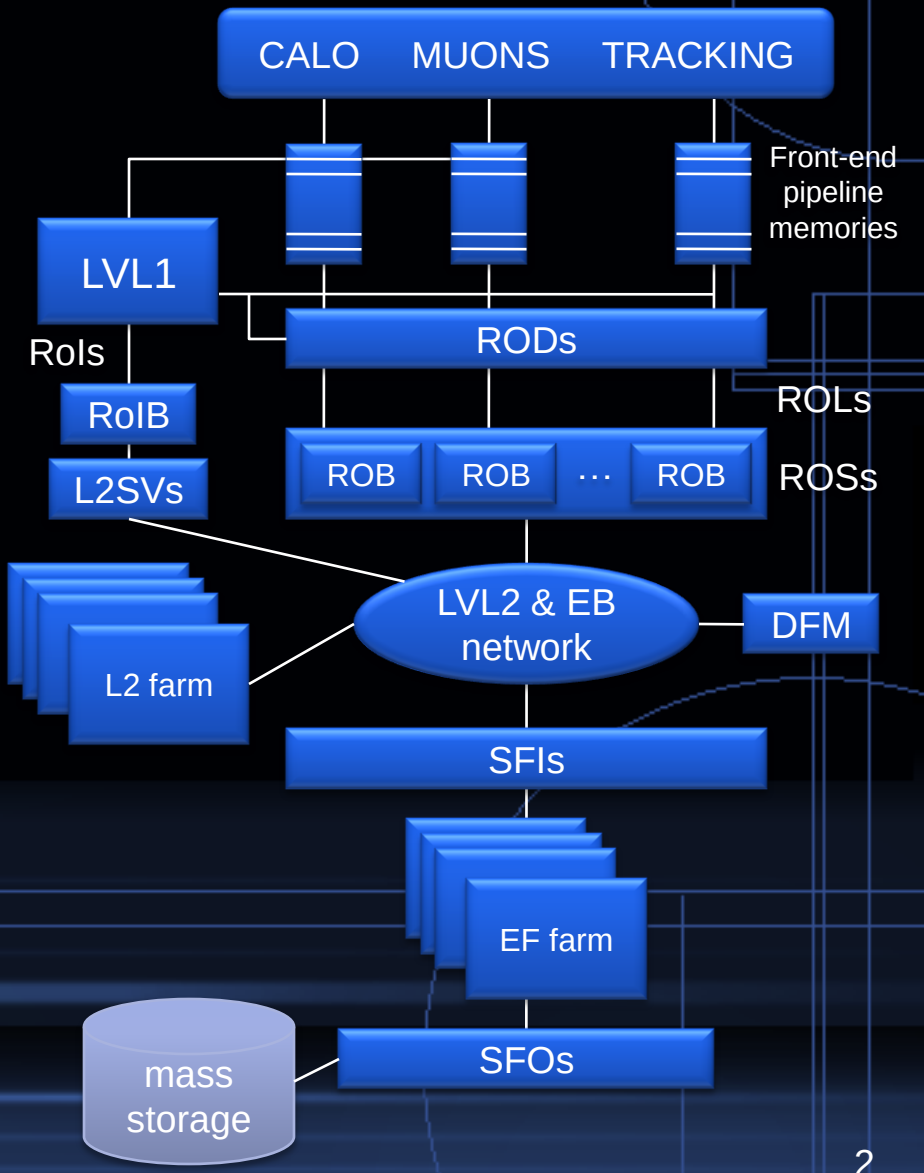
The ATLAS DAQ System Online Configurations Database Service Challenge

- Introduction
- Challenging requirements
- Implementation
- Status and Conclusions

J. Almeida, M. Dobson, A. Kazarov, G. Lehmann-Miotto,
J.E. Sloper, I. Soloviev and R. Torres

The Configuration Service

- Is a part of the ATLAS DAQ/HLT
- Provides configurations prepared by many DAQ, trigger and detector groups for:
  - overall DAQ system description (control, monitoring, data-flow)
  - partial trigger and detectors descriptions
- The configurations are accessed by huge number of online processes (up to 25K) during boot and config phases of run
- Once used, the configurations are archived by the service for offline access and analysis by experts



The Challenges (1/3)
the service has to ...

