# ORGANISATION DES DONNEES, ET LEUR INITIALISATION, DANS ARPEGE/IFS: CYCLE 45T1.

YESSAD K. (METEO-FRANCE/CNRM/GMAP/ALGO)

October 25, 2017

*Résumé:*
*Cette documentation donne quelques bases sur l'organisation des données dans ARPEGE/IFS, notamment en relation avec OOPS, et l'ordre dans lequel on initialise les différents objets.*

## Contents

# 1 Introduction.

Le code d'ARPEGE/IFS contient plus de 10000 variables. Dans le cadre de OOPS, ces variables ont été regroupées en objets. Cette documentation donne une présentation de ces objets, et quelques règles sur l'ordre dans lequel on les initialise. On donne quelques informations sur comment introduire une nouvelle variable ou un nouveau module.

# 2 Les objets et leur set-up.

Les variables sont regroupées en objets tels que **INIT**, **GEOMETRY**, **MODEL**, **FIELDS**, **MTRAJ**.

- Il n'y a pas d'encapsulation pour l'objet **INIT**. En cas de multi-instanciation (par exemple une application qui tourne plusieurs versions du modèle avec des résolutions différentes), les variables de l'objet **INIT** ont une valeur identique pour toutes les versions.
- Il y a de l'encapsulation pour les autres objets: en cas de multi-instanciation, chaque version a sa propre valeur. Les objets sont représentés par les variables **YRGEOMETRY**, **YRMODEL**, **YRFIELDS**, **YRMTRAJ**, déclarées dans **CNT0**.

Le set-up commence d'abord à initialiser les variables de l'objet **INIT**, en-dessous de **SU0YOMA** et avant l'appel à **SUGEOMETRY**. Une fois qu'on a initialisé ces variables et qu'on entre dans **SUGEOMETRY**, il est formellement interdit de les modifier. Par exemple des variables comme **NCONF**, **LELAM**, ou **LSLAG** sont dans l'objet **INIT**, et trouver au beau milieu du code des choses du type (on imagine que **NCONF**=601):

```
ICONF=NCONF
NCONF=131
on fait des choses
NCONF=ICONF
```

est parfaitement interdit.

Le set-up poursuit par l'appel à **SUGEOMETRY** (sous **SU0YOMA**), qui initialise l'objet **GEOMETRY** (variable **YRGEOMETRY**). Dans **YRGEOMETRY** on va trouver par exemple le nombre de longitudes (**YRGEOMETRY%YRDIM%NDLON**), le nombre de latitudes (**YRGEOMETRY%YRDIM%NDGLG**), le nombre de niveaux verticaux (**YRGEOMETRY%YRDIMV%NFLEVG**). Une fois qu'on sort de **SUGEOMETRY** il est interdit d'apporter des modifications au contenu de **YRGEOMETRY**. Cela veut dire que, dans les routines appelées après **YRGEOMETRY**:

- on passe **YDGEOMETRY** ou un sous-attribut de **YDGEOMETRY** en argument d'appel.
- cet argument d'appel doit être en INTENT(IN).
- on ne passe pas de variable par module.

Aucune des variables de l'objet **GEOMETRY** n'apparaît dans le set-up de l'objet **INIT**.

Après l'appel à **SUGEOMETRY**, le set-up initialise le contenu d'autres objets, comme l'objet **MODEL**. Les variables de ces objets sont toujours passées via des arguments d'appel. Aucune des variables de ces objets n'apparaît dans le set-up de l'objet **INIT** et dans le set-up de l'objet **GEOMETRY**.

Il faut noter que la séparation du set-up en **SU0YOMA** et **SU0YOMB**, qui date des débuts d'ARPEGE, est obsolescente. Il faut maintenant avoir à l'esprit l'ordre suivant:

- set-up de l'objet **INIT**, sous **SU0YOMA**.
- set-up de l'objet **GEOMETRY** (**SUGEOMETRY**), sous **SU0YOMA**.
- set-up des autres objets (qui dépendent de la géométrie): à la fin de **SU0YOMA**, sous **SU0YOMB**, ainsi que dans d'autres routines de set-up appelées après **SU0YOMB**.

# 3 Comment introduire une nouvelle variable dans un module existant.

On doit clairement savoir dans quel module l'introduire, et si elle doit être dans l'objet **INIT** ou un objet multi-instancié.

