CMS
Database Strategy
moving from development to operation

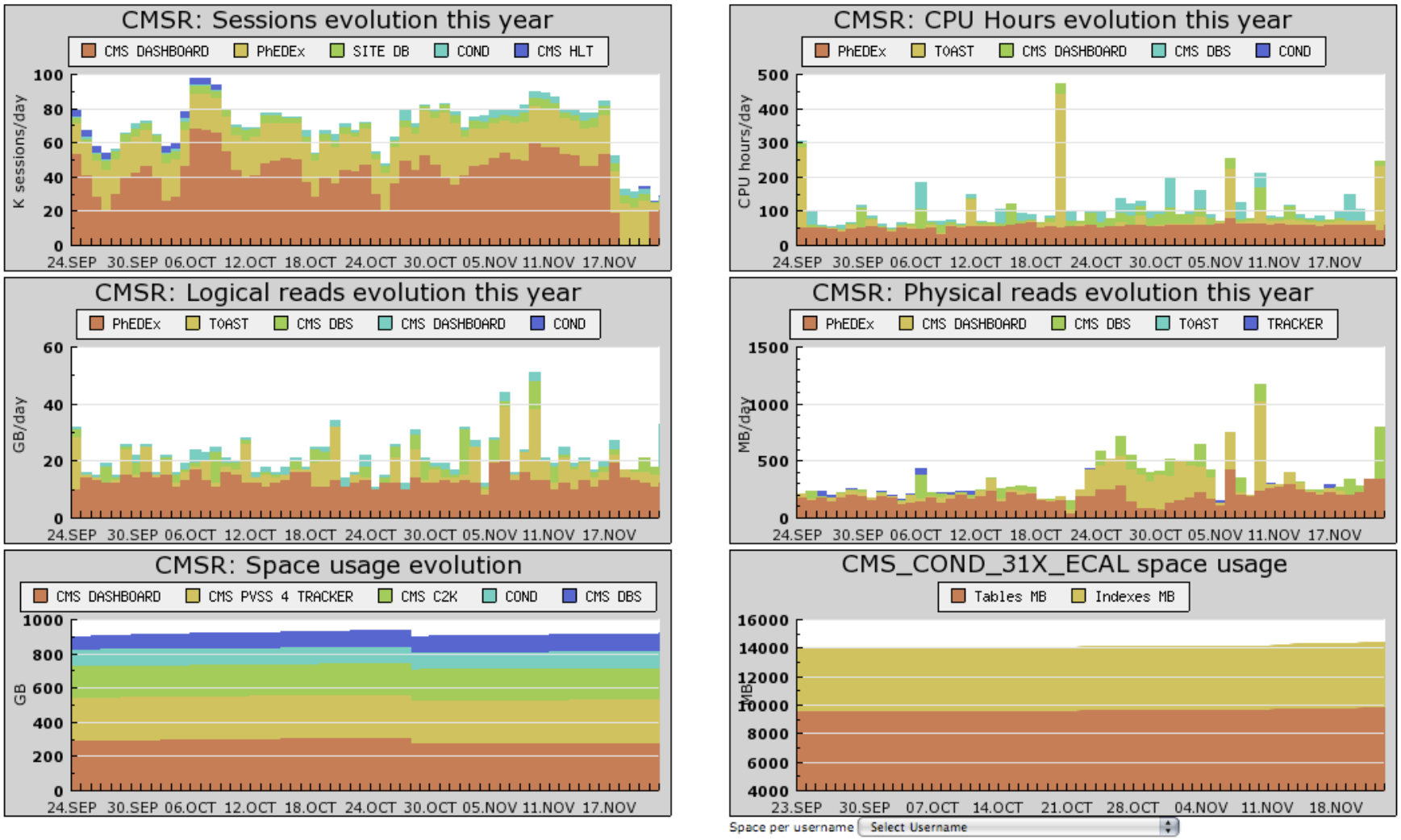
Vincenzo Innocente
CERN PH/SFT

DataBases in CMS

- Relational Databases are pervasive in CMS software applications
  - Used as strategic and tactical storage in many distributed application for logging, workflow management, configuration management, document storage etc
  - Applications are rather independent of each other
    - no central authority
    - small development teams
    - limited scope
  - Notable exception: Condition Database

