Structures of data sd_carte , sd_cham_no , sd_cham_elem and sd_resuelem

Summary:

Contents

1 General information

The structures of data representing the fields are:

- sd_cham_no : field on the nodes of a grid
- sd_carte : field on the meshs of a grid
- sd_cham_elem : field on the elements of a ligrel
- sd_resuelem : field of matrices (or vectors) elementary on the elements of a ligrel

We call “size” a “vector” of components (CMP) of the field.
For example, for a field of displacement: (‘DX‘,’DY‘,’DZ’).
A discretized field is a set of sizes located on nodes, points of Gauss or meshs.
All the sizes of a field do not have the same components inevitably: for example, on certain parts of the grid, the nodes can have 6 CMPS of displacement (elements of beam) whereas on other parts, the nodes have only 3 CMPS (voluminal elements).

The components of a size are a subset of the CMPS declared in the catalogue of the sizes [D4.04.01]. To describe a size, in addition to its digital values, it is necessary to know of which CMPS it act; for that, one uses the concept of “descripteur_grandor” who describes the presence (or not) of the whole of the CMPS of the catalogue. This concept is described below.

- sd_carte are fields discretized on the meshs of a grid (or late meshs of a ligrel). There exists 1 size by mesh,
- sd_cham_no are fields discretized on the nodes of a grid (or late nodes of a ligrel). There exists 1 size by node,
- sd_cham_elem are fields discretized on the elements of a ligrel. It can exist several sizes by element (for example a size by point of Gauss or node). The points of dicretisation (nodes or not of Gauss) can have under-points; if it is the case, all the points have the same number of under-points,
- sd_resuelem are fields discretized on the elements of a ligrel. The sizes associated with such fields are the sizes known as “elementary”: elementary matrices or elementary vectors. The whole of the values of one resuelem can be bulky, this is why the object containing these values (.RESL) a structure of dispersed collection has.

Notice important:

The structures of data described here are not easy use. They are SD normally used in operations of low levels: elementary calculations, assemblies, resolutions… When one wants to read or write in such SD, it is often preferable to transform them beforehand into more convenient SD to use (“simple” fields). Routines of ad hoc transformation: CNOCNS, CNSCNO, CELCES, CARCES… are described in [D4.06.06].

2 Tree structures

```
sd_carte  (K19)  :: =record
  ♦ '.NOMA'      :   OJB   S   E  K8
  ◊ '.NOLI'      :   OJB   S   V  K24
  ♦ '.DESC'      :   OJB   S   V  I
  ◊ '.LIMA'      :   OJB   XC  V  I
  ♦ '.VALE'      :   OJB   S   V  R/C/K8/K24/…

sd_cham_no  (K19)  :: =record
  ♦ '.DESC'      :   OJB   S   V  I
  ♦ '.REFE'      :   OJB   S   V  K24
  ♦ '.VALE'      :   OJB   S   V  R/C/K8

sd_cham_elem  (K19)  :: =record
  ♦ '.CELK'      :   OJB   S   V  K24
  ♦ '.CELD'      :   OJB   S   V  I
  ♦ '.CELV'      :   OJB   S   V  R/C/K8/…

sd_resuelem  (K19)  :: =record
  ♦ '.NOLI'      :   OJB   S   V  K24
  ♦ '.DESC'      :   OJB   S   V  I
  ♦ '.RESL'      :   OJB   XD  V  R/C
```

3 Contents of objects JEVEUX

3.1 DESCRIPTEUR_GRANDEUR

It is a vector of entireties. It describes the CMPS present indeed in a size.

All the possible CMPS of a size are described in the catalogue of the SIZES. They are ordered there. To describe the CMPS indeed present in a size one decides to keep a vector of Boolean which answers the following question: is the ième CMP (in the order of the catalogue of the sizes) it presents in the size which one wants to describe? To save place memory (and disc), one decides “to code” this vector of Boolean on a vector of entireties: on each entirety (called entier_codé), one codes 30 Boolean.

