Rules concerning the structuring of the data

Summary:

We indicate here the rules (and advices) concerning the Structuring of Data (SD):

- to define new types of SD,
- with which objects JEVEUX?

The reading of this document supposes the preliminary reading of documents [D4.01.01] and [D5.01.03].

Contents

1 To define and use the new ones type_SD

- 1 - To think “Structures of Data”: not only for the Structures of Data “users” but also for all the objects of work. As soon as one must handle several objects “at the same time”, to gather them in a kind of Structures of Data which one will document.

- 2 - “To consult the asset”:
  - to know the types of existing SD,
  - to measure the adequacy of the types of existing SD to its need,
  - to put the question (and to pose it with the others): is necessary it to modify an existing SD or to create a new one. In particular, it is necessary to avoid the multiplication of the types of SD “user”,
  - any modification (or introduction) of type of SD must be discussed in EDA.

- 3 - To use the names of the types of Structures of Data in the comments of the routines. Scrupulously to respect for that the orthography of these names [D2.01.02].

  To write for example:

```
    SUBROUTINE LOUSE (CART, NETTED)
C IN CART   : NAME OF A SD MAP
C IN NETTED: NAME OF A SD GRID
```

- 4 - To use new suffixes for very new type of Structures of Data. The list of the suffixes currently used is given in [D4.01.02]. For reasons of legibility, one can also begin his suffixes with a common character string.
  Example: SD LISTE_RELA (K19) :

  suffixes: '.RLCO', '.RLDD', '.RLNO',... (“.RL” – > “linear relation”).

- 5 - A suffix must start with one “.” (legibility).
  One can use the characters: “WITH, B,… Z, 0.1,… 9, _, &.”

- 6 - To respect the “standard” lengths for the names of the Structures of Data:
  K8, **K14** or **K19**.

- 7 - Not to multiply objects JEVEUX to make “prettier”:
  - one can store 1 K8 and 1 K24 in a vector of 2 K24
  - Boolean insulated can be coded in an entirety (0 or 1),
  - …

- 8 - When one “begins” the structuring (for the types of Structures of Data moreover low level), to use the longest names (K19) to be able later on to create new type of Structures of Data the container. (i.e to begin the suffixes “with the line “).

  Example:

```
type_1er      (K19)     record
   '. T1AA'     :         OJB     …
   '. T1BB'     :         OJB     …
type_2nd      (K14)     record
   '. T2AT1'    :         type_1er
   …
```

  The type of SD type_1er can be used as article of the type type_2nd.

- 9 - To think of the references “upstream”: it is sometimes useful to store in a Structure of Data the names of the Structures of Data which gave him rise. This supposes that these Structures of Data are perennial (have an image in the base ‘TOTAL‘).

- 10 - To avoid the redundancies within a type of Structures of Data, because the update becomes more problematic. A maugo example: listr8 (list of realities) where one stores at the same time the complete listing of realities and the list of the intervals of constant step (one of the objects (.VALE) can be constantly recomputed from the others).

- 11 - When one declares types of Structures of Data “hats”,

```
type_1 (K8):: = record
         / $VIDE     :        type_2
         / $VIDE     :        type_3
```

  it should be made sure that there exists a means “of crossing” them “/”; i.e. to know to recognize if the Structure of Data is of type type_2 or type_3.

  For example:

- 12 - Problem of SD “hidden”:
  It happens sometimes that one SD store the name of others SD (for example, SD_RESULTAT store in its object .TACH the name of the fields which compose SD_RESULTAT). When the referred SD are unknown (or hidden) of the user (it is the case of the fields of SD_RESULTAT), it is necessary to find a name for these SD Référencées.

  The following rule will be adopted:

  If one must name one SD hidden that one reference in a SD user named NOMU (K8), one will give to this SD hidden a name starting with NOMU.

  Thanks to the compliance with this rule, the order DETRUIRE/CONCEPT=NOMU will correctly destroy ALL objects JEVEUX created by the order having created NOMU.

  To obtain a name of hidden SD which complies with this rule, one can use the routine GNOMSD.

- 13 – To write the catalogue Python associated with the new structure of data. See D5.01.03

2 Objects JEVEUX basic

- 1 - Not to use the attribute **'DOCU'** OJB.

- 2 - To try not to use 'LONUTI': one will in general seek to allocate "with just" the objects. In this case, **'LONUTI'='LONMAX'**.

To use the attribute advisedly **'LONUTI'** : it is to the user to update it; another significance should not be given him only his: length really used of a vector.

- 3 - For the collections which one knows that they will be never very large, it is preferable to create them contiguous. In the contrary case, they should be created dispersed. It will be said that a collection (or an object) is large if:
  - its volume can be higher than 1Méga word,
  - or if its volume can be higher than 10 times the number of ddls of the model.

- 4 - Not to use the pointers (name or length) divided between several collections. If (for example) 2 collections must be reached by the same names, one can make:
  - to create a pointer of names,
  - to create the collections in "numbered" access
  - to make CAL JENONU before the access to the collections.

- 5 - Not to store in OJB addresses memory of others OJB (because an address memory is by "temporary" definition). One does it for the type of Structures of Data mater_code and for well identified reasons of performance, but that must remain exceptional.