Operator DEFI_MAILLAGE

1 Goal

To define a grid using macronutrients.

This order makes it possible to define a new grid starting from static or dynamic macronutrients. This new grid (containing only the geometrical supports of the macronutrients) can then be “assembled” with another grid (container for example of the “classical” meshs thanks to the order ASSE_MAILLAGE [U4.23.03] and the option specific to the under-structuring.

Product a structure of data of the type grid.

2 Syntax

```
my (grid) = DEFI_MAILLAGE (
    ♦  DEFI_SUPER_MAILLE = (_F(
       ♦  MACR_ELEM =      l_se   ,                    [l_macr_elem_*]
       ◊  SUPER_MAILLE =          l_mail,              [l_maille]
       ◊  |  ◊  TRAN      =  /   (tx, ty), or (tx, ty, tz), [l_R]
                             /   (0. , 0.) or (0. , 0. , 0.),  [DEFECT]
          |  ◊  ANGL_NAUT =  /   (A),  or (has, B, c),      [l_R]
                             /   (0.), or (0. , 0. , 0.),   [DEFECT]
             ◊  CENTER    =  /   (px, py) or (px, py, pz),  [l_R]
                             /   (0. , 0.), or (0. , 0. , 0.), [DEFECT]
                        ),),
    ◊  |  RECO_GLOBAL  = (_F  (
          ♦  /  ALL       =  'YES',
             /  SUPER_MAILLE =  l_maille  ,            [l_maille]
          ◊  |  CRITERION =  /  'ABSOLUTE'   ,
                             /  'RELATIVE'   ,            [DEFECT]
             |  PRECISION =  /  prec      ,            [R]
                             /  1.D-3     ,            [DEFECT]
                          ), ),
       |  RECO_SUPER_MAILLE = (_F(
          ♦  SUPER_MAILLE    =  l_mail ,               [l_maille]
          ♦  GROUP_NO        =  l_gno  ,               [l_group_no]
          ◊  /  OPTION =  'GEOMETRICAL',               [DEFECT]
                ◊  |  CRITERION =  /  'ABSOLUTE'   ,
                                   /  'RELATIVE'   ,      [DEFECT]
                   |  PRECISION =  /  prec   ,          [R]
                                   /  1.D-3  ,          [DEFECT]
             /  OPTION =  'NOEUD_A_NOEUD',
             /  OPTION =  'OPPOSITE',
                          ),),

    ◊  DEFI_GROUP_NO = _F  (
          /  ♦  /  ALL    =      'YES'  ,
                /  SUPER_MAILLE =      e-mail ,       [mesh]
             ◊  PREFIX   =      pref   ,               [KN]
             ♦  INDEX    =      (DM, Fm, dn, fn),         [l_I]
          /  ♦  GROUP_NO_FIN =  gno_fin,               [group_no]
             ♦  SUPER_MAILLE =  e-mail ,               [mesh]
             ♦  GROUP_NO_INIT=  gno_ini,               [group_no]
                   ),),
                                  )
```

3 General information

In the documentation of this order, one will speak about:

- macronutrient: object of the type `macr_elem_stat` or `macr_elem_dyna`,
- super-mesh: geometrical entity supporting one macro_élément,
- grid **initial** when one indicates the grid which was used to generate a macro - element,
- grid **final** to indicate the grid produced by this order.

By extension these adjectives **initial/final** will apply to the entities attached to the grids: node, mesh, group of nodes.

Practically, to build the final grid:

- one defines super-meshs while positioning in the space (2D or 3D) of the macro - existing elements (the same macronutrient can generate several super-meshs),
- one resticks the super-meshs between them,
- one re-elects, if it is wanted, certain nodes,
- one creates, if it is wanted, certain groups of nodes.

Note:

> *One can note that the grid created by this order is made only of super - meshs. One thus cannot (for example), to draw it with the post - usual processors. Possibilities of curing it will be able to exist with the order* `DEFI_SQUELETTE` *[U4.24.01].*
>
> *To mix finite elements "classical" and substructures, it is necessary to use the operator of "concatenation" of grids [U4.23.03]:* `mag = ASSE_MAILLAGE (MAILLAGE= (m1, m2))`

A grid resulting from the operator `DEFI_MAILLAGE` contains:

- super-meshs,
- nodes,
- groups of nodes.

The super-meshs are defined by translation/rotation of macronutrients.

As a "classical" mesh, a super-mesh is entirely defined by the list of its nodes. The coordinates of the nodes of the meshs are those of the external nodes of the macro - elements transformed by the geometrical transformation: translation, rotation…

If one does not carry out a sticking together (cf. `RECO_GLOBAL` / `RECO_SUPER_MAILLE`), the grid has as many nodes as the sum of the nodes of the super-meshs.