Example:

If size ‘DEPL_R’ was described in the catalogue by:

DX	DY	DZ	DRX	DRY MARTINI	DRZ	LAGR

On an element of the type beam the descripteur_grandor is worth 126. Indeed:

	DX	DY	DZ	DRX	DRY MARTINI	DRZ	LAGR
	1	1	1	1	1	1	0
126 =	2^1	+ 2^2	+ 2^3	+ 2^4	+ 2^5	+ 2^6	

On an element of the type voluminal the descripteur_grandor is worth 14. Indeed:

	DX	DY	DZ	DRX	DRY MARTINI	DRZ	LAGR
	1	1	1	0	0	0	0
14 =	2^1	+ 2^2	+ 2^3				

On an additional node creates for the kinematic introduction of condition by dualisation, the descripteur_grandor is worth 128. Indeed:

	DX	DY	DZ	DERX	DRY MARTINI	DRZ	LAGR
	0	0	0	0	0	0	1
128 =							2^7

A descripteur_grandor is a vector of entier_codés: V of dimension n_ec where n_ec is the number of entier_codés necessary to the description of the size described in the catalogue.

n_ec	many CMPS in the catalogue
1	1 to 30
2	31 to 60
3	…

I[ème] entier_codé about the presence (or not) CMPS numbered inform of 30* (i-1) +1 → 30*i.

4 SD map

4.1 General information

A map is a field discretized by mesh. Each mesh can be “affected” of a size (with more). The cards are in general SD create starting from the data of the user. Its structure is made to store (with less possible volume) information concerning the assignment of the sizes on “pieces” of the grid.

Note:

The selected structure is economic spaces some but she does not answer the question quickly: which size is affected on the M1 mesh? To answer this question, it is necessary “to extend” the map (that to create bulkier temporary objects); it is the object of the routine ETENCA called by CALCULATION. A map is thus an ordered list of couples (size, zone_affectée). The order of the couples is important because it is used to take into account the principle of overload of the assignments: the last assignment takes precedence over the preceding ones.

One zone_affectée can be:

- the whole of the meshs of the grid (ALL: 'YES'),
- the whole of the late meshs of one ligrel,
- one GROUP_MA grid,
- a list of meshs of grid,
- a list of late meshs of one ligrel.

4.2 Object .NOMA

Name of the grid associated with the map.

4.3 Object .DESC

'.DESC' S V I DIM = 3 + (2+n_ec) *n_gd_max

The field ' DOCU 'L' object. DESC contains: ' CART '

DESC (1)	Gd (number of the size associated with the map)
DESC (2)	n_gd_max (raising amongst zone_affectée)
DESC (3)	n_gd_edit (real number of zone_affectée) n_gd_edit can be = 0
DESC (3+1)	code_1er_zone ("code" of the zone_affectée first)
DESC (3+2)	number of 1st zone_affectée
.....	
DESC (3+2*n_gd_max-1)	code_der_ent (code of the zone_affectée last)
DESC (3+2*n_gd_max)	number of the zone_affectée last

Caution:

it is necessary n_gd_max=1 for a constant map on all the late meshs (for example CHTIME in me2mme.f)

"code" of one zone_affectée can be worth:

1	the whole of the meshs of the grid (ALL: 'YES '),
-1	the whole of the late meshs of U N ligrel ,
2	one GROUP_MA grid ,
3	a list of meshs of grid ,
-3	a list of late meshs of one ligrel .

If code=1 (or -1):
the number of zone_affectée corresponding is not used for nothing.

If code=2:
the number of zone_affectée corresponding is the number of group_ma
in the collection mailla.GROUPEMA

If code=3 (or -3):
the number of zone_affectée corresponding is the number of the object of the collection .LIMA
who contains the numbers of the meshs composing zone_affectée.

Comes then in the object.DESC a continuation of descripteur_grandor describing the various affected sizes.

