towards an abstract system for massive data processing

Jan Engels

EUDET Annual Meeting 2009
University of Geneva, 19th October 2009

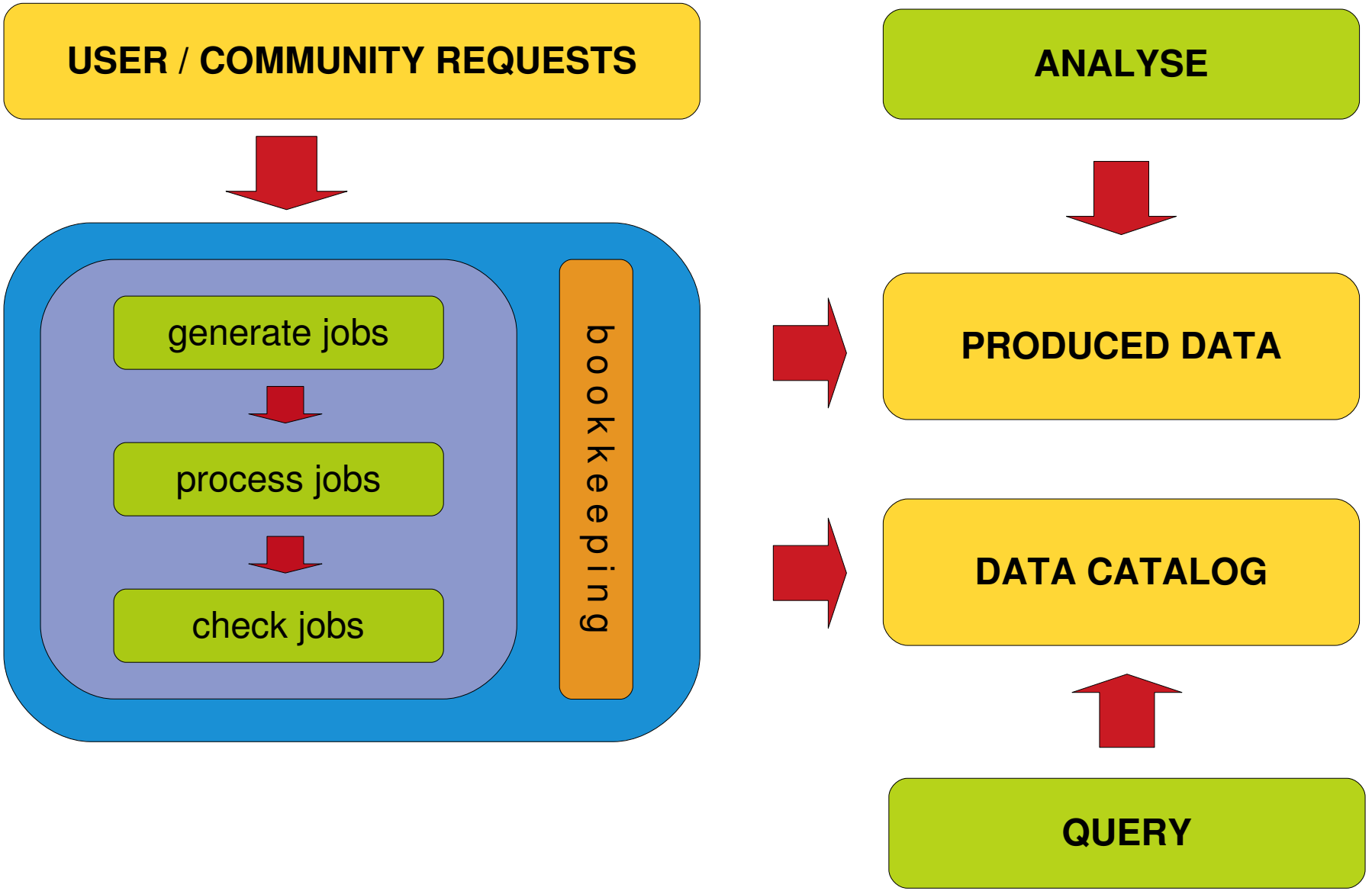
Outline

- Motivation
- System overview and properties
- Data model
- Implementation for ILD MC Production
- Outlook & Summary