- support **data model** able to describe extendable complex interconnected data
  - to speak on a common language and to re-use the same code, the core DAQ DB schema is used across all ATLAS systems and detectors (extendable to describe their specific properties)
- provide **generation of data access libraries** (DAL)
  - a DAL maps DB schema on types of programming languages, instantiates their objects from DB data and integrates user-defined algorithms
  - available for programming languages (C++, Java and Python) used by ATLAS software
  - no need to learn complex database API
  - often schema changes requires DAL auto generation (hard and error prone to update DALs manually)

The Challenges (2/3)
the service has to …

- provide **automatic generation** of databases:
  - descriptions for software releases and hardware
  - configurations for tests of ATLAS systems and data-taking on ATLAS Point-1 and various test-beds
- have **scalable architecture**:
  - up to 25 thousands clients accessing service simultaneously at boot and config stages
  - performance of configuration has significant impact on the ATLAS experiment downtime
  - partial reconfiguration during run
  - caching retrieved information on client's side is essential

The Challenges (3/3)
the service has to …

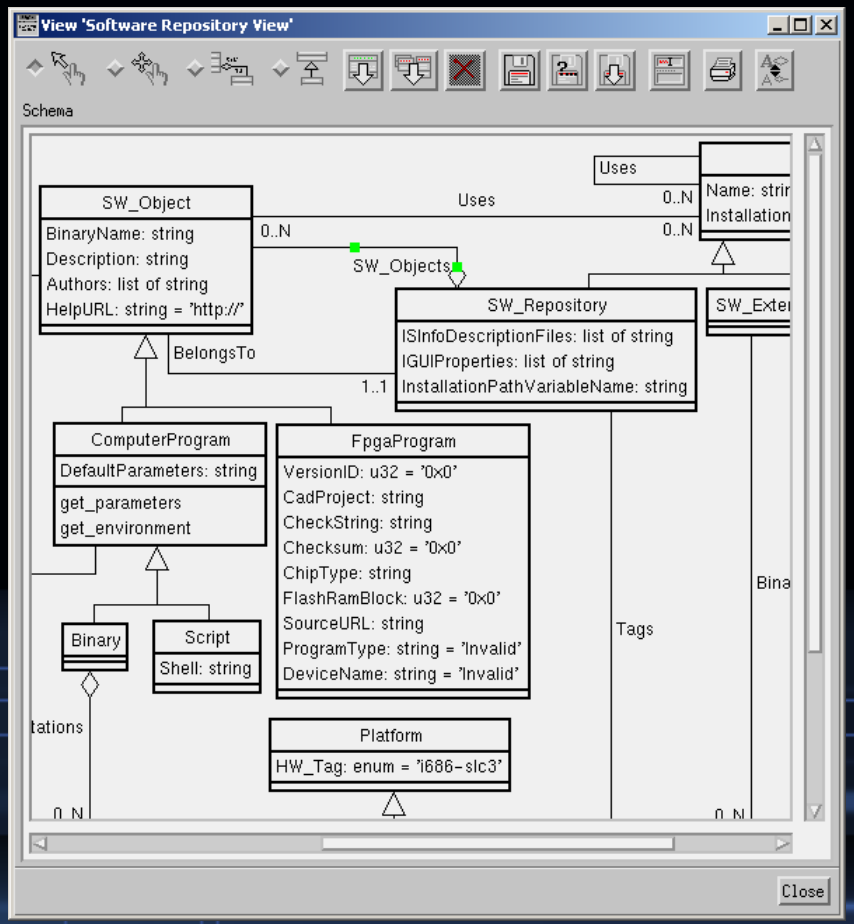
- automatically **archive used configurations** for later access by offline code and analysis by experts
  - guaranty the data are not removed, edited or corrupted
- be **easily accessible** and **installable**
  - the most users are not database experts
  - the TDAQ releases were installing by tens of ATLAS collaborating institutes around the world
    - many of them cannot use installation from CERN and do not have knowledgeable DBMS administrators
    - need simple procedures to install the configuration service

