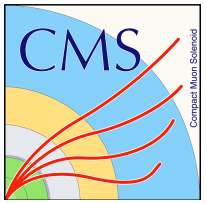


Designing a future Conditions Database based on LHC experience

D. Barberis[1], A. Formica[2], E J Gallas[3], G.Govi[4], G.Lehmann Miotto[5], A. Pfeiffer[6]

1. Universita' di Genova (ATLAS collaboration)
2. Irfu - CEA Saclay (ATLAS collaboration)
3. Department of Physics, University of Oxford (ATLAS collaboration)
4. Fermilab (CMS collaboration)
5. CERN (ATLAS collaboration)
6. CERN (CMS collaboration)

Irfu - CEA Saclay
Institut de recherche sur les lois fondamentales de l'Univers

Outline

- Conditions data @ LHC : learning from the past
  - review of solutions adopted in Run1 and Run2 (R1,R2)
  - common use cases
- Proposal for new architecture (ATLAS+CMS)
  - Main goals
  - Work packages definition
- Prototyping for Run3 (R3)

Conditions data @ LHC

What are the conditions data?

- We define them in general as “non-event” data varying with time.
  This definition includes for an LHC experiment:
  - Status and Configuration for Detectors and Trigger system, Run Information
  - Detector Calibrations and Alignments
  - Beam and luminosity informations

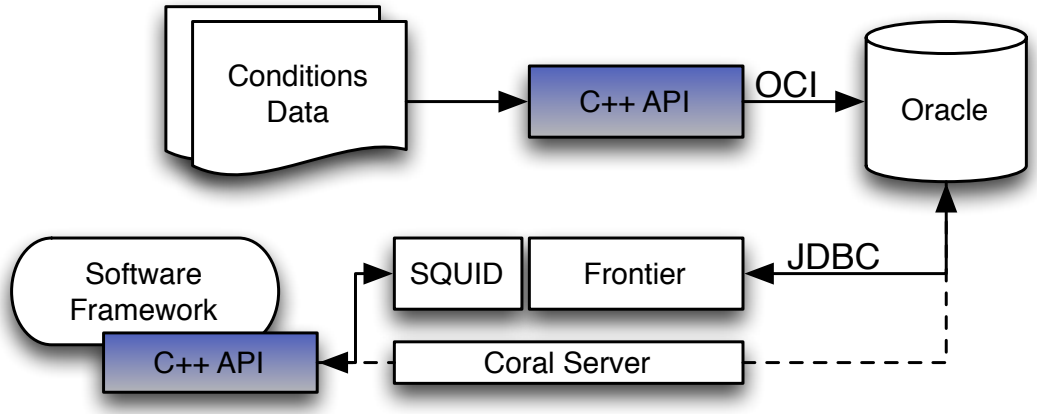
Data-flows

- Conditions data are needed at many stages of data handling
  - **Online** : includes conditions needed for online monitoring and data taking
  - **Express/Prompt Reconstruction** : in general the processing step at Tier0
  - **Reprocessing** : best set of conditions, available at TierN sites for large reprocessing campaign
- Conditions data are accessed by O(1M)/day jobs during standard data-flows

Experience from R1(..R2)

CMS and ATLAS infrastructure

- During Run1 both experiment have successfully deployed a conditions data infrastructure, based on the following architecture