Motivation

- Experience from last mass production for the ILD LOI
  - ~60 million events simulated and reconstructed
  - More than ½ million grid jobs recorded in central database
  - Software used for running last production was not initially designed to scale for such dimensions, result:
  - 2 FTE's working for 3 months
- No tool available which provides an abstract modular/multi-core designed application for massive data processing
- Looked into tools from other experiments
  - Most of them very specific to the experiment
  - Ganga showed to be a good tool to have a unified interface for job submission
  - Some ideas adopted from other experiments (CMS RefDB)

System overview



System properties

- LOTS OF DATA
  - Overview of all data is not really possible for conventional human beings
  - Data must somehow be categorized
  - Existence of a catalog is crucial
  - Special care must be taken for designing the data model

- LOTS OF JOBS
  - Production system running 24h/day - 7days/week
  - Avoid interruptions at any costs
  - In case of interruptions data integrity must be ensured

Chain of data processing

1. User requests Data

2. Production coordinator / operator

3. Data processing

4. User analyses Data

Database as the bookkeeping system

- Key constraints ensure data consistency
  - Duplicate entries immediately detected
  - Not possible to delete referenced data by mistake
- Additional check constraints (e.g. triggers)
  - Check type and range of fields
  - normalize paths ...
- Normalize data
  - Errors only need to be fixed in one place
  - Better performance
  - Need good backup system!!