That is to say n_ec the number of entier_codé necessary to describe the CMPS of the size Gd :

DESC (3+2*n_gd_max+1)	beginning of the first descripteur_grandor
….	….
DESC (3+2*n_gd_max + (n_gd_max-1) *n_ec +1)	beginning of the last descripteur_grandor

Note:

For a constant field (1 only size assigned to all the meshs of the grid). One has then:
DESC (2) = 1
DESC (3) = 1
DESC (4) = 1
DESC (5) = it does not matter
DESC (6) = beginning of the descripteur_grandor of zone_affectée (ALL: 'YES'
In this case.LIMA and.NOLI are not allocated (saving of space).

4.4 Object .NOLI

This object is present only if the map relates to late meshs.

It is a vector of K24 of dimension nb_gd_max. Opposite izone one finds, if this zone_affectée is a list of late meshs, the name of ligrel or these meshs are defined.

```
izone ---> nom_ligrel
```

4.5 Object .LIMA

It is a numbered contiguous family of vectors of entireties.

```
LIMA (izone): V (I)
```

V contains the numbers of the meshs constituting zone_affectée.

The numbers of meshs of the list are numbers relative to ligrel referred in .NOLI (izone).

if a number of mesh is > 0, it is a mesh of the grid associated with the map.
if a number of mesh is < 0, it is a mesh of additional ligrel.

4.6 Object .VALE

It is a vector of scalars dimensioned with nb_gd_max * nb_cmp_max, if nb_cmp_max is the number of CMPS in the catalogue for the size associated with the map.

Size associated with zone_affectée izone start in.VALE with the index:

```
izone -->  .VALE ((izone-1) *nb_cmp_max + 1)
```

Caution:

Only the affected CMPS are stored (consecutively and in the order of the catalogue) in the object.VALE For example, for a map of DEPL_R, if the 1st zone is affected by: (DX=2. and DZ=4.)
VALE (1) = 2.
VALE (2) = 4.

5 SD cham_no

5.1 Object .DESC

The field 'DOCU'L' object.DESC contains: 'CHNO'

DESC (1)	Gd (size associated with cham_no)
DESC (2)	num
DESC (3),…, DESC (3 + n_ec - 1)	descripteur_grandor of the size if num is < 0

If num is negative num = "-" nb_cmp

If num is < 0, its absolute value is the number of CMP of the size for ALL the nodes of the grid (e.g. the field of geometry). In this case the field relates to only the nodes of the grid (not of late nodes) and it is supposed that all the nodes have the same representation of the size.

The descripteur_grandor is then stored of DESC (3) with DESC (3 + n_ec - 1). If num is positive, it exists a structure of the type then prof_chno referred in the object.REFE.

5.2 Object .REFE

REFE (1)	name DU GRID.
REFE (2)	name of one prof_chno [D4.06.07] (if DESC (2) >0) The SD prof_chno described the CMPS carried by the nodes of cham_no. It is used to point in the object.VALE who contains the values.
REFE (3)	unutilised
REFE (4)	unutilised

5.3 Object .VALE

This object contains the "values" of the field to the nodes on the nodes of the grid or the late nodes of ligrel used in prof_chno.

6 In the case general

The description of the object.VALE if it cham_no is not with "constant representation" is made in [D4.06.07 §3].

7 If it cham_no is with "constant representation"

That is to say:
nb_no : the number of nodes of the grid.
ncmp : the number of CMPS ranges by all the nodes of the grid.

```
LENGTH (.VALE) = nb_no * ncmp
```

VALE (1)	value of the 1st CMP carried by the 1st node
VALE (2)	value of the 2nd CMP carried by the 1st node
…	…
VALE (ncmp)	value of the last CMP carried by the 1st node
VALE (ncmp+1)	value of the 1st CMP carried by the 2nd node
…	…

The order of the CMPS is that of the catalogue of the sizes (object '&CATA.GD.NOMGD' [D4.04.01]).

8 SD champ_elem

8.1 Case of cham_elem having under-points (elements of structure)

The number of points of discretization (nodes, points of Gauss,…) of one cham_elem on a mesh is a priori given by the number of points defined in the catalogue of type_elem associated with the mesh. For the elements of the type "structure", one wants to be able to store more sizes than of points defined in the catalogue.

