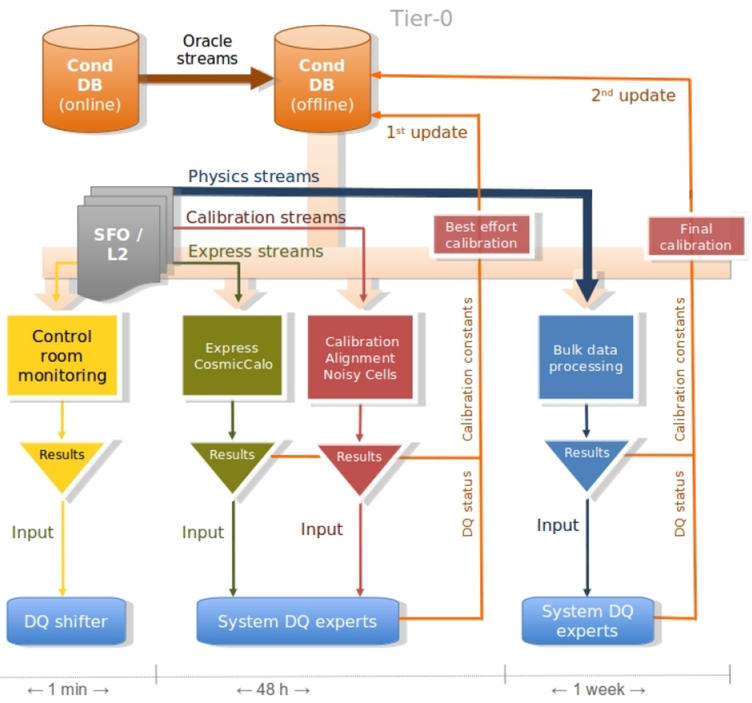
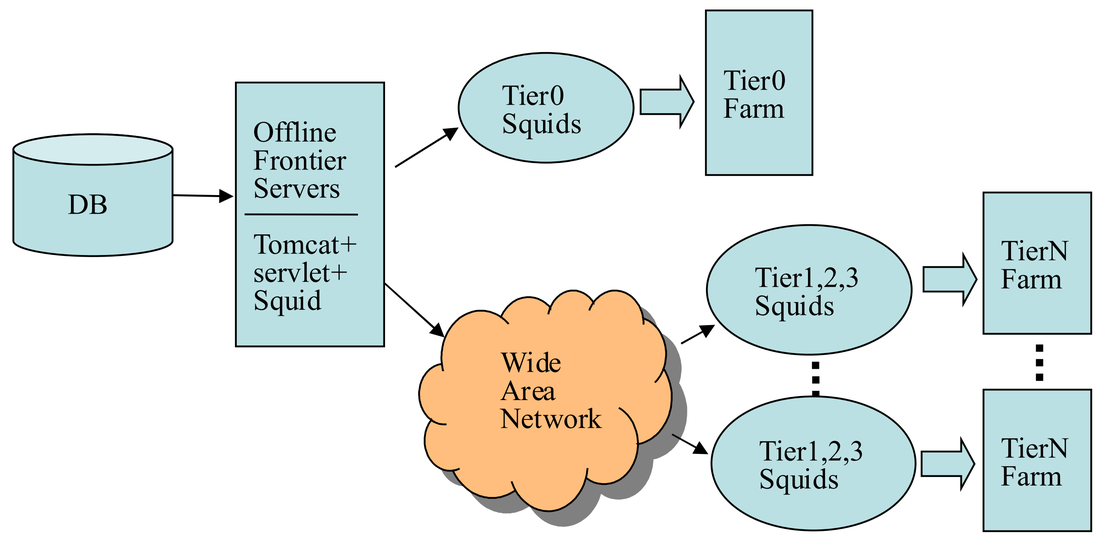
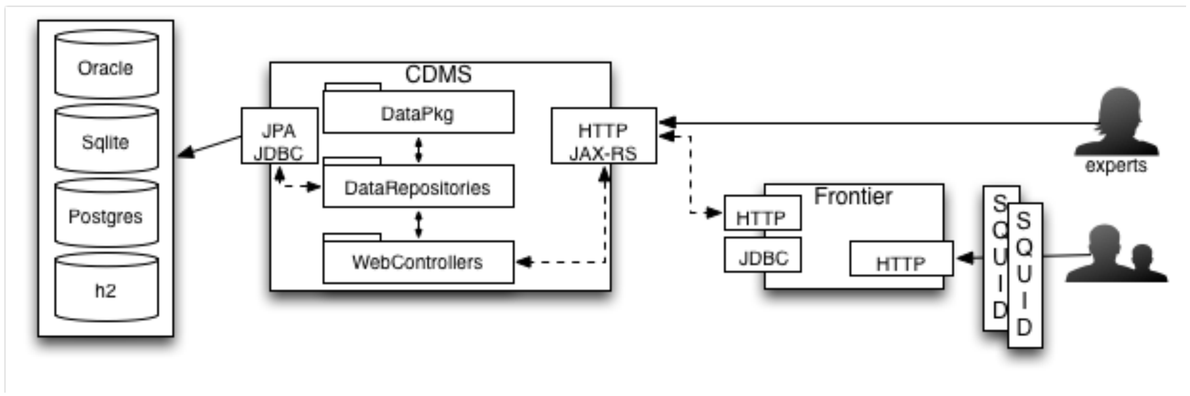
Conditions @ REST