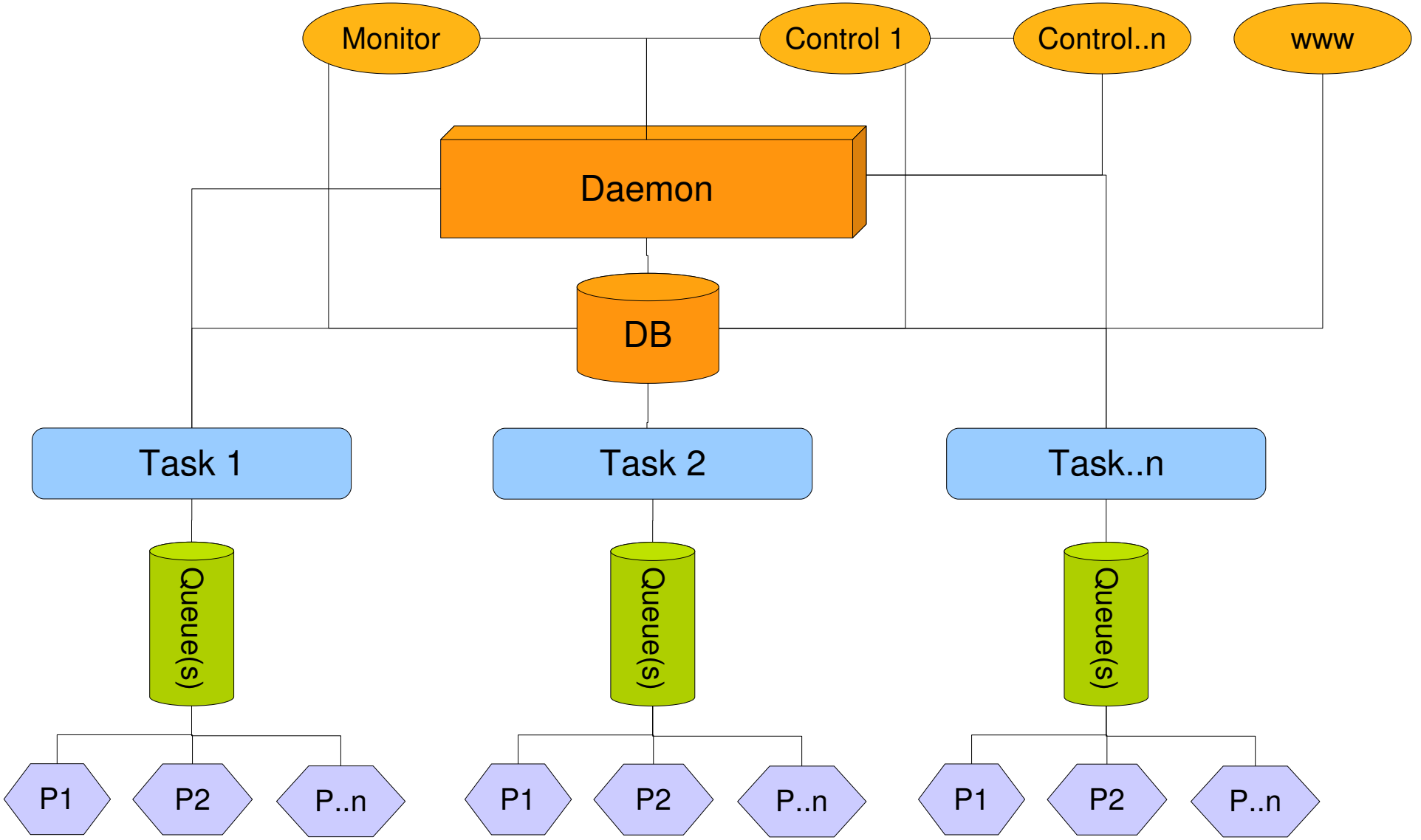
Database as the bookkeeping system

- Let the DB do what the DB can do!
  - Critical operations executed atomically in a single place (Increase reliability)
  - Less checking on DB client applications
  - Client applications can be easily ported to other languages/operating systems
  - No network overhead / bandwidth limitations
  - Fast!!!
- Views for creating an abstraction layer to the DB
  - Change underlying DB model without afecting client interfaces
- Transaction support