During a non-linear calculation on a hull (for example), the integration chosen for the non-linear behavior requires to store the state of stresses in several points in the thickness: it is necessary to discretize the thickness of the hull. For that, it will be said that each point of Gauss positioned on the surface of the element (their number is fixed in the catalogue of type_elem), is composed of N under-points representing the discretization of the normal to the element in this point.

In the same way, a non-linear element of pipe, will be able to discretize its section (circular ring) by cutting out it in sectors and layers.

For a given element, all the points of discretization have obligatorily the same number of under-points.

Caution:

Before creating one cham_elem with under-points, it is necessary to say for all the elements of ligrel the desired number of under-points. For that, one is used cham_elem_s size DCEL_I (argument DCELZ routine alchml.f). When one calls the routine of elementary calculations (calcul.f), the passage of this argument is underground: cham_elem_s must have the same name as it cham_elem (OUT) that it is used to dimension.

8.2 Case of cham_elem not having under-points

By means of computer, all them cham_elem under-points have. One cham_elem who does not need this concept is in fact one cham_elem for which each point of discretization has one under-point; one then confuses the point and his single under-point.

8.3 Case of cham_elem size VARI_R

Size VARI_R is the “reserved” size which is used to represent a size of which the component count (CMP) is unspecified on the level of the catalogues of type_elem.

One makes use for example of this size to represent the internal variables of the laws of behavior, because each law can have a number different from such variables.

In the catalogue of the sizes, this size has only oneule CMP : VARI.

At the time of the creation of one cham_elem_VARI_R, one must say for each element, how much components will have the size VARI_R. These components will be called then: ‘V1’, ‘V2’,…, ‘Vn’. For that, one uses the same mechanism as to declare the number of the under-points (see above).

8.4 Object .CELK

CELK (1)	name of ligrel associated with cham_elem.
CELK (2)	name of the option of calculation associated with cham_elem.
CELK (3)	/ ‘ELNO’: CHAM_ELEM with the nodes / ‘ELGA’: CHAM_ELEM at the points of Gauss / ‘ELEM’: CHAM_ELEM constant by element
CELK (4)	nume_couche : number of the layer (tallied on the left) for one CHAM_ELEM calculated on a layer of element of hull.
CELK (5)	nive_couche : position in the layer for one CHAM_ELEM calculated on a layer of element of hull: /‘INF’/‘MOY’/‘SUP’
CELK (6)	Name of the parameter of the option associated with cham_elem
CELK (7)	/‘MPI_COMPLET‘/’MPI_INCOMPLET’

CELK (7):

- ‘MPI_INCOMPLET‘: calculation was distributed (MPI) and L’ object.CELV is not complete on all the processors. Each processor calculated only one subset of the elements.
- ‘MPI_COMPLET‘: if not. C’ is to be said that calculation was not distributed, or, it cham_elem (MPI_INCOMPLET with its creation) “was supplemented” by calling the utility sdmpic.f

8.5 Object .CELD

.CELD : vector of entireties. The field 'DOCU'L' object.CELD contains: 'CHML'

This object is the descriptor of the object containing the values of cham_elem (.CELV).

CELD (1)	Gd	size associated with cham_elem .
CELD (2)	nb_gr	number of grel ligrel associated.
CELD (3)	mxsp	maximum amongst under-points for the elements of ligrel
CELD (4)	mxcmp	maximum amongst CMP (size VARI_R) for the elements of ligrel. 0 if size different from VARI_R
CELD (4+1)	debu_grel_1	address (- 1) in.CELD beginning of information concerning 1st GREL
…		
CELD (4+nb_gr)	debu_grel_n	address (- 1) in.CELD beginning of information concerning the last GREL

