, CHAM_ELEM, CARTE ou RESU.

4.1.1 Champs "isolés"

♦ CHAM_GD = champ

Nom du cham_no , cham_elem ou carte dont on veut extraire une valeur.

4.1.2 Opérande RESULTAT

♦ RESULTAT = res

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

Nom du concept resultat traité.

4.1.3 Opérande NOM_CHAM

♦ / NOM_CHAM = nosymb

Nom symbolique du champ à sélectionner.

4.1.4 Sélection d'un numéro d'ordre

♦ / NUME_ORDRE = nuor,

Numéro d'ordre du champ (ou du paramètre) recherché.

/ NUME_MODE = numo,
/ INST = inst,
/ FREQ = freq,
/ NOEUD_CMP = (noeud, cmp),
/ NOM_CAS = nocas,
/ ANGLE = α ,

Ces mots clés permettent d'identifier un numéro d'ordre dans un resultat [U4.71.00].

On les appelle des "variables d'accès".

Ils ne sont pas tous valables pour tous les types de resultat.

Lorsque l'accès se fait pas une valeur réelle (ANGLE, FREQ, INST) la valeur donnée ne doit pas être ambigüe (cf [§4.7]).

4.2 Sélection d'un paramètre dans un résultat

Pour sélectionner un paramètre dans un résultat, il faut préciser le numéro d'ordre voulu [§4.1.4] et donner le nom du paramètre.

♦ PARA = para

Nom du paramètre cherché. Ce nom est attaché au type du concept resultat traité.

4.3 Sélection d'une composante d'un champ

L'accès à une grandeur se fait pour un cham_no par :

- le nom d'un groupe de nœuds contenant le nœud qui porte cette grandeur (et lui seul).

L'accès à une grandeur se fait pour un cham_elem par :

- le nom d'un groupe de mailles contenant la maille qui supporte l'élément (et elle seule),
- quelque chose qui précise :
  - soit le nom d'un groupe contenant un nœud de cette maille pour les cham_elem "aux nœuds" (ELNO).
  - soit le numéro du point de GAUSS pour les cham_elem "aux points de GAUSS" (ELGA).

L'accès à une grandeur se fait pour une carte par :

- lle nom d'un groupe de mailles contenant la maille ,

♦ GROUP_MA = gma1

Permet de désigner la maille pour laquelle on veut tester le cham_elem ou la carte.
Le groupe ne doit contenir qu'une seule maille.

♦ GROUP_NO = grno

Nom du groupe contenant le nœud dont on veut vérifier une composante.
Le groupe doit être réduit à un seul nœud.

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

/ POINT = nupoint

L'entier nupoint précise le numéro du point de GAUSS dont on veut tester la valeur (cas des cham_elem "aux points de GAUSS").

◊ SOUS_POINT = nusp

L'entier nusp précise le numéro du sous-point duquel on souhaite obtenir la valeur (cas des cham_elem à sous-points, utilisés par les éléments de structure : poutre, tuyaux, coques).
Dans le cas des plaques et des coques multi-couches, le numéro du sous-point correspond au niveau dans l'ensemble des couches. Chaque couche est décrite par une peau inférieure, moyenne et supérieure. Par convention, pour N couches, ce numéro varie entre 1 et $3N$ où le premier point se situe au niveau de la peau inférieure de la première couche et le $3N$ ème point au niveau de la peau supérieure de la dernière couche (cf. [R3.07.03] et [R3.07.04] pour la numération des couches).

Dans le cas des poutres multi-fibres, cet entier est le numéro de la fibre dont la numérotation est décrite dans la documentation [U4.26.01] et [R3.08.08].

Dans le cas des tuyaux, il faut se référer à la description faite dans le document [R3.08.06].

/ NOM_CMP = ncmp

Nom de la composante que l'on veut tester [U2.01.04].

/ NOM_VARI = nvari

Pour les champs des variables internes (VARI_*), on peut donner le nom de la variable interne (voir [U4.51.11] pour les règles de nommage des variables internes) au lieu du nom de la composante (V1, ...).

4.4 Tester un champ "globalement"

Une fois un champ sélectionné [§4.1], on peut tester une quantité calculée globalement sur tout le champ. Pour cela, NOM_CMP ne doit pas être renseigné pour permettre de prendre en compte toutes les composantes du champ.

/ TYPE_TEST = 'SOMM_ABS'

La somme des valeurs absolues des composantes du champ.

/ TYPE_TEST = 'SOMM'

La somme des valeurs des composantes du champ.

/ TYPE_TEST = 'MAX'

Le maximum des valeurs des composantes du champ.

/ TYPE_TEST = 'MIN'

Le minimum des valeurs des composantes du champ.

4.5 Tester le contenu d'un objet JEVEUX

Cette fonctionnalité est réservée aux développeurs du Code. Pour l'utiliser, il faut connaître les noms des objets JEVEUX composant les concepts de l'utilisateur. Elle est destinée à vérifier la non régression des structures de données autres que les RESULTAT, CHAMPS, TABLE et FONCTION.