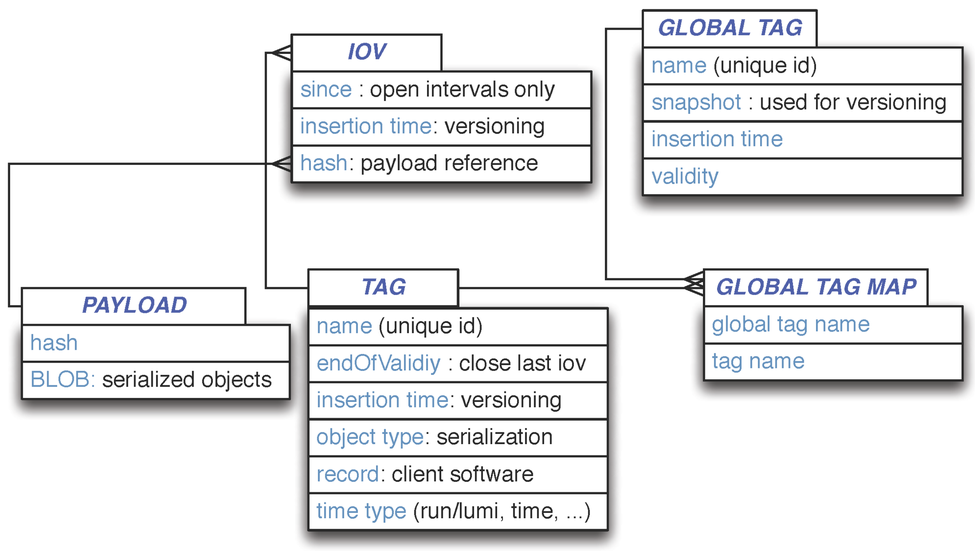




- Conditions Data are non-event data needed for optimal processing of event data
- Examples include calibrations, dead channel maps, subsets of configurations and more
- Typically stored in a relational DB
- Typically starting with subsystem-specific calibrations and hence often heterogeneous
- COOL (LCG Conditions DB) provides a multi-faceted interface to these data for experiments

- COOL is by design not cache-friendly
- Payloads are not uniquely addressed
- Problem for large-scale computing
- Frontier addressed the problem very well, providing a scalable caching solution built on RESTful interfaces
- Solution successfully tackles Run 2 challenges of LHC (and other) experiments
- Can we share more conditions data management solutions?





- The HSF Conditions DB working group converged on a management model for conditions data
- Exposing simple RESTful interfaces to clients, building on good experience with Frontier
- Client-server Java applications using industry-standard components

- Data Model uses relational DB to achieve cacheable queries by design, payloads uniquely identified by a hash
- Global tag to organise different subsystems with little or no time-correlation to conditions
- Git and filesystem approaches also work for simpler data

- Crest is a cross-experiment initiative
- Built on years of experience
- Crest is the only* conditions data management system that is pro-cache by design!
- When will ***your*** experiment start using Crest ?

* This statement may be an alternative fact