1 Oracle account per system
O(1000) tables in total
>1 TB for each experiment

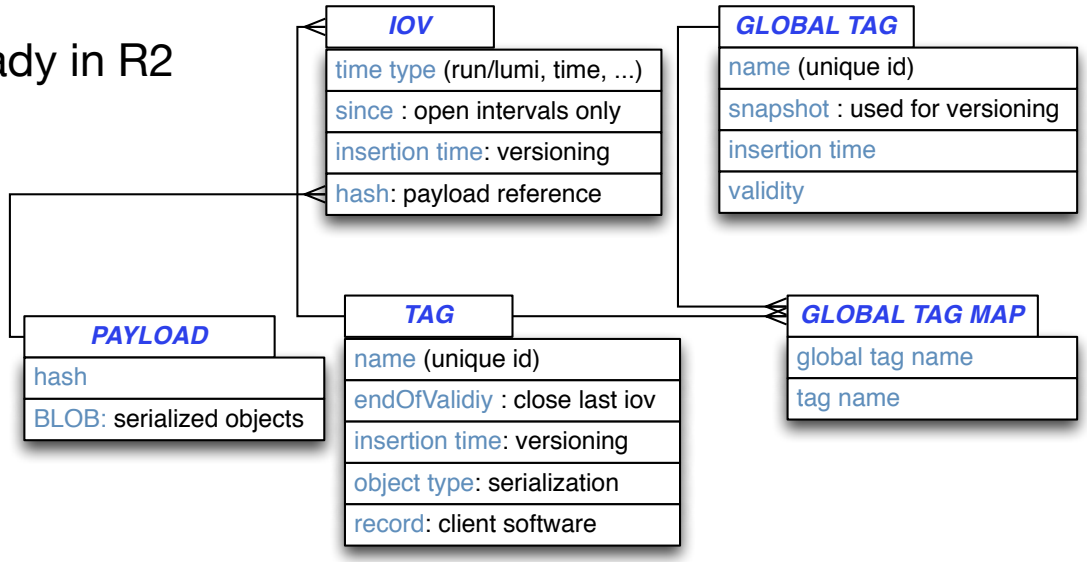
  - Usage of Cern-IT Coral for DB access from C++ clients (Coral + Cool for ATLAS)
  - Detailed mapping of Conditions Data in tables specific to every system

Considerations and potential limitations

- Heavy DB administration : large number of schemas and tables
- C++ software layer for DB access is difficult to maintain
- Limited number of DB plugins provided for Coral, limited monitoring possibilities
- User analysis conditions do not necessarily use this infrastructure

R2 infrastructure

- CMS : simplified relational structure
  - new DB architecture with 1 schema only and few tables
    - Metadata table structure : GlobalTag / Tag / Iov
    - Payload table structure : 1 table (containing serialised C++ objects)
- ATLAS : metadata extraction
  - Organise metadata information in a unique schema for a better management of global tags and tags dependencies (added into COMA since end of R1)
- *Common* Data model
  - applied by CMS already in R2



