COPE

An Overview

Eoin Whelan [Met Éireann], Mats Dahlbom, Bjarne Amstrup [DMI]
Frank Guillame [Météo France]

Or, a stroll down...

- Genesis
- Project definition
- Practical considerations
- HIRLAM/MF involvement
- Our plans for this year

The genesis of COPE

COPE genesis: ECMWF OD memo 2011

- Proposed by D. Vasiljevic and A. Hofstadler in May 2011
- Reorganise observation processing related responsibilities between RD and OD in ECMWF
- Currently quality control can be found in several places
  - pre-processing
  - extern pre-screening
  - screening within assimilation
- A common framework for all observation processing ... COPE
- The COPE framework will make observation processing and quality control more transparent and coordinated.

COPE genesis: goals

- Perform a substantial part of the observation processing before the cut-off time instead of after
- Extend the ODB domain so that most of observation processing tasks use ODB rather than BUFR
  - experiments could use archived ODBs rather than BUFR
- Enhance early detection and handling of observation anomalies that could cause failures in the operational suite and dissemination delays, and develop automatic decision mechanisms to exclude such observations.

COPE project description

COPE: Objectives

(Project Initiation Document)

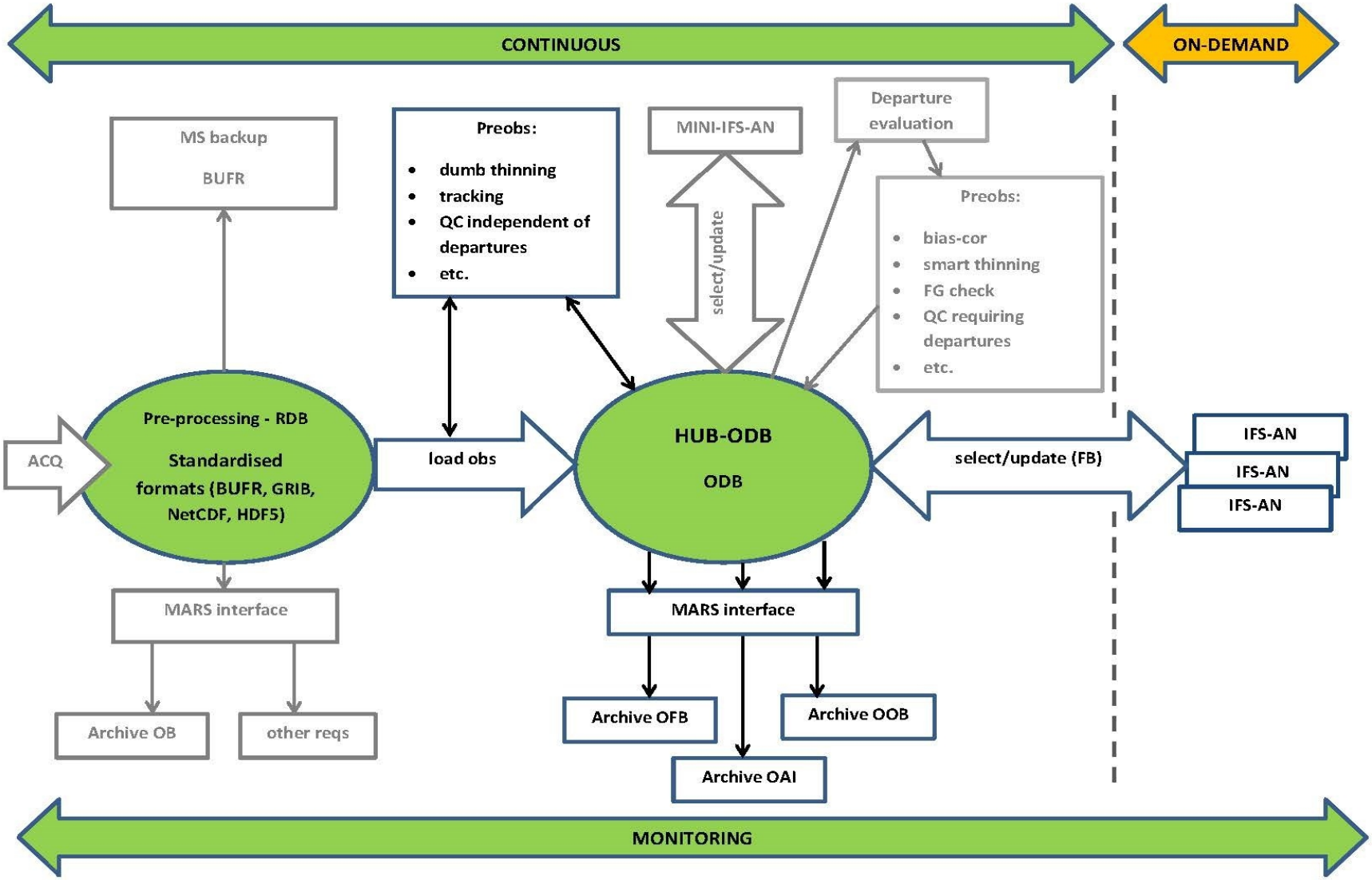
- Finish end 2014
- Enable continuous extraction, pre-screening, monitoring
- Externalise pre-obs and parts of screening to improve analysis scalability
- Facilitate collaboration with HIRLAM, ALADIN & Météo France on preprocessing and ODB loading of observations
- Prepare framework for departure dependent externalised screening and continuous monitoring
- Investigate if single ODB format and library can be used throughout the observation processing chain (adopt if possible)