C1 convention:

When one "resticks" the super-meshs, one eliminates certain nodes. By convention, during an elimination of coinciding nodes, one preserves the node (and thus its coordinates) which comes from the first mesh of the list `l_mail` (cf. `RECO_GLOBAL` / `RECO_SUPER_MAILLE`).

As in any grid *Aster*, the nodes are **named**. By defaults, the names of the nodes are given by the program in the form: `Nijk` where `ijk` is a number understood enters `1` and `999999.9`.

The keyword `DEFI_GROUP_NO` allow the user to re-elect certain nodes and to define groups of nodes.

4 Operands

4.1 Keyword DEFI_SUPER_MAILLE

♦ DEFI_SUPER_MAILLE =

This keyword factor makes it possible to define the super-meshs of the grid using the macronutrients.

4.1.1 Operand MACR_ELEM

♦ MACR_ELEM = l_se

l_se is the list of the names of the macronutrients which will generate the meshs.

4.1.2 Operand SUPER_MAILLE

◊ SUPER_MAILLE = l_mail

l_mail is the list of the names which one wants to give to the meshs. This argument is optional. In his absence, one will give to the meshs the names macronutrients (this is obviously impossible if one wants to use several times the same macronutrient).

4.1.3 Geometrical operands of transformations

◊ | ◊ TRAN =

This keyword defines the translation to be applied to the macronutrient:

- if one is in **2D**, 2 realities are expected: (tx, ty),
- if one is in **3D**, 3 realities are expected: (tx, ty, tz).

| ◊ ANGL_NAUT =

◊ CENTER =

These keywords define rotation to be applied to the macronutrient.

If one is in **2D**, 3 realities are expected:

- has is the angle (in degrees) of rotation in the plan for ANGL_NAUT,
- px and py are the coordinates of the centre of rotation for CENTER.

If one is in **3D**, 6 realities are expected:

- has, B, C are the nautical angles (α, β, γ) rotation (in degrees). (Cf the operator AFFE_CARA_ELEM [U4.42.01]) for ANGL_NAUT,
- px, py and pz are the coordinates of the centre of rotation for CENTER.

Notice important:

*It is known that the order of the keywords is not significant for Aster. The operation of translation/rotation is **conventionally** made in the order rotation **then** translation. These two operations do not commutate in general.*

4.2 Keyword RECO_GLOBAL

```
◊  |  RECO_GLOBAL =
   ♦  /  ALL          = 'YES',
      /  SUPER_MAILLE = l_maille,
   ◊  |  CRITERION =  /  'ABSOLUTE'   ,
                      /  'RELATIVE'   ,            [DEFECT]
      |  PRECISION =  /  prec,
                      /  1.D-3,                  [DEFECT]
```

This keyword makes it possible to restick **automatically** a set of super-meshs (indicated by the keyword SUPER_MAILLE or the keyword ALL) with a geometrical criterion of proximity: 2 nodes of 2 different super-meshs m1 and m2 will be confused if the distance which separates them is:

```
<  prec                    (CRITERION = 'ABSOLUTE'),
<  prec*min (D (m1), D (m2))  (CRITERION = 'RELATIVE').
```

where D (semi) note the smallest distance between 2 nodes of the super-mesh semi.

Note:

Two nodes of the same mesh will never be restuck.

If a mesh contains one node, it is necessary to use it CRITERION = 'ABSOLUTE'.

4.3 Keyword RECO_SUPER_MAILLE

◊ RECO_SUPER_MAILLE =

This keyword factor makes it possible to restick "**with the hand**" certain super-meshs indicated by the user. The super-meshs which one can restick are those which were defined by the keyword DEFI_SUPER_MAILLE. One then resticks the super-meshs via groups of nodes. To say what one wants to restick it is thus necessary to give of the couples (mesh, group of nodes (initial grid)).

Note:

When one gives a couple (mesh , group of nodes), one indicates the list of the nodes of the group of nodes which are external for the macronutrient which defines the super-mesh. It is in fact the intersection of the group of nodes and the edge of under - structure. This list is ordered as the initial group of nodes.

In theory, when one resticks 2 meshs via 2 groups of nodes, the whole of the indicated nodes must restick (cf the convention chosen by the keyword OPTION). A message of alarm will be transmitted if it is not the case.

4.3.1 Operands SUPER_MAILLE / GROUP_NO

◊ SUPER_MAILLE =

One gives the list of the meshs here to be restuck. In general, one resticks meshs 2 by 2.