Offline RAC Usage

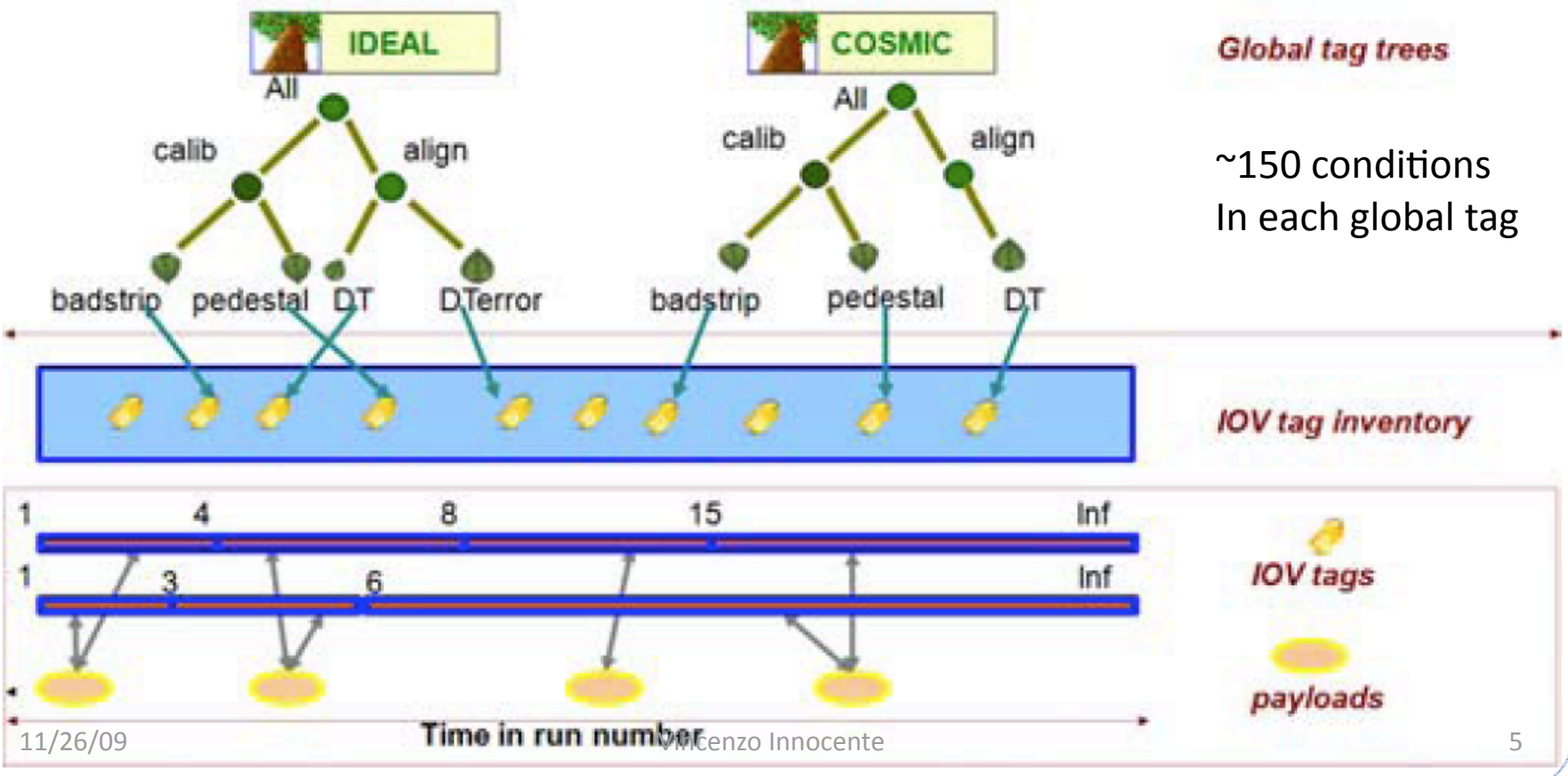
CMS Offline RAC



CONDITION DB DESIGN

Condition DB Structure

2. **Conditions DB content**: time-variant condition data (calibration and alignment) are stored in the **Condition Database** together with their sequences versions **IOV**. IOV sequences are identified by tags (IOV tag) which are organized as trees (tag tree). A tag tree can be traversed from any node (global tag).