Payload Hash

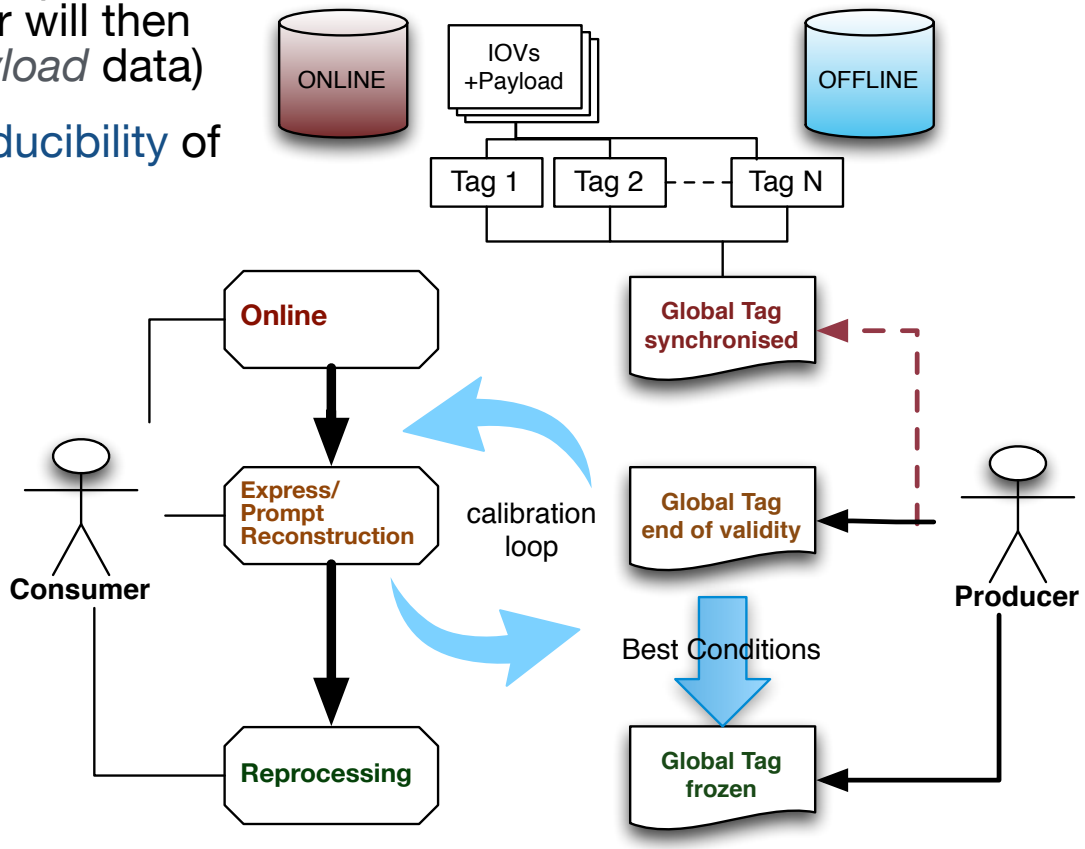
eliminates duplicates

Common use cases (ATLAS+CMS)

- *Consuming* Conditions Data
  - Specific workflows access conditions data using *Global Tag* as entry point
  - Global Tags provide direct access to specific system *tags*, for which the consumer will then retrieve appropriate *IOVs* (hence *payload* data)
  - Global Tags should guarantee reproducibility of processing steps

- *Online (HLT)*
  - Conditions are synchronised from offline
  - Last valid IOV is the current run

- *Offline (Prompt Reco/ Reprocessing)*
  - Conditions are updated during calibration loop : end of validity is last processed run
  - Reprocessing needs stable conditions



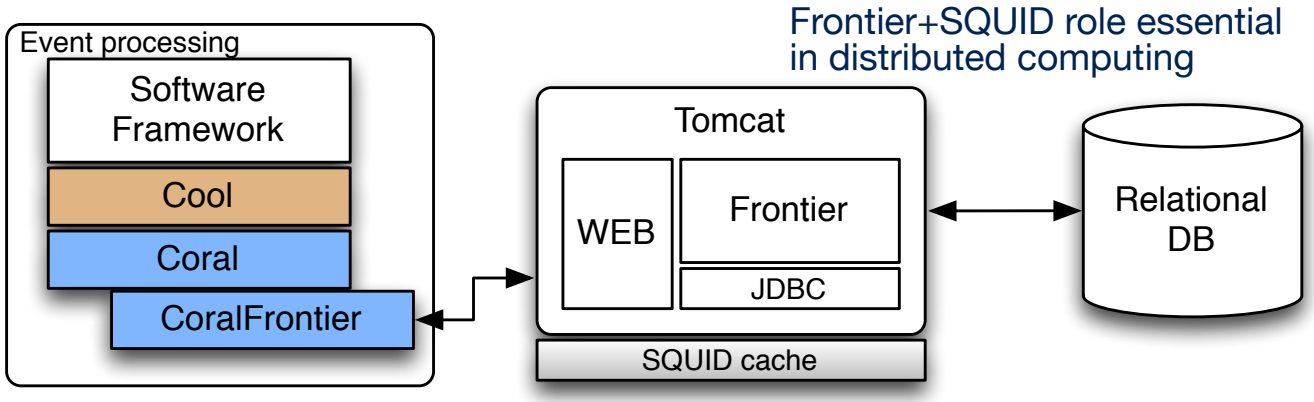
What do we want to address ?

- Future infrastructure should handle :
  - Official Data-Flows from data taking to large reprocessing
    - experience from Run1/2 allow us to converge on common use cases
  - MC productions
    - conditions for MC have very similar need respect to those for data processing
  - User analysis
    - Run1/2 architecture has several limitations for this use case: users do not access the DB from their own laptops in general
  - **General requirements**
    - Architecture supporting multiple database backends, transparent for the infrastructure
    - Simple installation on many platforms
    - Web based monitoring of the services
- We do not want to replace database systems in general
  - Monitoring of fast changing detector parameters (e.g. HV,LV) is in general managed via sophisticated control systems infrastructures
  - Detector monitoring and configuration requires ad hoc set of relational tables