4.5.1 Opérande NOM

NOM = nomobj

Nom de l'objet jeveux que l'on veut tester.

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

Remarque :

L'objectif de ce type de test (NOM + VALE_CALC/_I) est de tester globalement tout un vecteur. La "somme" qui est testée est malheureusement un mauvais "check sum" de l'objet : une permutation au sein du vecteur ne change pas cette somme. Un test plus prudent consiste à imprimer l'objet dans un fichier (IMPR_CO) puis à tester le contenu de ce fichier avec un vrai "check sum" (TEST_FICHIER).

4.5.2 Opérandes CRITERE et PRECISON

Voir ci-dessous [§4.7].

4.6 Mot clé facteur MAILLAGE.

Ce mot clé permet de valider les commandes du code qui produisent (ou modifient) des maillages.
Il permet de tester quelques caractéristiques (entières) des maillages : nombre de mailles, de nœuds, de groupes de mailles, de groupes de nœuds, nombre de mailles dans un groupe de mailles et nombre de nœuds dans un groupe de nœuds.

4.6.1 Opérande MAILLAGE

MAILLAGE = mailla

Nom du maillage que l'on veut tester.

4.6.2 Opérande CARA

Permet de choisir la caractéristique du maillage à tester : 'NB_MAILLE', 'NB_NOEUD', 'NB_GROUP_MA', 'NB_GROUP_NO', 'EXI_GROUP_MA' ou 'EXI_GROUP_NO'.

4.6.3 Opérandes NOM_GROUP_MA et NOM_GROUP_NO

Ces opérandes sont à utiliser pour CARA='EXI_GROUP_MA' (resp. CARA='EXI_GROUP_NO').
Il permettent de désigner le groupe que l'on veut tester.
Dans ce cas, la valeur testée (VALE_CALC_I ou VALE_REFE_I) doit contenir le nombre de mailles (ou de nœuds) du groupe.

4.7 Définition de la valeur de non régression et de référence

On fait systématiquement un test de non régression par rapport à une valeur précédemment calculée, avec une tolérance très faible : mots-clés VALE_CALC et TOLE_MACHINE.
Aussi souvent que possible, on ajoute un test par rapport à une valeur de référence par rapport à une solution analytique, une valeur obtenue d'une source externe ou une autre modélisation : mots-clés REFERENCE, VALE_REFE, PRECISION.

♦ / VALE_CALC = val

Valeur réelle de non régression. C'est la valeur calculée par Code_Aster.
Lorsque cette valeur est nulle (inférieure à 1.e-16 en valeur absolue), il faut renseigner le mot-clé ORDRE_GRANDEUR = ordgrd. Il n'a de sens qu'en absolu.
On vérifie alors que : $|val| \leq tole * ordgrd$
Si ORDRE_GRANDEUR n'est pas fourni alors que VALE_CALC est nul, le test de non régression est ignoré (apparaît avec SKIP dans le fichier résultat). Attention dans ce cas, un test avec VALE_REFE est obligatoire.

/ VALE_CALC_C = val

Valeur complexe de non régression.

/ VALE_CALC_I = val

Valeur entière de non régression.

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

/ VALE_CALC_K = val

Chaîne de caractères de non régression. Uniquement par TEST_FICHIER et TEST_TABLE.

◊ TOLE_MACHINE = tole

Précision demandée (par défaut 1.D-6) pour accepter la valeur calculée par rapport à la valeur de non régression (VALE_CALC). Voir ORDRE_GRANDEUR ci-dessus.

◊ VALE_ABS

= 'NON' la valeur de référence et la valeur calculée par Aster sont comparées telles quelles.
= 'OUI' la valeur de référence et la valeur calculée par Aster sont comparées en valeurs absolues.

◊ CRITERE =

Type de test à effectuer. S'applique au test de non régression et au test par rapport à une valeur de référence le cas échéant.

Si v est la valeur extraite, le test portera pour :

- 'RELATIF' sur : $|val-v| \leq prec \cdot |val|$
- 'ABSOLU' sur : $|val-v| \leq prec$

◊ REFERENCE =

/ 'ANALYTIQUE' : la valeur de référence fournie est "analytique"

/ 'SOURCE_EXTERNE' : la valeur de référence fournie provient d'un programme autre que Code_Aster , d'une référence bibliographique, d'une mesure, etc.

/ 'AUTRE_ASTER' : la valeur de référence fournie est celle obtenue par un autre calcul avec Code_Aster (autre commande, autre modélisation, option de calcul, ...)

C'est la présence du mot-clé REFERENCE qui indique que l'on dispose d'une référence externe et conditionne la présence des mots-clés VALE_REFE[_I/_C] et PRECISION.