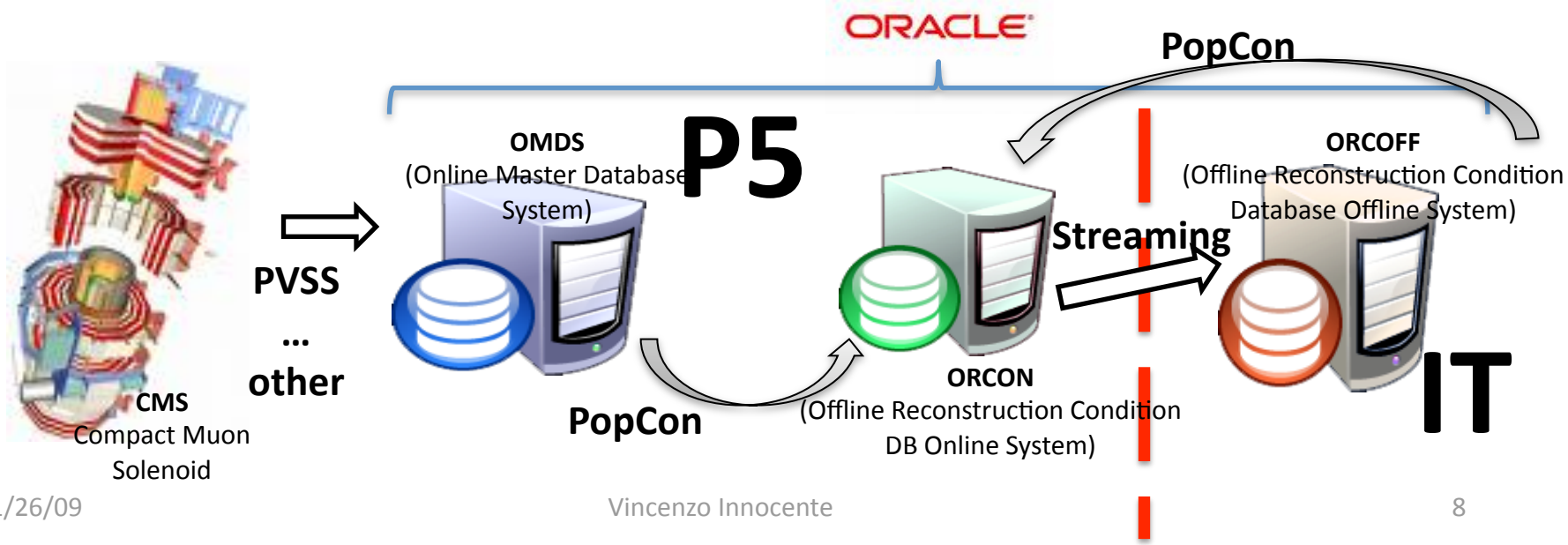
Relations in conditions

- All relations in conditions DB are purely logical and application specific
  - No RDBMS consistency enforced
  - Full flexibility in copying (deep and shallow) at all level of the structure
  - Simple policy: NO DELETE, NO UPDATE
    - Only the current IOV-Sequence can be extended
  - Payloads implemented as POOL/ORA objects
  - Management through application specific tools