  - Rollback changes in case of error

Production system

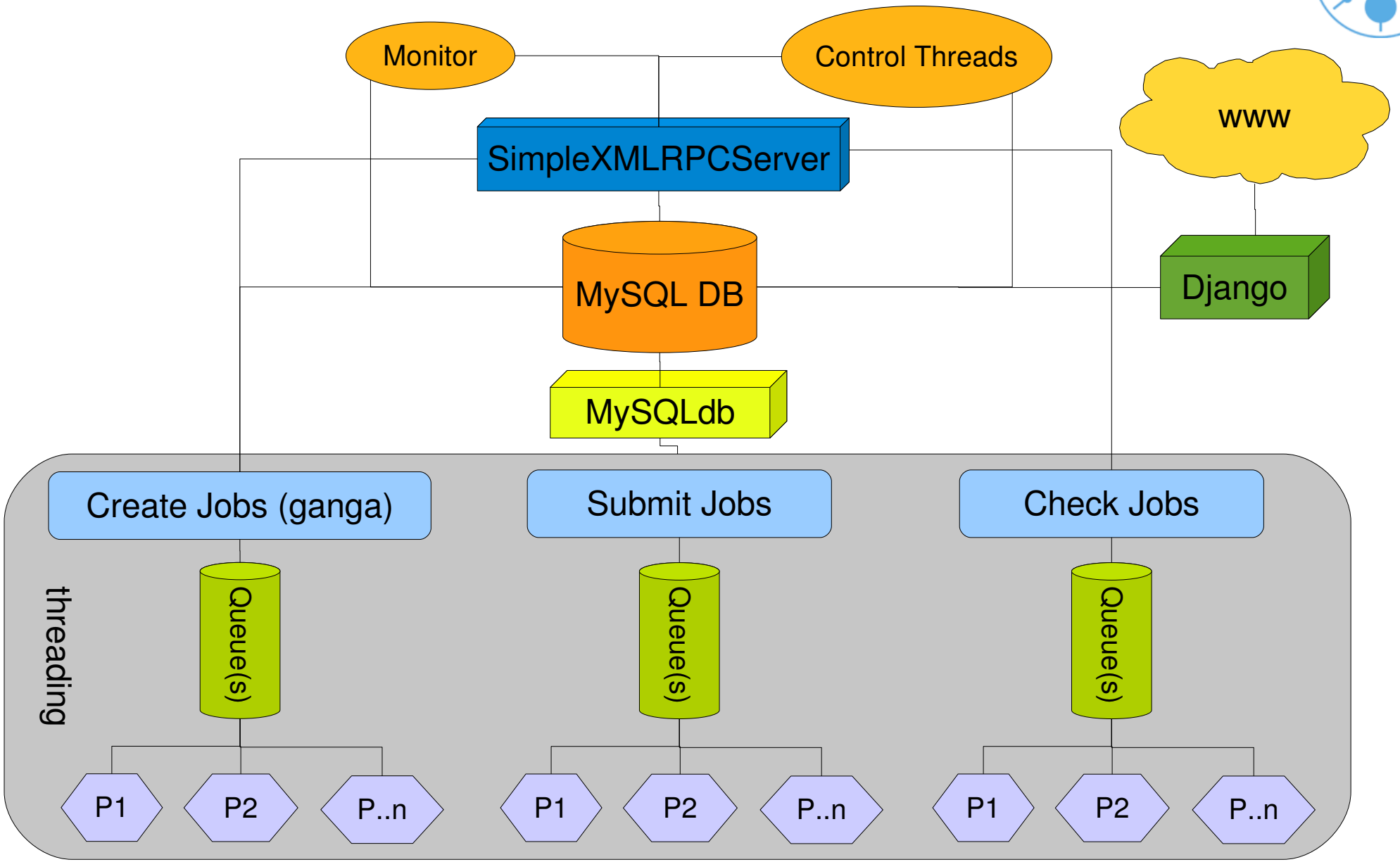
- Execute in parallel whenever possible
  - Multi-core or even cluster environment
- Reliability
  - Errors always happen!!
  - Minimize data corruption caused by errors
  - Error detection
  - Error handling
- Traceability of errors
  - Proper logging
  - Choosing what to log is not trivial
  - Too noisy vs. too silent
- Modularity
  - Modular → Extendable
- Dynamic / flexible
  - Dynamically change settings @ runtime