COPE: Project exclusions

(Project Initiation Document)

- Some more advanced aspects of COPE are excluded
- The externalisation of pre-obs functionalities requiring an evaluation of departures is not part of COPE (in 2014):
  - smart thinning
  - FG check
  - advanced monitoring
- This is to allow a completion and implementation of the basic COPE framework within the next 12-15 months.
- The full potential of COPE can only be realised if there is a follow-on project to address these additional aspects.
- This work will also be necessary to achieve a better level of scalability for the assimilation

COPE: (ECMWF) overview



COPE: Work packages
(Project Initiation Document)

- **WP1 ODB:**
  - Hub-ODB
  - single format/library
  - ODB loading
  - ecCodes to decode obs data used and to encode in ODB format
- **WP2 Externalise QC**:
  - externalise pre-obs, pre-screening and QC
  - loading or update to/in Hub-ODB
  - plan for future upgrades to use analysis departures for “intelligent” QC
- **WP3 Observation monitoring:**
  - Integrated observation monitoring from acquisition to feedback
- **WP4 MARS:**
  - New ODB observation types in MARS
  - New MARS clients/features for SAPP and ODB
  - NMS backup functionality; feeding RD experiments and e-suites

COPE: Hub-ODB

- A set of ODBs "merged" together
- Hub-ODB server will be used for post-processing as well as to start the analysis.
- Feedback from blacklisting, screening, ... added to the ODB server
- Hub-ODB requirements and technical details are still being discussed
- “... fancy name for a high-availability cache of observations ...”
- Users of Hub-ODB:
  - Operations
  - Research using same observations as operations
  - Research using observations from HUB-ODB not used in operations
  - Research using observations not in HUB-ODB
  - External users (MF, HIRLAM, other NMSs)

COPE: NMS implementation

- Emphasis in the Project Implementation Document on NMS implementation ...
- Objective: “Facilitate collaboration with HIRLAM, ALADIN, Météo France and other NMS on pre-processing and ODB loading of observations”
- WP1: “Hub-ODB … service for (ECMWF) RD and NMS users”
- WP1: “ecCodes decoder supporting all observation types used by ECMWF and by external COPE partners”
- WP4: “Implement NMS backup functionality using new MARS clients for SAPP and ODB”