POPULATION STRATEGY

Condition DataBase Population

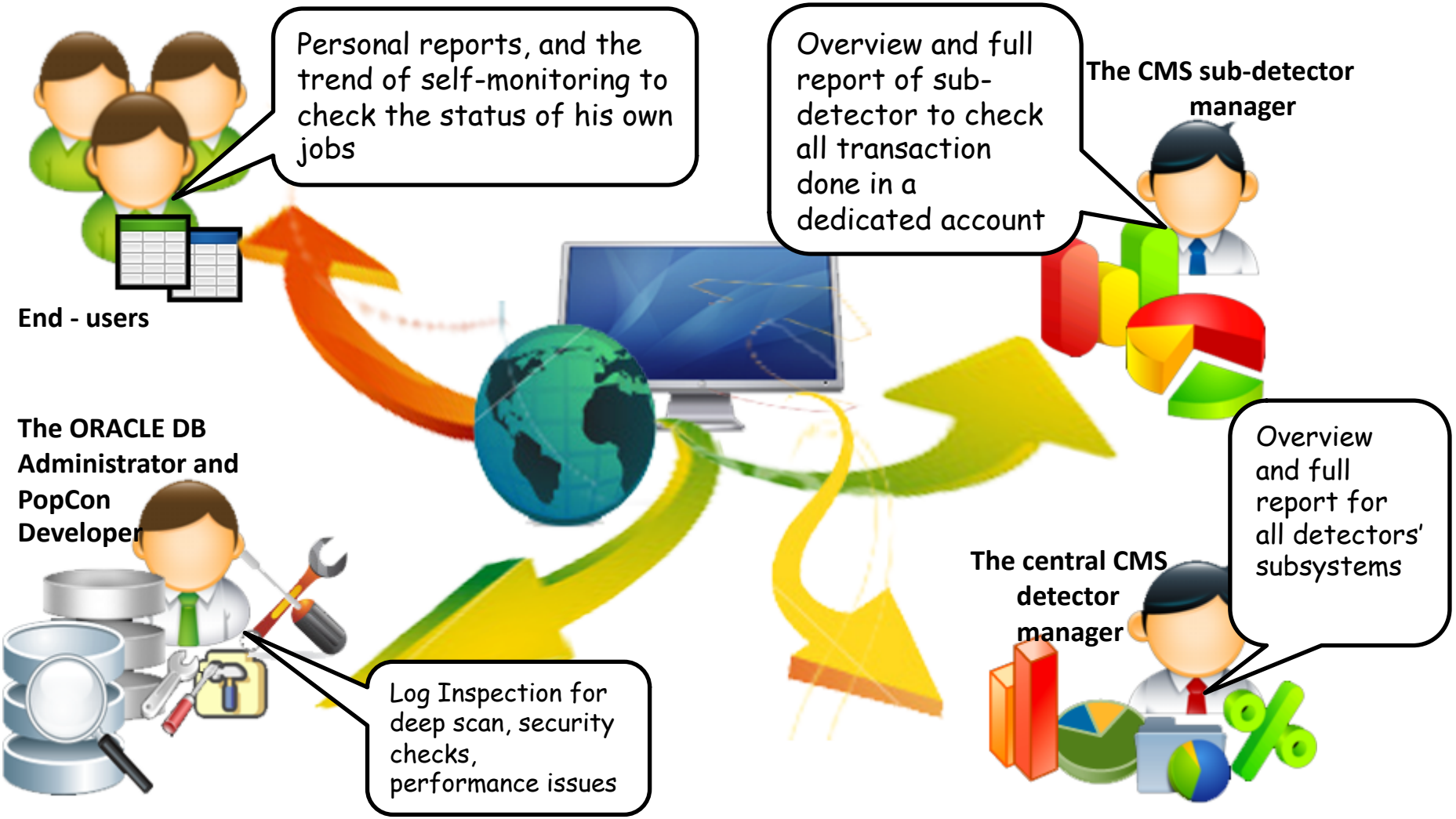
- PopCon (Populator of Condition Objects tool):
  - is an application package fully integrated in the overall CMS framework intended to transfer, store, and retrieve condition data in the Offline Databases;
  - Assigns metadata information (tag and IOV).
- CMS relies on three ORACLE databases for the condition data.



