CHEP'07
5 September 2007

LHCb Distributed Conditions Database

Marco Clemencic
marco.clemencic@cern.ch

Overview

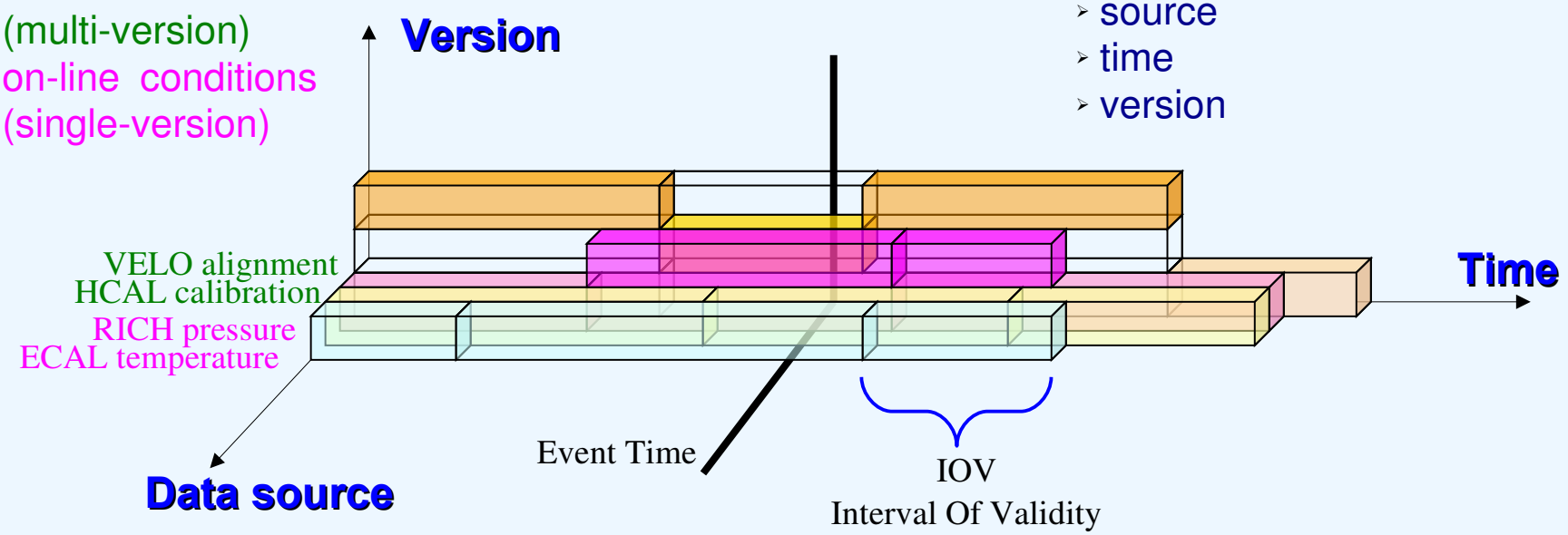
- The Conditions Database
- LHCb CondDB Structure
- LHCb Computing Model
- CondDB Deployment Model
- CondDB in Production
- Tests
- Conclusions

The Conditions Database (CondDB)

- Database for time-varying non-event data (Conditions)
- LHCb uses LCG developed library COOL

2 types of conditions:
- off-line conditions (multi-version)
- on-line conditions (single-version)

3 degrees of freedom:
- source
- time
- version



COOL

- LCG project for Conditions Database (http://lcgapp.cern.ch/project/CondDB)
  - Main developers
    - Andrea Valassi
    - Sven A. Schmidt
    - Marco Clemencic
- C++ API, Python interface
- Database application logic in the client
- Based on the LCG project CORAL

CORAL

- LCG project for Database Access (http://pool.cern.ch/coral)
  - C++ API, Python interface
  - SQL-free
  - Back-end independent
    - Supports: Oracle, MySQL, SQLite, Frontier
- Advanced functionalities
  - replica selection and authentication
    - XML files
    - LGC File Catalog LFC

LHCb CondDB Organization (1)

- LHCb CondDB contains
  - detector description
  - conditions needed for reconstruction/analysis
- Th CondDB is used in
  - High Level Trigger (HLT)
  - Reconstruction
  - Analysis
  - Simulation