Proposal for a new architecture

- Review software architecture
- Expand the role of the intermediate server
  - usage of a
    handling at the level of the central server
  - use a
    web access and increase the multi language support
  - use standards from IT industry in order to support multiple DB platforms in a transparent manner

OLD



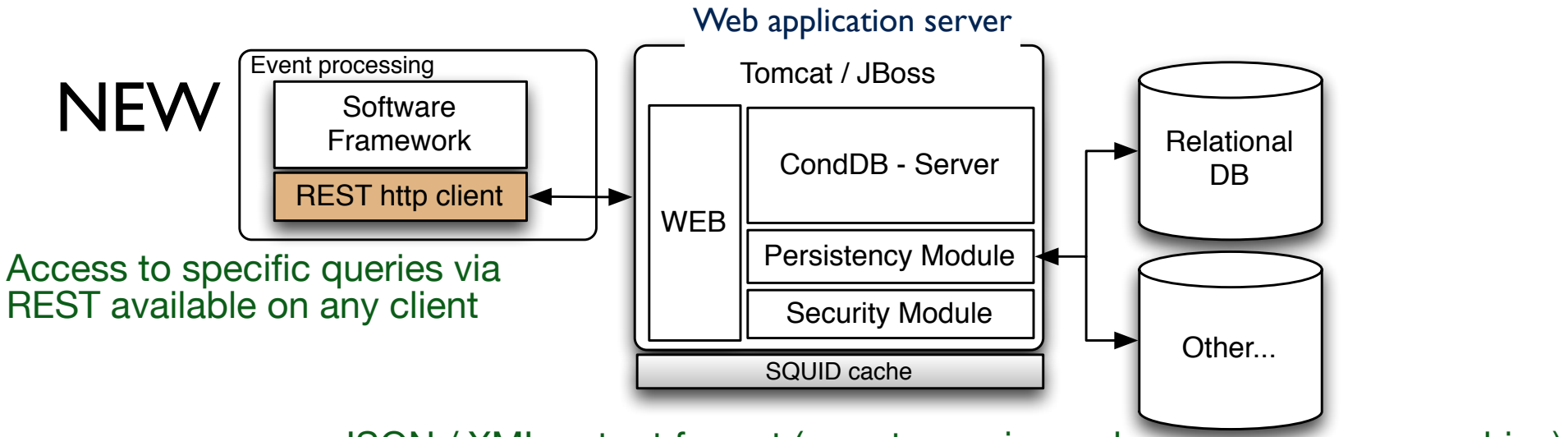
Complex client libraries integrated in offline software frameworks

Light server to send generic queries to the DB via JDBC

Complex installation of client software on user machines

Proposal for a new architecture

- Review software architecture
- Expand the role of the intermediate server
  - usage of a **multi-tier model architecture** providing all business methods for DB handling at the level of the central server
  - use a light software client (*curl*-like) during event processing : profit more of the REST web access and increase the multi language support
  - use standards from IT industry in order to support multiple DB platforms in a transparent manner



ex: JSON / XML output format (easy to use in any language on user machine)

WP1 : payload storage

- Core component:
  - **Payload choice** : aggregate set of objects stored via BLOBs in relational DB
    - Simple DB table structure
    - e.g. : we can handle several C++ types without any need to expose our system to the complexity of the detector needs
    - Do not perform queries selecting specific payload fields content
      - From Run1 experience this is a minor need for LHC experiments
- R&D directions for payload storage
  - Optimal payload structuring solutions :
    - study of object serialisation
  - Alternative payload storage solutions :
    - Explore solutions based on NoSQL technologies
    - Simple file-system storage can also be an option in some cases