Practical Considerations

COPE: Practical Considerations

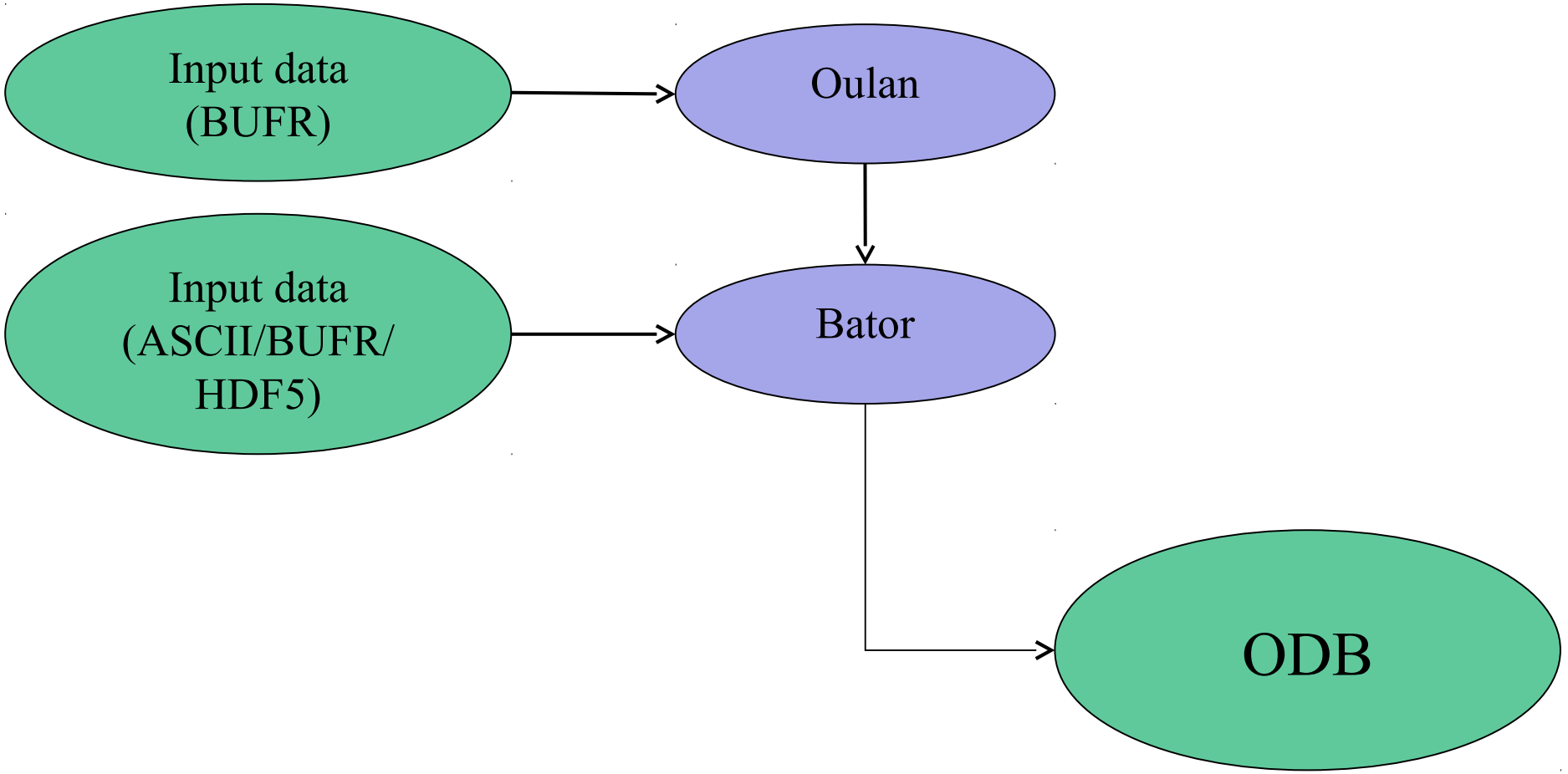
- Source code maintained in ECMWF GIT repository
- COPE details/documentation available on ECMWF wiki
- Access to GIT and wiki granted on user by user basis
- COPE code included in CY40 release
- COPE compiled using CMake and make
- COPE requires the following:
  - C++ compiler
  - CMake
  - ODB-API
  - EcLib
  - ecBuild
  - bufrdc (optional)

COPE: Practical Considerations

- Current inputs formats: ODB-2, BUFR, ASCII
- Current output formats: ODB-2, ASCII
- COPE operates on data cast into ODB-2 space
- C++ filters process and quality control observation data

- Observation processing happens in the “COPE Filter Factory”
- Factory filters selected by the user via a JSON config file

Current data flow



COPE: Data flow

Input data (BUFR/ODB/ netCDF/HDF5)

COPE

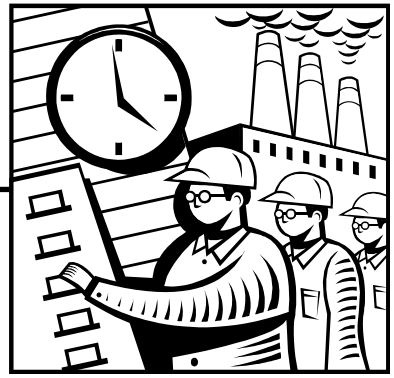
reportReader

report

COPE filter(s)