then one stores end to end the description of the field for each GREL ligrel

CELD (debu_grel +1)	nel	number of element of GREL
CELD (debu_grel +2)	modelo	mode_local associated with the local field (or 0 if non-existent field on GREL)
CELD (debu_grel +3)	lgcata	length of the local field within sight of the catalogue. i.e. without taking account of the under-points and the multiple components of VARI_R. (= 0 if modelo = 0)
CELD (debu_grel +4)	lggrel	length total of the segment containing all the values of the field on GREL
then:C iel = 1, nel		(if modelo > 0)
CELD (debu_grel+4+4* (iel-1) +1)	nbsp	number of sous_points for the element iel
CELD (debu_grel+4+4* (iel-1) +2)	ncdyn	many CMP (VARI_R) for the element iel
CELD (debu_grel+4+4* (iel-1) +3)	lgchel	many values of the local field for the element iel. lgchel= lgcata * nbsp * ncdyn
CELD (debu_grel+4+4* (iel-1) +4)	adiel	address in the object.CELV 1st value of the element iel

8.6 Object .CELV

It is a vector containing end to end the values of the local fields of the various elements.

The description of the segment concerning an element is given by mode_local defined for type_elem. This description is possibly supplemented by the data amongst under-points and amongst CMPS (VARI_R).

For a field of size (different from VARI_R) not having under-points, all the elements of same grel having the same one type_elem, their local fields have all the same length and the same organization.

One moves in the object.CELV thanks to the object.CELD.

One can describe the organization of the object.CELV by these definitions:

CELV (ligrel)	continuation of CELV (GREL) put end to end
CELV (GREL)	continuation of CELV (element) put end to end
CELV (element)	continuation of CELV (not) put end to end
CELV (not)	continuation of CELV (under-point) put end to end
CELV (under-point)	continuation of CMP (scalar) settings end to end

8.7 Some “formulas” frequently used in the programming

8.7.1 LIGREL

number of the size associated with CHAM_ELEM :

```
NUMGD=ZI (JCELD-1+1)
```

number of GREL LIGREL associated with CHAM_ELEM :

```
NGREL=ZI (JCELD-1+2)
```

maximum number. under-points of the elements of one CHAM_ELEM: (perhaps = 0)

```
MXSP=ZI (JCELD-1+3)
```

maximum number. of CMPS (VARI_R) elements of one CHAM_ELEM:

```
(/=0 <=> VARI_R)
MXCDY=ZI (JCELD-1+4)
```

8.7.2 GREL : IGR

many elements of one GREL (IGR):

```
NEL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +1)
```

mode_local of one GREL (IGR):

```
IMOLO=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +2)
```

cumulated length of the elements of one GREL (IGR):

```
LGGREL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4)
```

address (in.CELV) beginning of GREL IGR :

```
DEBUGR=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +8)
then: ZR (JCELV -1 +DEBUGR) =…
```

length (CATALOGUE) of an element of one GREL (IGR):

```
LGCATA=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +3)
```

8.7.3 Element IEL GREL IGR

address (in.CELV) beginning of element IEL GREL IGR :

```
ADIEL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +4)
then: ZR (JCELV -1 +ADIEL) =…
```

length of element IEL GREL IGR :

```
LGIEL=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +3)
```

many under-points of element IEL GREL IGR :

```
NBSPT=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +1)
there are no under-points: NBSPT=0 (or 1; notice JD)
```

number of CMPS (VARI_R) of element IEL GREL IGR :

```
NCDYN=ZI (JCELD-1+ZI (JCELD-1+4+IGR) +4 +4* (IEL-1) +2)
0 - > the size is not VARI_R
```

9 SD resuelem

9.1 Object .NOLI

NOLI (1)	name of the ligrel associated with resuelem .
NOLI (2)	name of the option associated with resuelem .
NOLI (3)	/'MPI_COMPLET'/'MPI_INCOMPLET'
NOLI (4)	''(unutilised)

NOLI (3) :

- ' MPI_INCOMPLET' : calculation was distributed (MPI) and the object. RESL is not complete on all the processors. Each processor calculated only one subset of the elements.
- ' MPI_COMPLET ': if not. C' is to be said that calculation was not distributed, or, it resuelem (MPI_INCOMPLET with its creation) "was supplemented" by calling the utility sdmpic.f

9.2 Objet .DESC

The field 'DOCU'L' object.DESC contains: 'RESL'