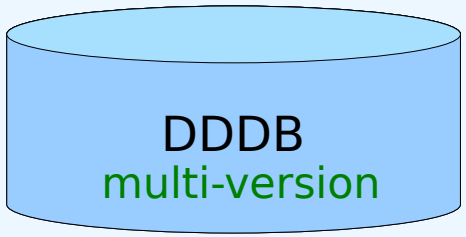
LHCb CondDB Organization (2)

- Each use-case has different needs
  - High Level Trigger
    - no variation / stable running conditions
  - Reconstruction / Analysis
    - best knowledge of detector status
  - Simulation
    - possibility to generate with different conditions
- We need to insert data generated both on-line and off-line

LHCb CondDB Organization (3)

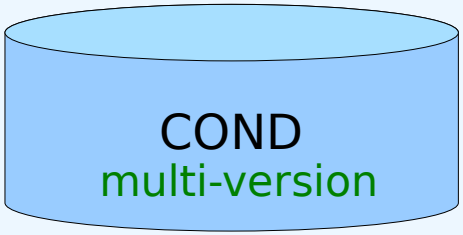
Plain Detector Description

common between reconstruction and simulation



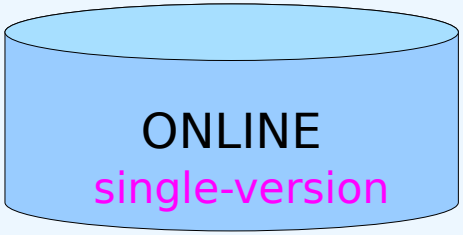
Off-line Cond.

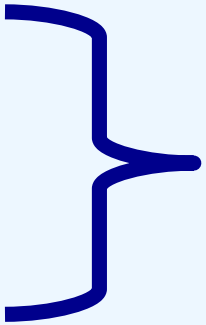
writable from Tier-0

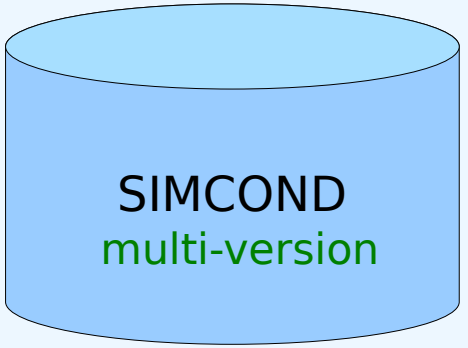


On-line Cond.