Si c'est dans l'objet **INIT** on fait comme avant:

- la variable est initialisée dans un setup qui initialise ce module, et pas retouchée après.
- elle n'est pas encapsulée dans un type dérivé.
- elle peut être passée à une routine via un module ou via un argument d'appel.

Si c'est dans l'objet **GEOMETRY**:

- la variable est initialisée sous **SUGEOMETRY**.
- elle est encapsulée dans un des types dérivés existant.
- elle peut être passée à une routine via un argument d'appel: **YDGEOMETRY** ou un sous-attribut de **YDGEOMETRY**.
- l'exemple suivant peut se profiler:
  - on rajoute une variable **NDIMXXX** dans **YOMDIM**.
  - on l'utilise dans une routine TOTO.
  - TOTO a en argument d'entrée **YDDIMV**, mais ni **YDGEOMETRY** ni **YDDIM**.
  - il est préférable ici de remplacer **YDDIMV** par **YDGEOMETRY**, et d'utiliser dans la routine **YDGEOMETRY%YRDIM** et **YDGEOMETRY%YRDIMV**.
  - la nouvelle variable va être utilisée sous la forme **YDGEOMETRY%YRDIM%NDIMXXX**.
- cette variable ne doit pas apparaître dans le set-up de l'objet **INIT**.

Si c'est dans un autre objet multi-instancié:

- la variable est initialisée dans le set-up approprié.
- elle est encapsulée dans un des types dérivés existant.
- elle peut être passée à une routine via un argument d'appel.
- on peut être amené à passer un argument d'appel plus général à une routine qui utilise la nouvelle variable, par exemple remplacer **YDDYN** par **YDMODEL**.
- cette variable ne doit pas apparaître dans le set-up de l'objet **INIT**.
- cette variable ne doit pas apparaître dans le set-up de l'objet **GEOMETRY**.

Une variable qui évolue avec le temps et qui est utilisée dans le modèle direct a de bonnes chances d'être dans l'objet **FIELDS**.

Concernant les objets passés en argument, il est souhaitable d'assurer une cohérence entre une routine, son TL, son adjoint, son miroir LAM (cela concerne aussi l'ordre dans lequel les arguments sont passés). Par exemple si on a:

```
SUBROUTINE TOTO(YDGEOMETRY,YDMODEL,..)
```

alors on doit avoir également:

```
SUBROUTINE ETOTO(YDGEOMETRY,YDMODEL,..)
SUBROUTINE TOTOTL(YDGEOMETRY,YDMODEL,..)
```

# 4 Comment introduire un nouveau module.

On doit clairement savoir dans quel objet l'introduire.

Si le contenu entre dans l'objet **INIT** on fait comme avant:

- on code une routine de setup qui initialise ce module.
- le contenu de ce module n'est pas retouché après.
- il n'y a pas d'encapsulation dans un type dérivé.
- une variable de ce module peut être passée à une routine via un module ou via un argument d'appel.

Si le contenu entre dans l'objet **GEOMETRY**:

- il est initialisé dans une nouvelle routine de set-up appelée sous **SUGEOMETRY**.

- ce nouveau module définit un nouveau type dérivé.
- les variables sont encapsulées dans ce nouveau type dérivé.
- l'utilisation dans une routine peut se faire par exemple via **YDGEOMETRY**.
- on peut être amené à passer un argument d'appel plus général à une routine qui utilise les variables de ce nouveau module, par exemple remplacer **YDDIM** par **YDGEOMETRY**.
- cette variable ne doit pas apparaître dans le set-up de l'objet **INIT**.

Si le contenu entre dans un autre objet multi-instancié:

- il est initialisé dans une nouvelle routine de set-up.
- l'appel à cette routine de set-up intervient après l'appel à **SUGEOMETRY**.
- il y a création d'un nouveau type dérivé, avec de l'encapsulation.
- le contenu peut être passé à une routine via un argument d'appel.
- on peut être amené à passer un argument d'appel plus général à une routine qui utilise les variables de ce nouveau module, par exemple remplacer **YDDYN** par **YDMODEL**.
- cette variable ne doit pas apparaître dans le set-up de l'objet **INIT**.
- cette variable ne doit pas apparaître dans le set-up de l'objet **GEOMETRY**.

Si le nouveau module contient des variables qui évoluent avec le temps et qui sont utilisées dans le modèle direct, ce nouveau module a de bonnes chances d'être dans l'objet **FIELDS**.