For the "corners", it can be pleasant to restick all the convergent meshs in only once (for example the 4 super-cubic ones which divide the same edge).

◊ GROUP_NO =

One gives here the list of the groups of nodes to be restuck. This list is of the same length than the list of the meshs.

4.3.2 Operand OPTION

◊ OPTION =

This word makes it possible to choose the convention of sticking together of the lists of nodes defined by the groups of nodes.

- 'GEOMETRICAL' :

  The program will confuse the nodes by considerations of geometrical proximity. (Cf keyword: RECO_GLOBAL)

- 'OPPOSITE' 'NOEUD_A_NOEUD'/ :

  Soit : $G1 = \{A1, B1, C1\}$
  $G2 = \{A2, B2, C2\}$
  $G3 = \{A3, B3, C3\}$

  If OPTION = 'NOEUD_A_NOEUD' , GROUP_NO = (G1, G2, G3)

  on va recoller : *A1* avec *A2* avec *A3*
  B1 avec *B2* avec *B3*
  C1 avec *C2* avec *C3*

  If OPTION = 'OPPOSITE' , GROUP_NO = (G1, G2, G3)

  on va recoller : *C1* avec *A2* avec *A3*
  B1 avec *B2* avec *B3*
  A1 avec *C2* avec *C3*

  Caution:

  For option 'OPPOSITE' , only the first group of nodes of the list of GROUP_NO "is turned over".

4.4 Keyword DEFI_GROUP_NO

◊ DEFI_GROUP_NO =

This keyword factor makes it possible to define groups of nodes starting from groups existing in the initial grids of the macronutrients.

Note:

An initial group of nodes can contain nodes which do not belong to the edges of the macronutrients. These internal nodes thus do not exist in the final grid. By convenience, one takes convention nevertheless to create the group reduced to his intersection with the edge of the macronutrient.

4.4.1 Operands ALL / SUPER_MAILLE / PREFIX / INDEX

```
| ♦ / ALL = 'YES',
  / SUPER_MAILLE = e-mail,
  ◊ PREFIX = pref,
  ♦ INDEX = (DM, Fm, dn, fn),
```

These keywords make it possible to create all the groups of nodes corresponding to the groups of the initial grid associated with the mesh e-mail or with all the meshs if:

TOUT= 'YES'.

The convention of renaming is the following one (in pseudonym FORTRAN):

gno_fin (k8) = pref//no_mail (DM: Fm) //gno_ini (dn: fn)

What wants to say that the name of a group of nodes will be formed while concaténant:

- the prefix possibly given by the user,
- a under-chain of characters extracted the name of the mesh,
- a under-chain of characters extracted the name of group_no initial grid.

It is necessary thus that:

ltot= length (prefix) + (FM-dm+1) + (fn-dn+1) □ 8

A frequent case is the following: the grids which gave rise to the macro - elements come from a preprocessor which generates names of the form GRNOijkl. If the user gives to his super-meshs of the names with 2 characters: SA, SB,...., the sequence:

```
DEFI_GROUP_NO=_F (TOUT= 'YES', PREFIXE=' GN', INDEX=
(1,2,5,8))
```

Will generate groups of nodes of names:

GNSA0001, GNSA0002,… , GNSB0001.

4.4.2 Operands GROUP_NO_FIN / SUPER_MAILLE / GROUP_NO_INIT

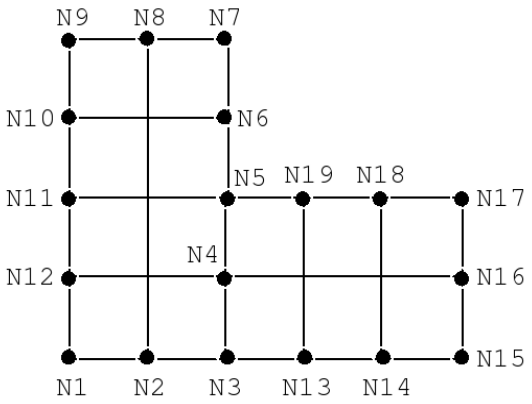
```
| ♦ GROUP_NO_FIN = gno_fin,
  ♦ SUPER_MAILLE = e-mail,
  ♦ GROUP_NO_INIT = gno_ini,
```

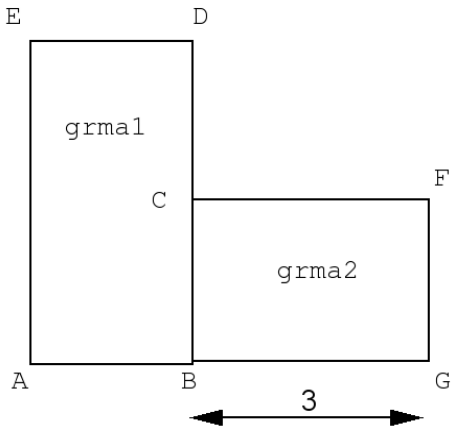
These keywords make it possible to create groups of nodes **one by one** :

- gno_fin is the name which one wants to give to GROUP_NO,
- e-mail and gno_ini identify it GROUP_NO initial:
  - e-mail is the name of the super-mesh which carries it GROUP_NO,
  - gno_ini is the name of GROUP_NO initial grid.