WP 2/3: core services and metadata

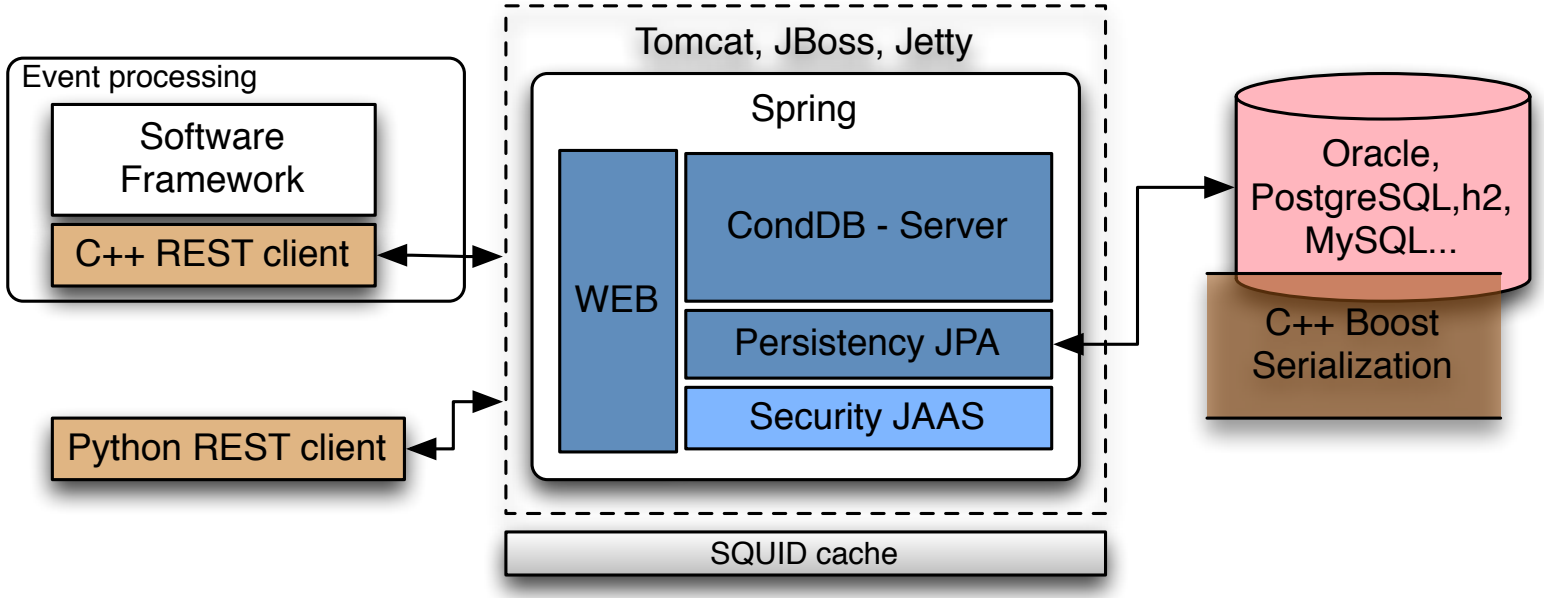
- Metadata catalogue
  - Manage the informations describing different systems interacting with the Conditions infrastructure
  - Provides authorisation functionalities
- Core services
  - Implement all base services on top of the data model : insertion and retrieval of the conditions data
  - All core services are deployed in the middle tier
  - Persistency layer deals with DB access, DAOs implement the queries identified as important for the different use cases, exposing web services based on REST architecture to the client

WP 4,5,6 : monitoring

- Storage monitoring
  - Interact with previous services to gather information on data volume or time coverage for set of conditions data
  - Critical to understand conditions volume growth and completeness
- Conditions Server monitoring
  - Monitor the server access
  - Critical to understand bottlenecks and typical patterns in the client access
- Transactional storage and monitoring
  - Use core services and user job logging to provide monitoring on accessed conditions inside data-flows
  - Might best be handled via a non-relational storage system

Prototyping for R3

- Developing core services for a prototype
  - Java based prototype



- Tools
  - Build system: **Maven**
  - Versioning : Cern **git** server
  - Integration Server: **Jenkins**

Outlook

- Planning for R3 Conditions Data architecture
  - 2015 : people busy for the moment with R2 …
    - Validate the prototype architecture : tests using “real” conditions
    - Explore Payload storage solutions
  - 2016 : start development following the identified working packages
    - Benchmark and migration of existing data has to be performed well in advance of R3 (2018/2019)
- Collaboration between ATLAS and CMS
  - The identification of common use cases has triggered the will to mutualise efforts in this area of software development
  - Project started inside these 2 experiment but could be interesting for others as well