♦ VALE_REFE, VALE_REFE_C, VALE_REFE_I, VALE_REFE_K

Similaires aux mots-clés VALE_CALC ci-dessus. Ils définissent la valeur obtenue par la source externe.

◊ PRECISION =

Précision demandée (par défaut 1.D-3) pour accepter la valeur calculée par rapport à la valeur de référence (VALE_REFE).

Remarque :

Lorsque la définition du numéro d'ordre d'un RESULTAT se fait par une variable d'accès réelle (FREQ , INST , ANGLE), il ne faut pas qu'il y ait ambiguïté sur ce numéro d'ordre. Pour cela l'utilisateur définit un petit intervalle autour de la valeur demandée grâce aux mots clés CRITERE et TOLE_MACHINE .

Dans ce cas (accès "réel") les mots clés CRITERE et TOLE_MACHINE attendront donc 2 valeurs chacun : (crit1 , crit2) et (prec1 , prec2).

crit1 et prec1 concernent la valeur de non régression.

crit2 et prec2 permettent de choisir l'intervalle de recherche du numéro d'ordre.

Les valeurs par défaut de crit1 et prec1 sont 'RELATIF' et 1.D-6.

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

Les valeurs par défaut de crit2 et prec2 sont 'RELATIF' et 1.D-3.

On ne peut définir explicitement crit2 et prec2 sans définir crit1 et prec1.

Remarque :

Les tests de non-régression (mots-clés VALE_CALC) sont systématiquement ignorés dans les tests de validation mais ils doivent quand même être renseignés (à une valeur quelconque) car requis par le catalogue de la commande. Les tests de validation sont identifiés comme tels par la présence de testlist validation dans leur fichier .export.*
I l y a toutefois une exception pour les tests sur les chaînes de caractères (VALE_CALC_K) qui ne sont pas ignorés car il n'y a pas de tolérance/précision sur ces tests.

4.8 Ajout d'une spécificité à la valeur testée

◊ LEGENDE =

Chaîne de caractères d'au plus 16 caractères décrivant le test effectué.
L'utilisateur a donc la possibilité de commenter son test.

4.9 Mot-clé TEST_NAN

| TEST_NAN = / 'NON' , [DEFAUT]
/ 'OUI' ,

Ce mot-clé sert à valider le fonctionnement du NaN (*Not-a-Number*) de Code_Aster. Ce mot-clé est à utiliser uniquement dans l'optique de mener des tests. Son utilisation doit provoquer une erreur fatale en FPE (*Floating Point Exception*) en mode *debug* dans TEST_RESU.

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

5 Exemples

```
TEST_RESU(   CHAM_NO = _F  (  CHAM_GD   = f,
                              GROUP_NO  = 'GN2',
                              NOM_CMP   = 'DX' ,
                              REFERENCE = 'ANALYTIQUE' ,
                              VALE_CALC = 1.1999996845E-5 ,
                              VALE_REFE = 1.2E-5 ,
                              PRECISION = 1D-4 ,))

TEST_RESU(   CARTE  =_F    (  CHAM_GD   = CART1,
                              GROUP_MA  = 'GM3',
                              NOM_CMP   = 'X4' ,
                              REFERENCE = 'AUTRE_ASTER',
                              VALE_CALC_I = 3,
                              VALE_REFE_I = 3,))

TEST_RESU(   CHAM_ELEM=(_F(  CHAM_GD   = SIGGA,
                              GROUP_MA   = ' G M3',
                              POINT     = 3,
                              NOM_CMP   = 'SIXX' ,
                              VALE_CALC = 3.4E6,     ),

                         _F (  CHAM_GD   = SIGNO,
                              GROUP_MA  = 'GM5',
                              GROUP_NO  = 'GN1',
                              NOM_CMP   = 'SIXX' ,
                               VALE_CALC = 3.4E6  ,  ),))

TEST_RESU (  RESU =(_F     (  RESULTAT  = evolth,
                              NOM_CHAM  = 'FLUX_ELGA' ,
                              TYPE_TEST = 'MAX' ,
                              REFERENCE = 'ANALYTIQUE',
                              VALE_CALC = 154.35000201404,
                              VALE_REFE = 154.35),))

TEST_RESU (  OBJET =   _F  (  NOM = 'CH1    .CHME.LIGREL.LIEL' ,
                              VALE_CALC_I = 1278484 ,))


TEST_RESU (  MAILLAGE =(
                 _F  (  MAILLAGE= ma , CARA= 'NB_MAILLE'
                        VALE_CALC_I = 1000 ,),
                 _F  (  MAILLAGE= ma , CARA= 'NB_GROUP_NO'
                        VALE_CALC_I = 12 ,),
                 _F  (  MAILLAGE= ma , CARA= 'EXI_GROUP_MA'
                        NOM_GROUP_MA='GAUCHE',
                        VALE_CALC_I = 25 ,),
              ))
```


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