5 Example

That is to say grid m1:



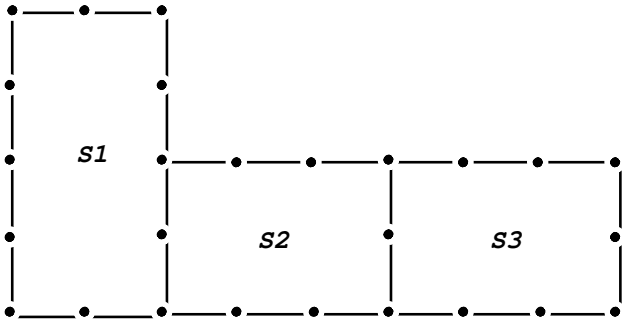



```
GROUP_NO:                          GROUP_MA:

AB = (N1 N2 N3)                       grma1
BC = (N3 N4 N5)                       grma2
CD = .....
......
```

On this grid m1 2 are defined macr_elem_stat.

```
mo1 =  AFFE_MODELE      (   AFFE = _F (GROUP_MA = grma1)…)
mo2 =  AFFE_MODELE      (   AFFE = _F (GROUP_MA = grma2)…)

S1 =   MACR_ELEM_STAT   (   DEFINITION = _F  (MODEL = mo1…)
                                OUTSIDE  = _F(GROUP_NO = (AB, BC, CD, OF, EA))
                            …)
S2 =   MACR_ELEM_STAT   (   DEFINITION = _F  (MODEL = mo2…)
                                OUTSIDE  = _F(GROUP_NO = (BC, BG, FG, CF))
                            …)
```

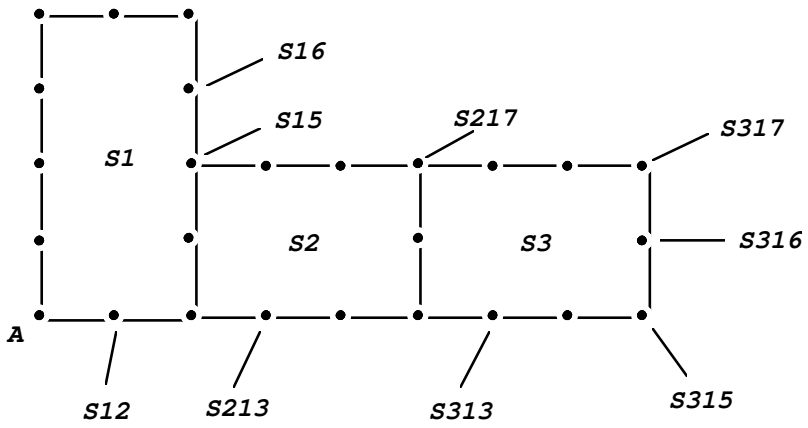
One can then define the grid m2 :



```
m2 =   DEFI_MAILLAGE(

    DEFI_SUPER_MAILLE= (
        _F (MACR_ELEM = S1) ,
        _F (MACR_ELEM = S2  ,   SUPER_MAILLE = S2,  ) ,
        _F (MACR_ELEM = S2  ,   SUPER_MAILLE = S3,  TRAN = 3.),),

    RECO_SUPER_MAILLE= (
        _F (SUPER_MAILLE= (S1, S2), GROUP_NO= (BC, BC), OPTION='
    NOEUD_A_NOEUD'),
        _F (SUPER_MAILLE= (S2, S3), GROUP_NO= (FG, BC), OPTION=' INVERSE'),),


    DEFI_GROUP_NO =
       _F (GROUP_NO_FIN = FG, SUPER_MAILLE = S3,   GROUP_NO_INIT = FG),)
```

The grid obtained contains:

- 3 super-meshs : S1, S2, S3
- 26 nodes : WITH, S12,…, S317
- 1 GROUP_NO : FG = (S315, S316, S317)



Note:

The sticking together of the super-meshs could have been made more simply by:
RECO_GLOBAL = _F (ALL = 'YES').