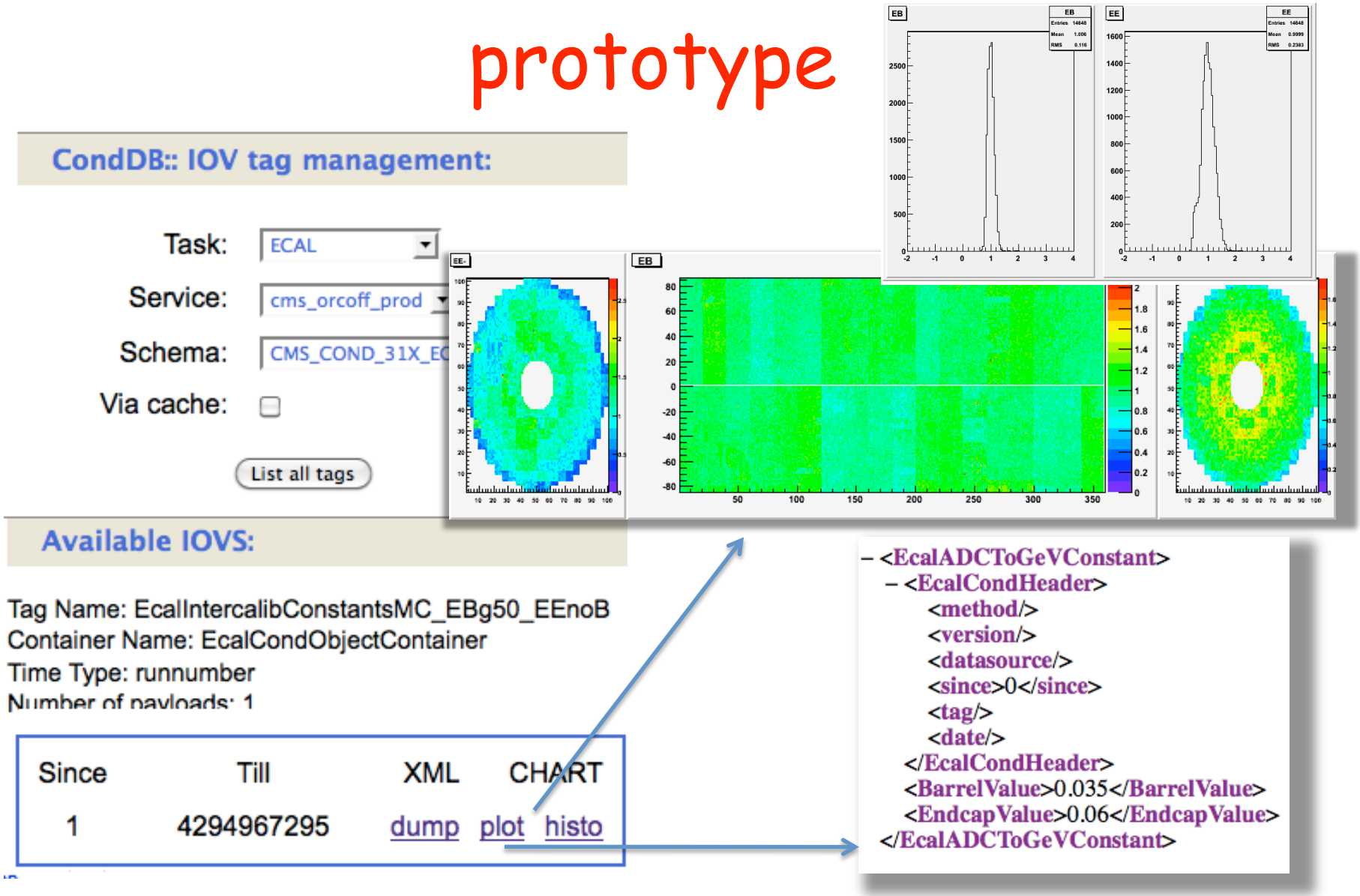
Centralized Population of Condition Databases

- Two possibilities for each sub-detector:
  - Run automatically the so-called O2O application that reads from any online source, assigns tag and IOV and uploads data in the dedicated ORCON account (condition data);
  - Dropbox (calibration data): users copy data in SQLite format into a dedicated folder, then these data are automatically exported to the sub-detector's ORCON account.
- PopCon transfers data into the DB accounts:
  - Creates log information stored in a DB account.
- Watchdog to monitor automatic jobs' status:
  - Monitoring information stored in the DB.