report

reportLoader

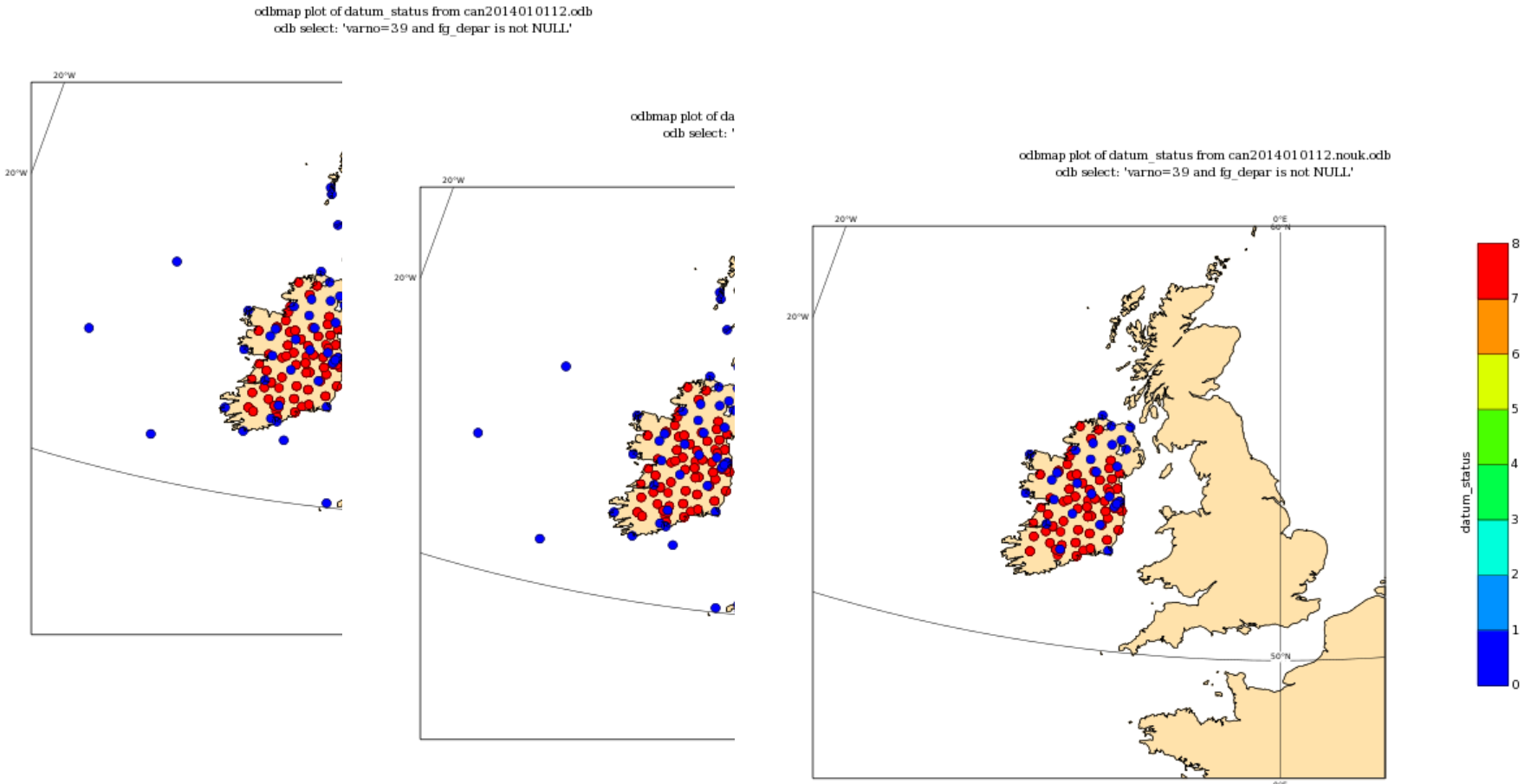


Hub-ODB

```
synopFilter
{
  "filters": [
    { "name": "LocationValidator" },
    { "name": "DateTimeValidator" },
    { "name": "InstrumentTypeAssigner" },
    { "name": "LandSynopVerticalCoordinateAssigner" },
    { "name": "BiasCorrector" },
    { "name": "WindComponentsAssigner" },
    { "name": "PrescribedErrorAssigner",
       "options": {
          "statistics_file": "../resources/error_statistics.csv"
       }
    },
    { "name": "RelativeHumidityAssigner" },
    { "name": "PrecipitationSplitter" },
    { "name": "FinalErrorAssigner" },
    { "name": "FinalChecker", "disabled": true }
  ]
}
```

COPE: Simple example

- Hacked “LocationValidator” filter ...
- Filter for LAM usage: min/max latitude and longitudes



COPE: Simple example

- Hacked “LocationValidator” filter ...
- Filter for LAM usage: min/max latitude and longitudes

```
for (View<Header1>::iterator it = records.begin(),
          end = records.end(); it != end; ++it)
   {
      Record<Header1> record = *it;

<< CODE TO CHECK ODB COORDINATES>>

 if (error)
       {
          cout << "Found invalid coordinates::" << endl;
          record.lat=odb::MISSING_VALUE_REAL;
          record.lon=odb::MISSING_VALUE_REAL;
          records.update(record);
       }
   }
```

My/Our COPE Plans

Plans for 2014

- Implement local NMS BUFR loaders in COPE
- Devlop LAM specfic filters and other filters not deemed necessary by ECMWF
- Compare COPE ODB with Oulan/Bator ODB
- Implement OPERA HDF5 loader in COPE

Fin

Questions?