The Implementation

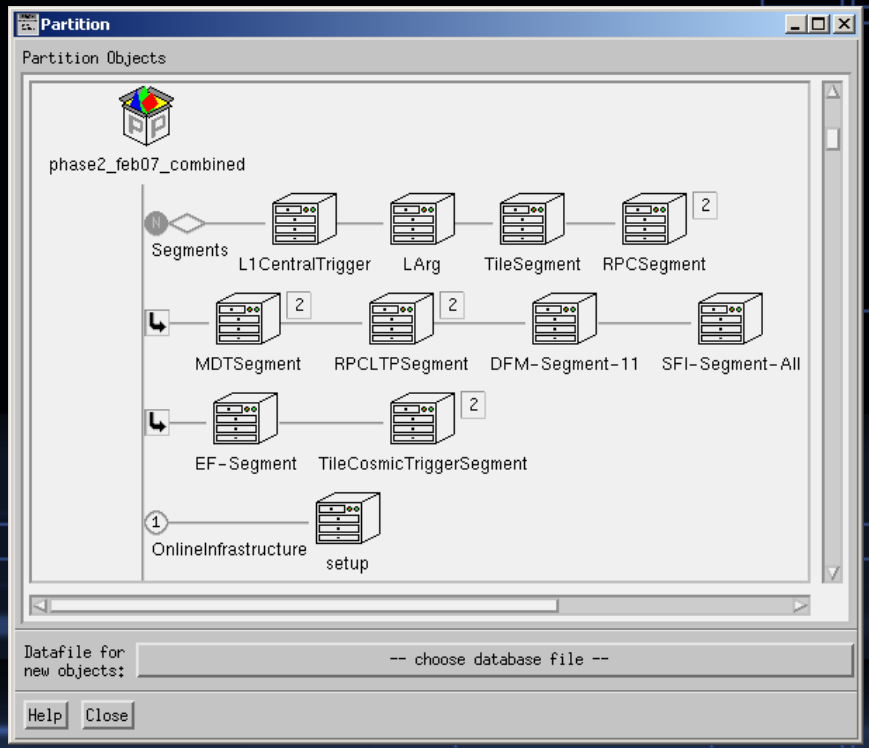
- We evaluated many DBMS and persistent managers as candidates for the service implementation, but no one satisfying all requirements were found
- As a prototype, we started to use own OKS persistent object manager and later made it capable of fulfilling DAQ configuration service needs
  - from beginning the OKS is based on object data model, uses XML files as persistent storage of the schema and data, and provides GUI tools for schema design and data modifications (see next slide for examples)
  - later we added:
    - remote access (RDB – remote DB server using CORBA)
    - archiving into relational database
    - abstract API layer with plug-ins for different implementations and DAL generation tools