writable from experimental area



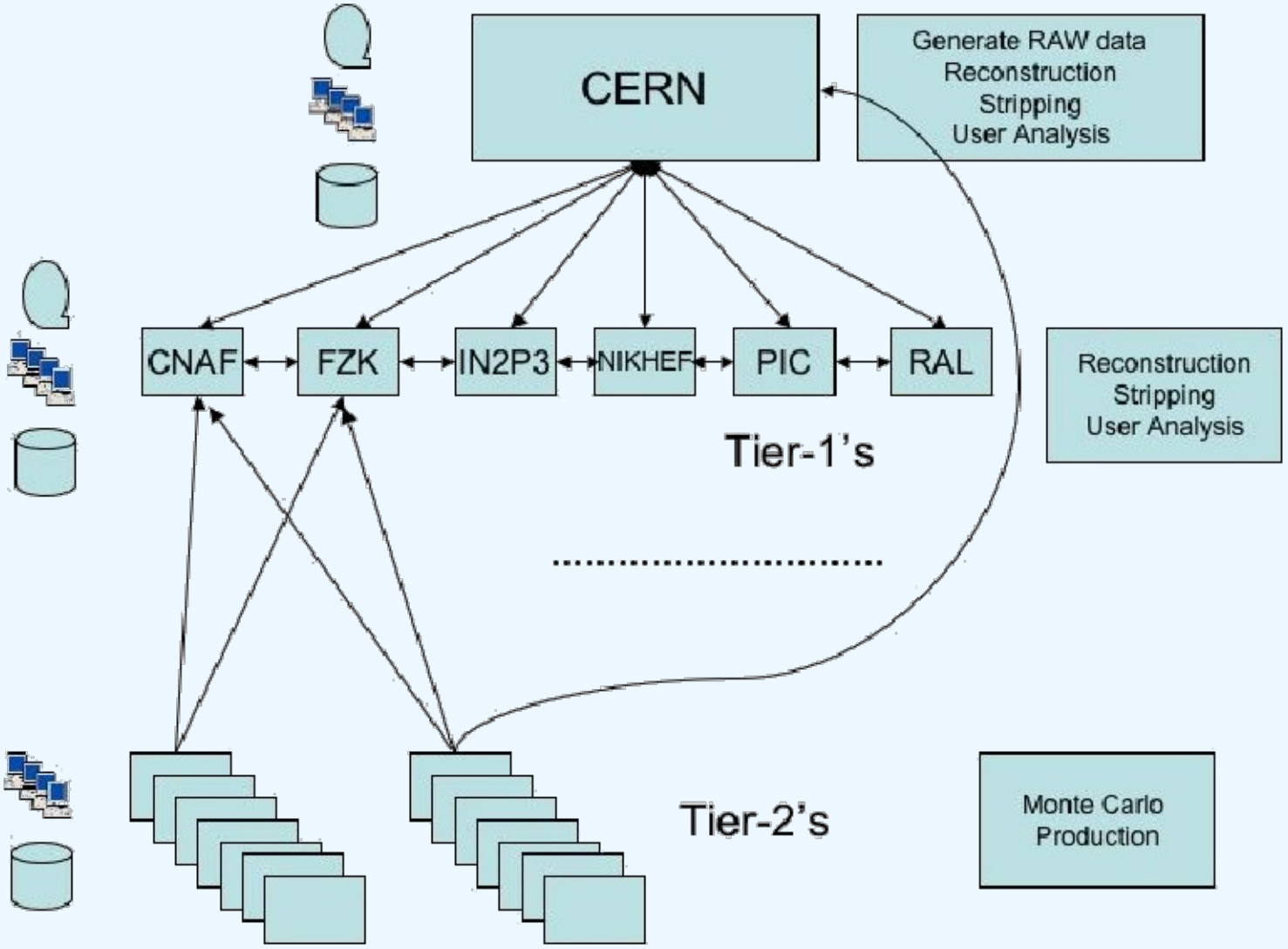




Cond. for Simulation

same structure of a merge of off-line and on-line

LHCb Computing Model



produces and consumes conditions

consumes conditions

needs a limited subset of conditions

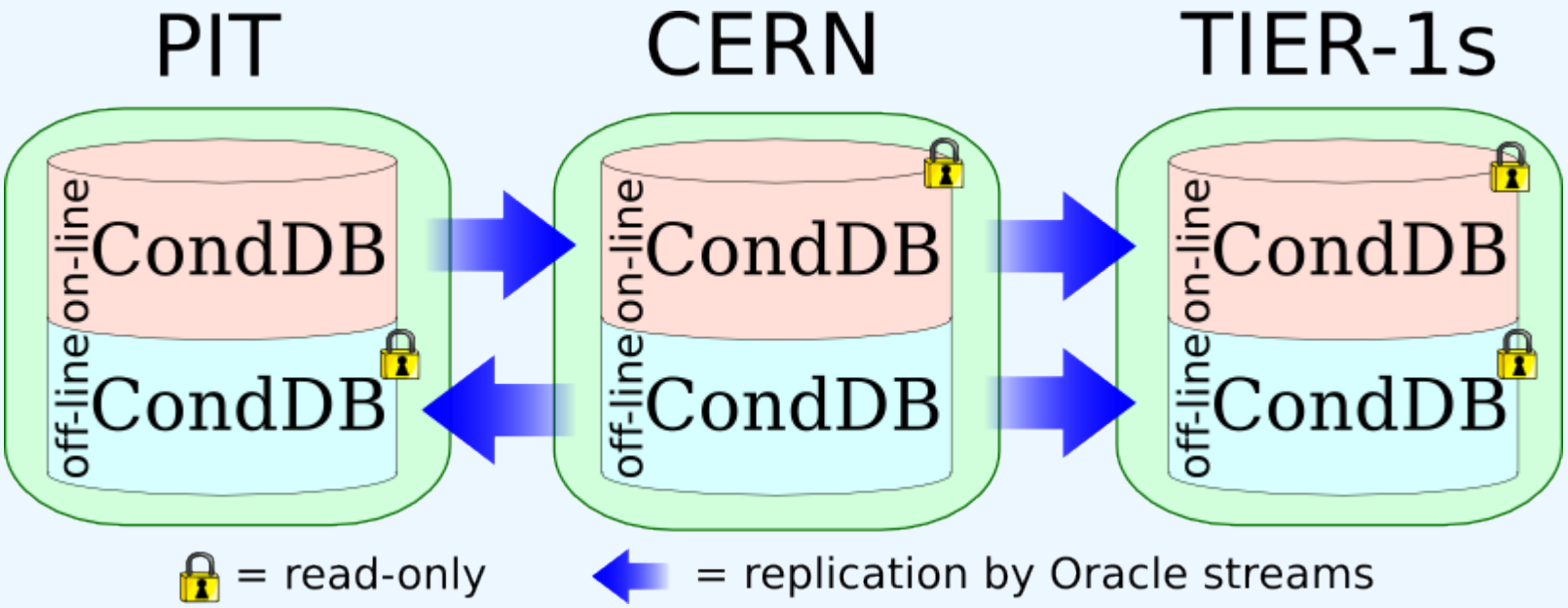
Usage/Access (1)

