Object Orientation
In Post-Processing

After two years
of code refactoring:
where are we now?

R. El Khatib (Météo-France)
Plan

- Objects concerning the post-processing
- Methods with these objects
  - Forecast
  - Simple post-processing server
  - Multi-post-processor
  - FIELDS transformer for the 4Dvar assimilation
- Status in cycle 46
- Developments beyond OOPS
Objects concerning the post-processing

**Geometries**
- spectral dimensions,
- gridpoint dimensions,
- orography

**Fields**
- spectral fields,
- surface fields,
- Gridpoint upper-air fields

**Models**
- Dynamics,
- Physics,
- GFL attributes

**IFS initializer**
- MPI,
- spectral transforms, …
- and orphan things

**Post-processors**
- Interpolators, filters
A Forecast made from these objects

IFS_init

Sugeometry → Create_model → Fields_create || Read_fields

Model_step

Constructor

Method
Simple post-processing server

IFS_init → Sugeometry → Read_fields

Create_model → Sufpdata → Fullpos

Fields_create (metadata) → Allfpos → A Fullpos method

Output data → Method

Subfpos → Fullpos constructor

FIELDS constructor
(transportable) multi post-processor

**Constructors**
- Sugeometry
- Create_model
- Fields_create

**Methods**
- Fullpos #1 init
- Fullpos #1
- Output #1

- Fullpos #2 init
- Fullpos #2
- Output #2

**FIELDS**
- IFS_init

**Initialized**
Beware of the power of the objects!

THERE ARE LINKS BETWEEN OBJECTS:

- \texttt{YRMODEL\%YRML\_GCONF\%GEOM} => \texttt{YRGEOMETRY}
- \texttt{YRFIELDS\%GEOM} => \texttt{YRGEOMETRY}
- \texttt{YRFIELDS\%STATE\_MODEL} => \texttt{YRMODEL}
- \texttt{YRFIELDS\%YRGFL\%YGFL} => \texttt{YRMODEL\%YRML\_GCONF\%YGFL}

\textbf{My\_Beloved\_Method (YDGEOMETRY,YDMODEL,YDFIELDS)}

If (I am your father) Call My\_Beloved\_Method &
& (YRGEOMETRY\_1, &
& YRMODEL\_2, &
& YRFIELDS\_3)

If (the\_Force\_is\_with\_you) Call My\_Beloved\_Method &
& (YRFIELDS\_1\%GEOM, &
& YRFIELDS\_1\%STATE\_MODEL,&
& YRFIELDS\_1)
Fields transformer for the 4DVar assimilation (1/3)

Model # 1

Fields # 1 metadata

Fpos 1=>2

Geometry # 1

IFS_init

Constructors

Model # 2

Fields # 2 metadata

Fpos 2=>1

Geometry # 2
Fields transformer for the 4DVar assimilation (2/3)

- Fields #1 (computed)
- Fields #1 (interpolated)
- Fields #2 (metadata)
- Fields #2 (interpolated)

Integrate

Minimizer
Fields transformer for the 4DVar assimilation (3/3)

Output:
- geometry
- fields request

As Fullpos knows 2D arrays only, use of Fortran 2003 pointer remapping:
\[ n = n_1 \times n_2 \]
\[ P(1:n) \Rightarrow T(1:n_1,1:n_2) \]

Spectrally-consistent output FIELDS object

Input:
- geometry
- model physics
- fields

On each elementary 2D or 3D data array

ordinary Fullpos request for gridpoint interpolations

SOILB  interpolate
RESVR  interpolate
SNOWG  interpolate
VARSF  interpolate
GMV/GFL  interpolate
GMVS  interpolate

OUTPUT MODEL SPECTRAL FIT

METEO FRANCE
Post-processing refactoring status in cycle 46

- **Post-processing self refactoring complete**
  - ✔ Can be called from the OOPS layer
  - ✔ Consecutive multiple instantiation possible
  - ✗ I/O server support with multiple instantiations to be tested
  - ✗ Simultaneous multi-instantiations requires the refactoring of the spectral transforms

- **Test program for FIELDS object transformation available**
  - ✔ Robustness to be confirmed by intensive testing
  - ✔ May need optimization
  - ✗ Written in fortran ; not yet in OOPS
  - ✗ Vertical definition not fully encapsulated yet
    => change of vertical levels not possible yet
Post-processing developments beyond OOPS

- **Refactoring of internal methods:**
  - For optimizations
  - To simplify developments by newcomers

- **FIELDS object transformator:**
  - Can it be used in another context than 4DVar?
  - Code Fullpos-TL to transform FIELDS increments?

- **Multi-post-processors support:**
  - Possible in Fortran framework (define & swap namelists file)
  - Low frequency vs High frequency post-processing
  - Multi-couplers post-processing server
  - Could be coupled with the I/O server
Thank you
for your attention!