DESC (1)	Gd (size associated with resuelem)
DESC (2)	nb_gr (number of GREL of.NOLI (1))
DESC (2+1)	mode_1er_gr (mode_local local fields of the first GREL)
…	
DESC (2+nb_gr)	mode_der_gr (mode_local of the last GREL)

9.3 ObjeT .RESL

It is a dispersed collection of vectors of R (or C) . The access to this collection is done by the number of GREL: .RESL (IGREL) → V

If ncmpel is the number of scalars representing the local field for an element of GREL ,

V (1,…, ncmpel)	values of the field on the 1st element of GREL
V (ncmpel+1,…, 2*ncmpel)	values of the field on the 2nd element of GREL
…	…

Convention of storage of the symmetrical elementary matrices:

1		
2	3	
4	5	6

Convention of storage of the nonsymmetrical elementary matrices:

1	2	3
4	5	6
7	8	9

Caution:

Because of the "distribution" of the elements (AFFE_MODELE/ DISTRIBUTION / METHOD = 'GROUP_ELEM ') objects of the collection. RESL can exist within the meaning of JEEXIN without one being able to do one JEVEUO (…, 'L',…) above because they do not have an image disc. To test if these objects really exist, the routine should be used JAEXIN .

10 Examples

10.1 SD map

```
MAP = CREA_CHAMP (TYPE_CHAMP: 'CART_META_R', OPERATION: 'AFFE',
                  GRID: NETTED
                  AFFE: (ALL    : 'YES'
                            NOM_CMP: ('ZF' 'ZP' 'ZB' 'ZM' 'P')
                            VALE    : (0.0   0.0  0.0  0.0  0.0) )
                  AFFE: (GROUP_MA    : GM2
                            NOM_CMP: ('ZF' 'ZP')
                            VALE    : (0.2  0.3  ) )
                  AFFE: (MESH    : T2
                            NOM_CMP: ('ZP' 'ZM' 'P')
                            VALE    : (0.4  0.5  0.6 ) )
                 ) ;
```

```
 SEGMENT IMPRESSION OF VALUES >CARTE              .DESC        <
      1 -           64            3            3            3            1
      6 -            3            2            3            3          254
     11 -          254          254
```

```
--------------------------------------------------------------------------------
 IMPRESSION OF THE COLLECTION: MAP              .LIMA
 OBJECT IMPRESSION OF COLLECTION CONTIGUE>CARTE              .LIMA<  OC:      1
      1 -            1            3
 OBJECT IMPRESSION OF COLLECTION CONTIGUE>CARTE              .LIMA<  OC:      2
      1 -            2
 OBJECT IMPRESSION OF COLLECTION CONTIGUE>CARTE              .LIMA<  OC:      3
      1 -            4            5
--------------------------------------------------------------------------------
 SEGMENT IMPRESSION OF VALUES >CARTE              .NOLI        <
      1 - >                        <>                      <
      3 - >                        <
--------------------------------------------------------------------------------
 SEGMENT IMPRESSION OF VALUES >CARTE              .NOMA        <
      1 - >MAILLA  <
--------------------------------------------------------------------------------
 SEGMENT IMPRESSION OF VALUES >CARTE              .VALE        <
      1 -  2.00000E-01  3.00000E-01  0.00000E+00  0.00000E+00  0.00000E+00
      6 -  0.00000E+00  0.00000E+00  2.00000E-01  4.00000E-01  0.00000E+00
     11 -  5.00000E-01  0.00000E+00  0.00000E+00  6.00000E-01  0.00000E+00
     16 -  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
     21 -  0.00000E+00
```

Note:

The contents of the objects printed above can surprise: it does not correspond to what is known as higher. Indeed this map "was finished" by a call to the routine `tecart.f`.
The purpose of this optional action is to allow a "fine" overload of the values affected in the order `CREA_CHAMP`.