- High Level Trigger
  - ~4000 processes and 1 Oracle server
  - Uses only a well defined subset
- Simulation
  - Runs at Tier-2s
  - Oracle servers only at Tier-1s
  - Uses only a well defined subset
- Use SQLite based snapshots

Usage/Access (2)

- Reconstruction/Analysis
  - Run at Tier-1s
  - Non-predictable needs
- Direct access to Oracle replicas hosted at Tier-1s

CondDB Deployment Model



- Size:
  - Few GB
- Write:
  - PIT & CERN only
- Read:
  - few hundreds conn. per site
  - read 50-100 MB

Oracle Streams Set-up

- The replication of the CondDB is achieved with Oracle Streams
- CERN to Tier-1s
  - Streams shared with LFC replication
  - Using downstream capture box
    - LFC: real-time replication
    - CondDB: periodic capture of change logs (currently every 30 min.)
- CERN to/from PIT
  - Direct streams connection (real-time replication)

Access from the GRID

- Problems:
  - Authentication
    (Oracle does not accept GRID proxy certificates)
  - Closest replica discovery
- Solution: use LFC
  - Via CORAL advanced functionalities

CORAL LFC Replica Service

- CORAL provides means to store and retrieve from LFC all the informations needed to connect to all replicas of the CondDB
  - Connection details
  - (DB) User credentials
- The closest replica is chosen with a call-back function passed to the CORAL library

Site Availability/Selection

- The version of the CondDB to be used is defined with a *tag*
- We need to be sure that the required tag is available in the site DB before allowing jobs (i.e. it has been replicated)
- We use LCG software tags published at the site after the local CondDB has been validated

CondDB in Production

- COOL based Detector Description in use since March 2007
  - Informations stored as XML files
  - Using SQLite while waiting for Oracle production set-up
  - Data volume (initial phase)
    - DDDB: 18MB, 411 objects (2448 files)
    - COND: 8MB, 74 objects (74 files)
- Switch to Oracle planned for September
  - SQLite still used for local analysis (laptop)

Tests

- Replication performances
  - Negligible latency at 1Hz insertion for 8h with peak of 100Hz using real-time replication
  - Reliable replication with 100 insertions every 2h using periodic capture
- DB Access performances
  - Read a full snapshot on a 2.8GHz Xeon
    - SQLite: ~25s (almost all CPU)
    - Oracle: ~135s (25s CPU, 110s latencies)
  - It is better to use SQLite when possible

Scalability test at CNAF

- ...

Conclusions

- The LCG projects COOL and CORAL are successfully used
- Oracle Streams replication demonstrated to be efficient and reliable
- The database hardware set-up tested at CNAF fulfill our needs
- SQLite is more efficient than Oracle for single client access on a small database