The OKS GUI Editors

The OKS Schema editor allows to design schema using UML syntax

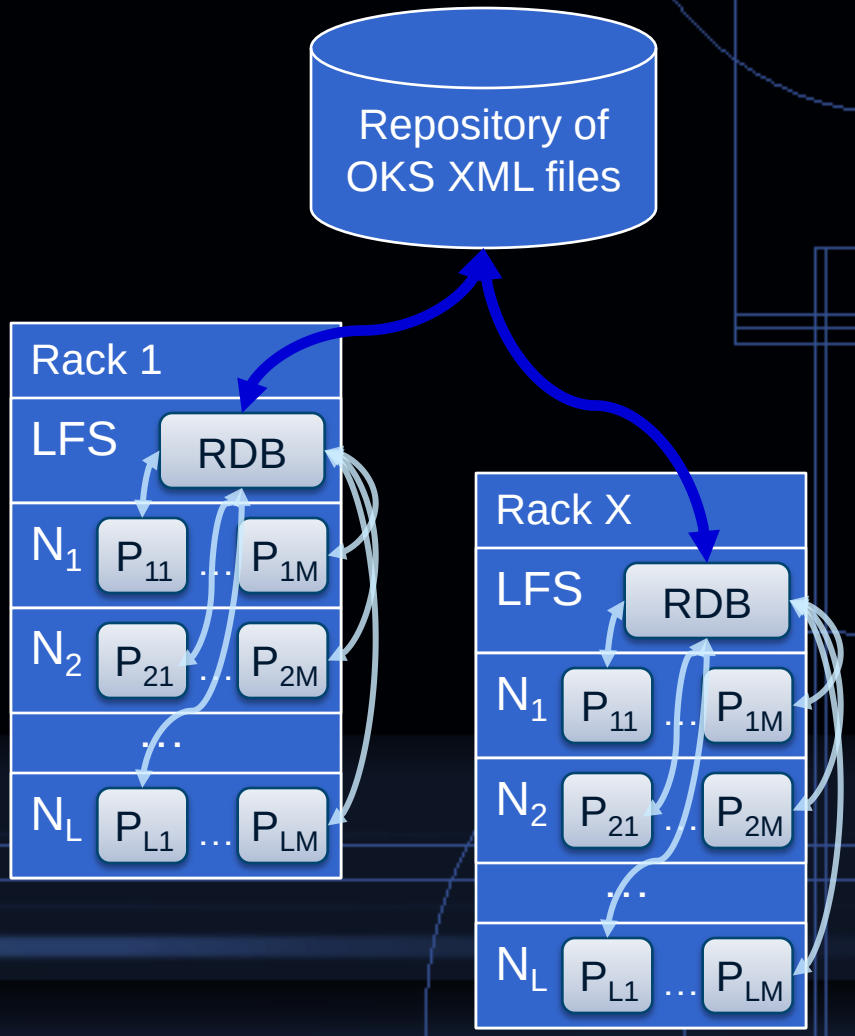


The OKS Data editor allows to create and to modify data using customized appearance of user-defined views graphically presenting relations between objects



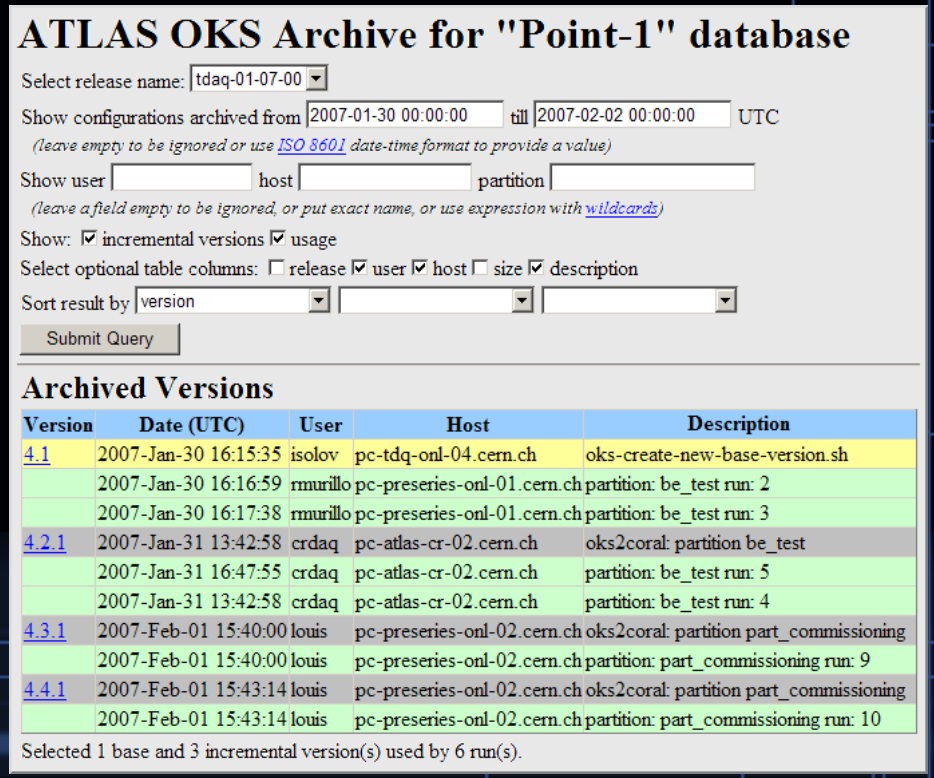
Remote Access Performance & Scalability

- The distant access to OKS is implemented by RDB server on top of CORBA
- RDB preloads configuration from master copy of configuration data
- All programs running on a rack @ Point-1 require similar configuration data and are accessing the same RDB server running on rack's local file server
- RDB caches results of queries
- During last technical run 500 L2 applications got full config from single RDB server in ~5"



Archiving

- At start of each new run the configuration service automatically archives the used configuration into OKS relational archive
- The data can be read from the archive using OKS API and can be extracted as XML files
- The Web GUI allows to select archived data by release name, time interval, user, host and partition masks, to compare, and to save
- The OKS supports incremental archiving, when only differences between given and base versions are really stored



ATLAS OKS Archive for "Point-1" database

Select release name: tdaq-01-07-00

Show configurations archived from 2007-01-30 00:00:00 till 2007-02-02 00:00:00 UTC

(leave empty to be ignored or use ISO 8601 date-time format to provide a value)

Show user host partition

(leave a field empty to be ignored, or put exact name, or use expression with wildcards)