10.2 SD cham_no

```
cham_no = CREA_CHAMP (GRID: netted, TYPE_CHAMP: 'NOEU_DEPL_R',
          OPERATION: 'AFFE',
          AFFE: (GROUP_NO: gn1
                nom_cmp: 'DX' VALE_R: 1.0)
          AFFE: (NODE: (N2, n7)
                NOM_CMP: ('DX', 'DZ') vale_r: (2. , 4.))
                );
```

```
 SEGMENT IMPRESSION OF VALUES >CHAM_NO            .DESC        <
      1 -           32            6
--------------------------------------------------------------------------------
 SEGMENT IMPRESSION OF VALUES >CHAM_NO            .REFE        <
      1 - >MAILLA                  <>cham_no                  <
--------------------------------------------------------------------------------
 SEGMENT IMPRESSION OF VALUES >CHAM_NO            .VALE        <
      1 -  2.00000E+00  4.00000E+00  1.00000E+00  1.00000E+00  2.00000E+00
      6 -  4.00000E+00
```

10.3 SD cham_elem

```
FLUXN=CALC_CHAM_ELEM (    MODELE=MOTH,  TEMP=T2,
            CHAM_MATER=CHMAT,   OPTION=' FLUX_ELNO')
```

```
SEGMENT IMPRESSION OF VALUES >FLUXN              .CELD        <
```

```
      1 -           47            2            1            0            6
      6 -           18            2         6520            8           16
     11 -            1            0            8            1            1
     16 -            0            8            9            3         6857
     21 -            6           18            1            0            6
     26 -           17            1            0            6           23
     31 -            1            0            6           29

 SEGMENT IMPRESSION OF VALUES >FLUXN              .CELV        <

      1 - -8.78595D-12 -4.27645D-12 -8.78595D-12 -4.08919D-12  6.96696D-12
      6 - -4.07954D-12  6.96696D-12 -4.77838D-12  4.96957D-12 -4.15161D-12
     11 -  4.96957D-12 -4.26679D-12 -1.33159D-12 -4.54543D-12 -1.33159D-12
     16 - -3.57760D-12  7.27596D-12 -8.41283D-12  7.27596D-12 -8.41283D-12
     21 -  7.27596D-12 -8.41283D-12  0.00000D+00 -8.86757D-12  0.00000D+00
     26 - -8.86757D-12  0.00000D+00 -8.86757D-12  0.00000D+00 -8.86757D-12
     31 -  0.00000D+00 -8.86757D-12  0.00000D+00 -8.86757D-12

 SEGMENT IMPRESSION OF VALUES >FLUXN              .CELK        <

      1 - >MOTH    .MODELE         <>FLUX_ELNO          <
      3 - >ELNO                    <>                        <
      5 - >                        <>PFLUX_R
```

10.4 SD `resuelem`

```
CHTH= AFFE_CHAR_THER (MODEL: MODEL  TEMP_IMPO: (NODE: N8 TEMP: 3.4)
                     SOURCE: (ALL: 'YES'  SOUR: 7.)        );
VECTEL=CALC_VECT_ELEM (LOAD: CHTH OPTION: 'CHAR_THER');
```

`resuelem` is extracted from `VECT_ELEM VECTEL: 'VECTEL .VE001'`

```
 SEGMENT IMPRESSION OF VALUES >VECTEL  .VE001     .DESC        <
      1 -          105            3         5781         5648            0
--------------------------------------------------------------------------------
 SEGMENT IMPRESSION OF VALUES >VECTEL  .VE001     .NOLI        <
      1 - >MODEL   .MODELE         <>CHAR_THER_SOUR_R        <
--------------------------------------------------------------------------------
 IMPRESSION OF THE COLLECTION: VECTEL  .VE001     .RESL
 OBJECT IMPRESSION OF COLLECTION >VECTEL  .VE001     .RESL<  OC:      1
      1 -  3.50000E+00  3.50000E+00  3.50000E+00  4.66667E+00  4.66667E+00
      6 -  4.66667E+00
 OBJECT IMPRESSION OF COLLECTION >VECTEL  .VE001     .RESL<  OC:      2
      1 -  4.08333E+00  4.66667E+00  4.66667E+00  4.08333E+00
```