Web Monitoring from different users' perspectives

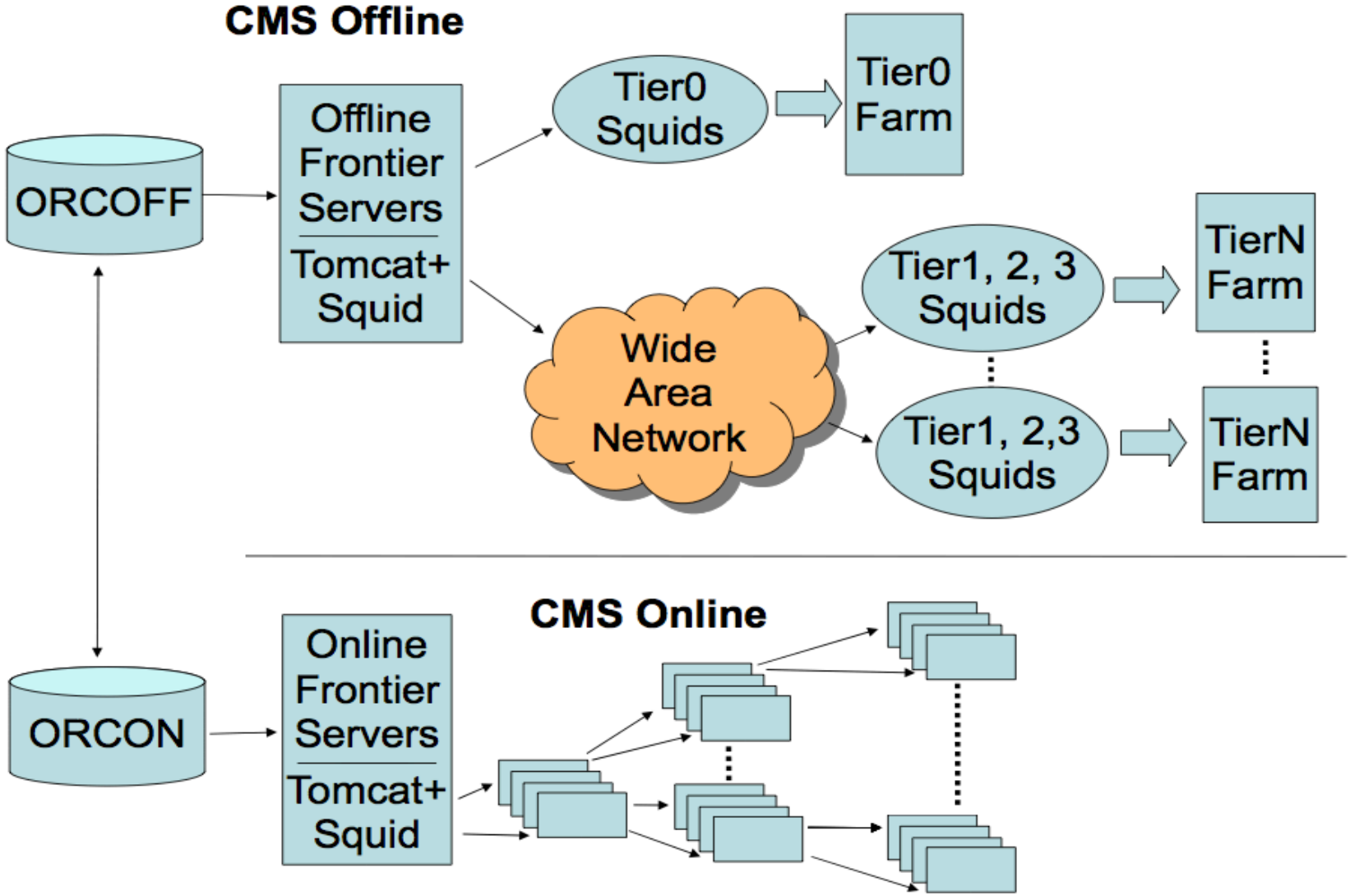


CondDB web server - GUI prototype



ACCESS STRATEGY

Data Access using web caches (FronTier)



PHYSICAL LAYOUT

“Partitioning Strategy”

- Till now the database has been “partitioned” into accounts following development and deployment criteria
  - Keep separated areas of independent development
    - By sub-detectors, by software release
- Move to “partitioning” by use-cases
  - Keep separated independent use-workflow
    - MonteCarlo: copy all relevant data into a dedicated account
      - Even a single sqlite file!
    - Re-processing at remote Tiers: make a read-only snapshot of the whole condition DB and use that (through FronTier)
    - Prompt processing (reconstruction, calibration, analysis at T0 and CAF) keeps using the master DB (updated in real time)

Open issues

- As the Database grows and applications evolve more “partitioning” issues will show up
  - Manage “schema evolution”
  - Use of oracle partitions
  - Archive and pruning
  - Snapshot frequency

- Add new “dimensions” and/or “abstractions” at the application level

Summary

- CMS has in production a large number of applications critically relying on DB backends
- Condition Database adds complexity challenges due to its close connection to “physics” applications
- Stable LHC operation will add new challenges and new dimensions to the project
  - Balance between development and production
  - Performance issues
  - Robustness and flexibility issues