Show: ☑ incremental versions ☑ usage

Select optional table columns: ☐ release ☑ user ☑ host ☐ size ☑ description

Sort result by version

Submit Query

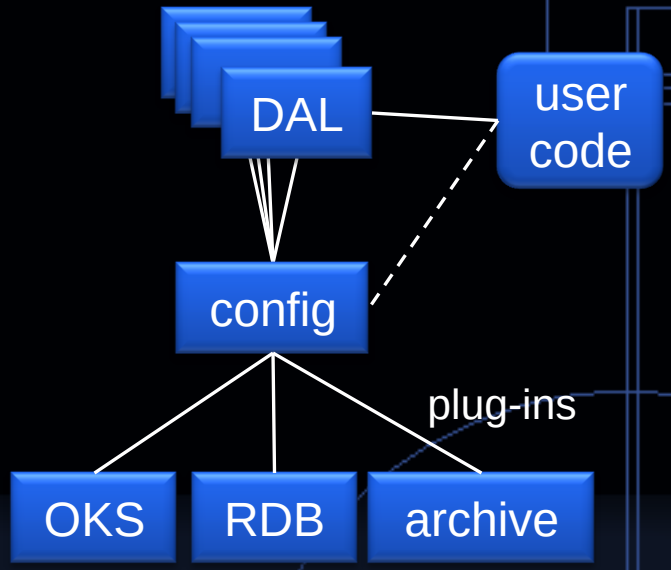
Archived Versions

Version	Date (UTC)	User	Host	Description
4.1	2007-Jan-30 16:15:35	isolov	pc-tdq-onl-04.cern.ch	oks-create-new-base-version.sh
	2007-Jan-30 16:16:59	rmurillo	pc-preseries-onl-01.cern.ch	partition: be_test run: 2
	2007-Jan-30 16:17:38	rmurillo	pc-preseries-onl-01.cern.ch	partition: be_test run: 3
4.2.1	2007-Jan-31 13:42:58	crdaq	pc-atlas-cr-02.cern.ch	oks2coral: partition be_test
	2007-Jan-31 16:47:55	crdaq	pc-atlas-cr-02.cern.ch	partition: be_test run: 5
	2007-Jan-31 13:42:58	crdaq	pc-atlas-cr-02.cern.ch	partition: be_test run: 4
4.3.1	2007-Feb-01 15:40:00	louis	pc-preseries-onl-02.cern.ch	oks2coral: partition part_commissioning
	2007-Feb-01 15:40:00	louis	pc-preseries-onl-02.cern.ch	partition: part_commissioning run: 9
4.4.1	2007-Feb-01 15:43:14	louis	pc-preseries-onl-02.cern.ch	oks2coral: partition part_commissioning
	2007-Feb-01 15:43:14	louis	pc-preseries-onl-02.cern.ch	partition: part_commissioning run: 10