Possible layout for a production system

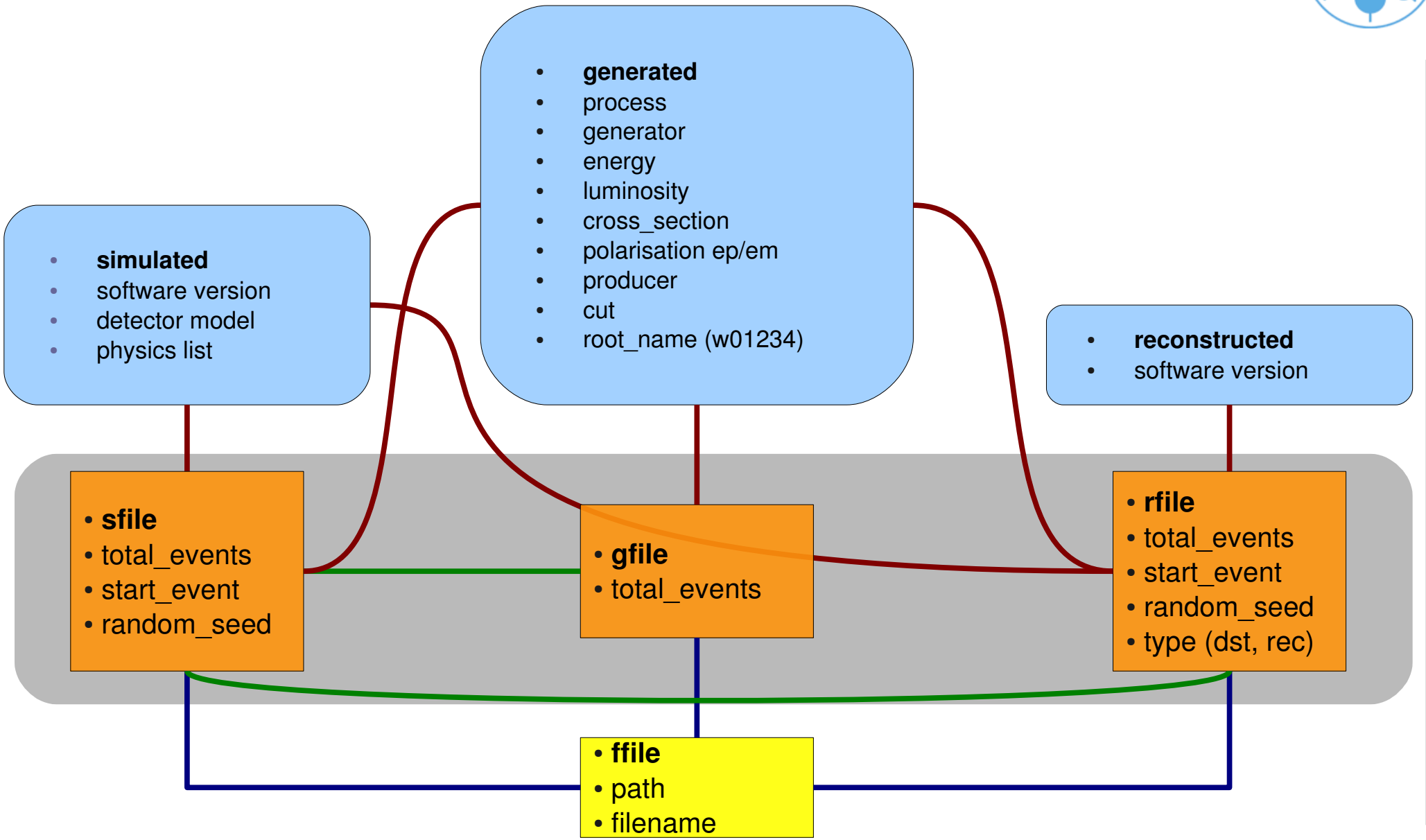


IMPLEMENTATION FOR THE ILD MONTE CARLO PRODUCTION SYSTEM

ILD MC Production system



ILD MC Production System (Database Model)



ILD MC Production System (Database Model)

- MySQL 5
  - Support for stored routines, triggers, views
- Key constraints
  - InnoDB engine
- Additional constraints
  - Using triggers
  - Normalize paths
- Transactional support
  - InnoDB engine (master DB)
  - Rollback in case of error :)
- Backup system
  - Master → Slave (synchronized frequently)
  - Hourly / Daily / Weekly backups
  - Monthly backups (and on-demand snapshots) written to tape

ILD MC Production System (Database Model)

- Normalized data
  - Splitting of file / job meta-data
  - (More than one file and/or job can share the same attributes)
- Abstraction layer between user and data model
  - Database model can be extended/updated without affecting client interfaces
  - Implemented using views and stored routines
- Methods for manually inserting new records into the DB
- Methods for generating jobs from user requests
- Statistic information used for job monitoring / system control
- Views defined to easily find related information all together

ILD MC Production System

- Implemented in python (v2.4)

- Designed from scratch for multi-core environments
  - Possibility to extend for running on a small cluster
- Reliable
  - Proper shutdown methods
- Minimize data corruption
  - In case of errors production coordinator/operators are notified
- Traceability of errors
  - Proper logging by using the standard python logging library

ILD MC Production System

- Modular
  - Abstracted backend for job processing by using ganga
  - http://ganga.web.cern.ch/ganga/
- Dynamic
  - Settings dynamically changeable @ run-time
- Universal www Interface
  - Provided by django http://www.djangoproject.com/
  - admin interface and user authentication for free

Summary and Outlook

- Experience from last monte carlo production
  - Made us aware of critical points of failure
  - Performance and scalability are important!!
  - Too many files in one directory → Grid commands too slow!!
  - File and directory naming conventions are important!

- Outlook
  - New system for massive data processing available very soon
  - Run small production next month
  - Monitoring (request tracker)
  - Less man power needed for running production :)
  - Extend system to bookkeep and manage grid software installations

Thank you! Your feedback is welcome!