OOPS

Claude Fischer, Météo-France/CNRM/GMAP

FORTRAN re-factoring

- Pros (opportunities):
  - **Cleaner code**
  - Usually a positive feedback from scientists involved who acknowledge the effort and the outcome
  - In places, build a new, shared understanding of the code. Remove some old code that nobody understands anymore
- Cons (risks):
  - Broken configurations (not that many for the time being ?)
  - Difficult cycles, code merging (science v/s re-factoring)

A new programming language (for this community), C++

- Pros (opportunities):
  - Potential of OO coding: offers a new way to think programming in assimilation (closer to the algorithm, importance of clean interface specs)
  - **Test programs of base classes and more complex classes are available** => first step is to evaluate the constructor, the copy-er and the destructor of any object (for IFS => requires a clean set-up code for the OOPS-IFS objects)
  - OOPS already offers a framework for innovation: EnVar prototypes at MF
- Cons (risks):
  - Sustainability of the Fortran/C++ mix ?, of the OO and functional programming codes ?
  - Performance aspects
  - A code only for a few C++ experts (freaks ?) ?

How does OOPS invite/force us to reassess some Arpège/LAM configurations ?

Baseline issue: what to do with those configurations that are driven from the Fortran CNT-level, and have not yet an OOPS counterpart ?

- FEMARS (compute δx of two sets of FA files and write in GRIB files): becomes a candidate for EPyGrAM …
- Sensitivity computation (801-type) => should move to OOPS (ECMWF ?)
- CANARI: very open issue …
  - The Fortran re-factoring has an impact on conf 701, but seems so far « under control »
  - Not firmly decided whether this code should be « oopsified » …
  - Or develop a new surface analysis code, building on OOPS ?
- DFI: should move to the OOPS layer (requires the Model object to be finished)
- 923, other 9xy confs used in Arpège (Arome): ?

Management questions: timing

Shift of calendar (*but wasn't this to be expected ?*):

- Kick-off conference at ECMWF (Nov 2009): OOPS was announced as a 2 years project
- Real start around 2010-2011
- Implementation schedule at ECMWF (from last Board meeting): plan to have all research assimilation experiments under OOPS by end of 2017, and operations to move to OOPS in 2018
- But even *before this target*: we can decide to routinely run some test programs while building new cycles
- For MF&partners: be ready to move to OOPS *not more* than 1 joint cycle after IFS
- For partners: **first get-in-touch with concrete OOPS programs via the *test codes***; later move to OOPS at home when installing for the first time an OOPS-common cycle with MF

Management questions: collaboration on the codes

Expertise, migration, scope of collaboration:

- FORTRAN/C++ co-phasing:

⇒ three SCRs at present: we would like only one for OOPS-IFS & IFS/Arpège (for coordination, versioning, commitments)

⇒ Reviewing process of C++ code ?

- Move experimental environment of 4D-VAR (3D-VAR) from Arpège-Fortran to OOPS-Arpège: steps of (multi-incremental) VAR move from scripts to binary file, but a progressive approach seems feasible (keep scripting level unchanged for the start ?).
- Scripts per se: Namelists + JSON (instead of XML); scripts will become simpler; accommodate OLIVE/Vortex (at MF) or build new bricks ?; PyOOPS ??

Project management & expertise

- Yannick Trémolet is the major OOPS code expert at ECMWF.
- Issue for the mid-term: how will the central OOPS code (at ECMWF) be managed ? (IFS-like or specific project ?)
- ECMWF have approached various other partners to join OOPS (NCAR, INRIA, Cerfacs, etc.). At present, no clear view on who actually would participate in the further development of the OOPS abstract layer (except Cerfacs probably). Should there later be an OOPS governance ?

Other changes for the not-so-distant future …

The IFS is going to progressively experience a number of other "exogenous" code imports:

- ATLAS/MIR: structured/unstructured grids and data handling methods, including parallelization (refer to ESCAPE WP4). Written in C++
- COPE: observation pre-processing software, BUFR2ODB, model-related QC with an interface to OOPS
- (GRIB_API, not really new)
- any other ?

⇒ Rather a matter of integration than phasing ?
⇒ development is delegated to specific (non-NWP) expert teams