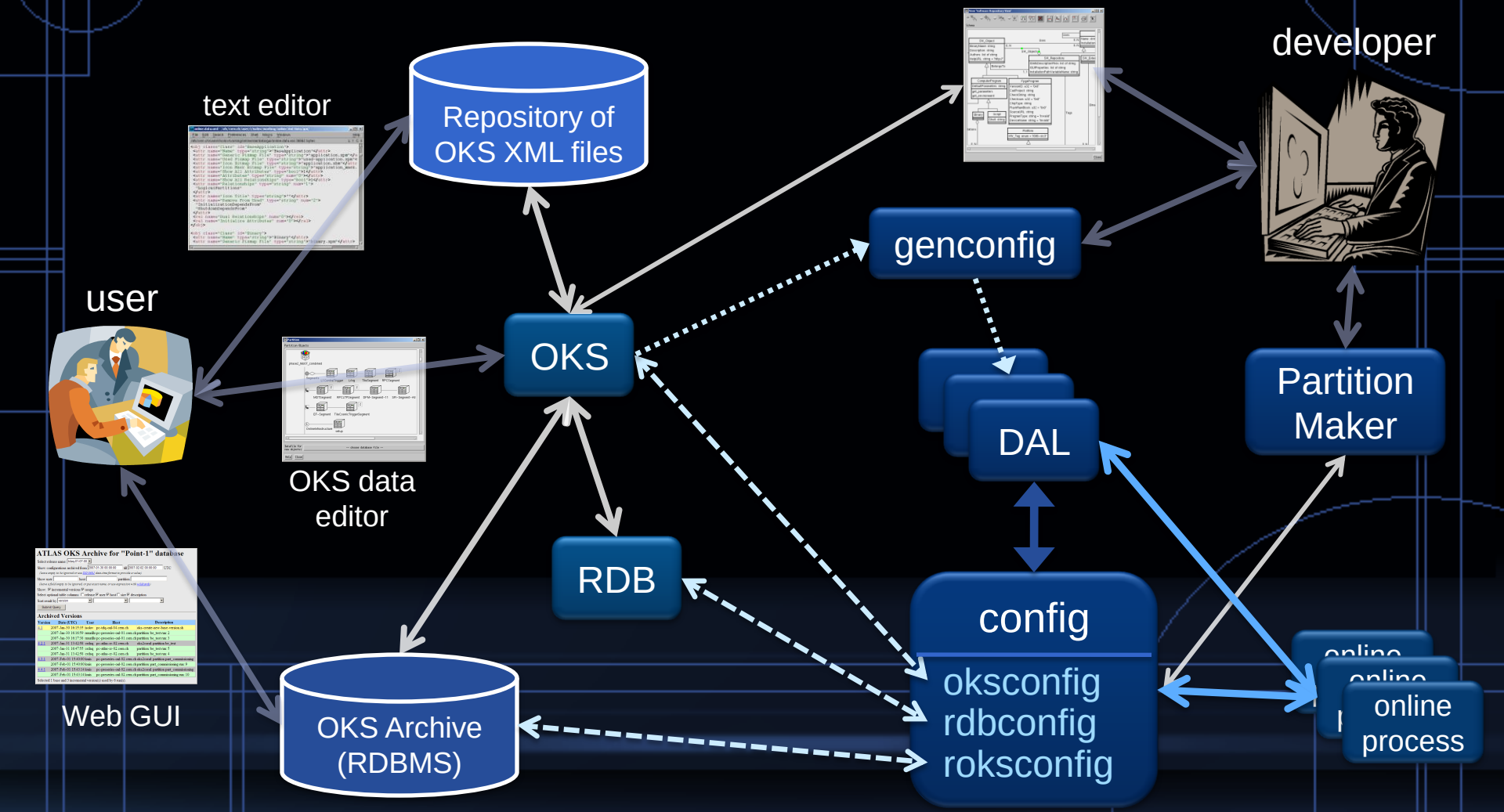
Selected 1 base and 3 incremental version(s) used by 6 run(s).

Programming Interfaces

- The code of ATLAS software never uses OKS programming interfaces directly. There are two layers available to hide them:
  - the config layer defines abstract interface to work with arbitrary databases and configuration objects
    - implements caching of results on client side
    - exists for C++, Java and Python
    - implementations are available for XML, RDB and archive
  - the DAL layer is using above config layer



Configuration Service: Interfaces and Users



The Current Status

- The present implementation is used several years and has been tested during:
  - ATLAS combined test beams
  - ATLAS TDAQ technical and detector commissioning runs
  - TDAQ large scale performance and scalability tests
- Each time an important feedback was received, we had modified the configuration service software to address user's needs
  - full code is under our control from the very base levels
- For the moment there are no any major design, reliability and performance problems
- The configuration service is mostly ready for ATLAS

Should We Use Anything Else?

- An object database can be a better choice (at least requires less development from our side), but we failed to find scalable and reliable one ☹
- Relational database are often used for similar purposes:
  - prevalence and maturity in database world
  - commercial and freeware solutions
  - promising area for specialists (not only in science sectors)
- But relational technology itself does not address any challenging requirement we have
  - the installation and support at collaborating institutes is difficult
  - pure relational data model does not support inheritance we need and some useful data types, like arrays which we are using
  - one has to write the DALs; automatic generation is difficult because of gap between required data model and the limited relational one
  - to deal with simultaneous requests from thousands of clients one has to provide tools to cache at some level results of queries (similar to RDB)
  - archiving of relational data is not trivial; to prepare next configuration one cannot simply modify existing data since the historical data can be overwritten; versioning of relational data is not trivial, since a copying of single data can be not enough because of relations from other data

Conclusions

- The huge size and complex organization of the ATLAS system puts challenging requirements on the DAQ system configuration service
- There is no any third-party solution satisfying all of them is found
- Current implementation is based on the software designed and developed by the DAQ group and addresses all requirements
- The configuration service is mostly ready for ATLAS run in the beginning of 2008