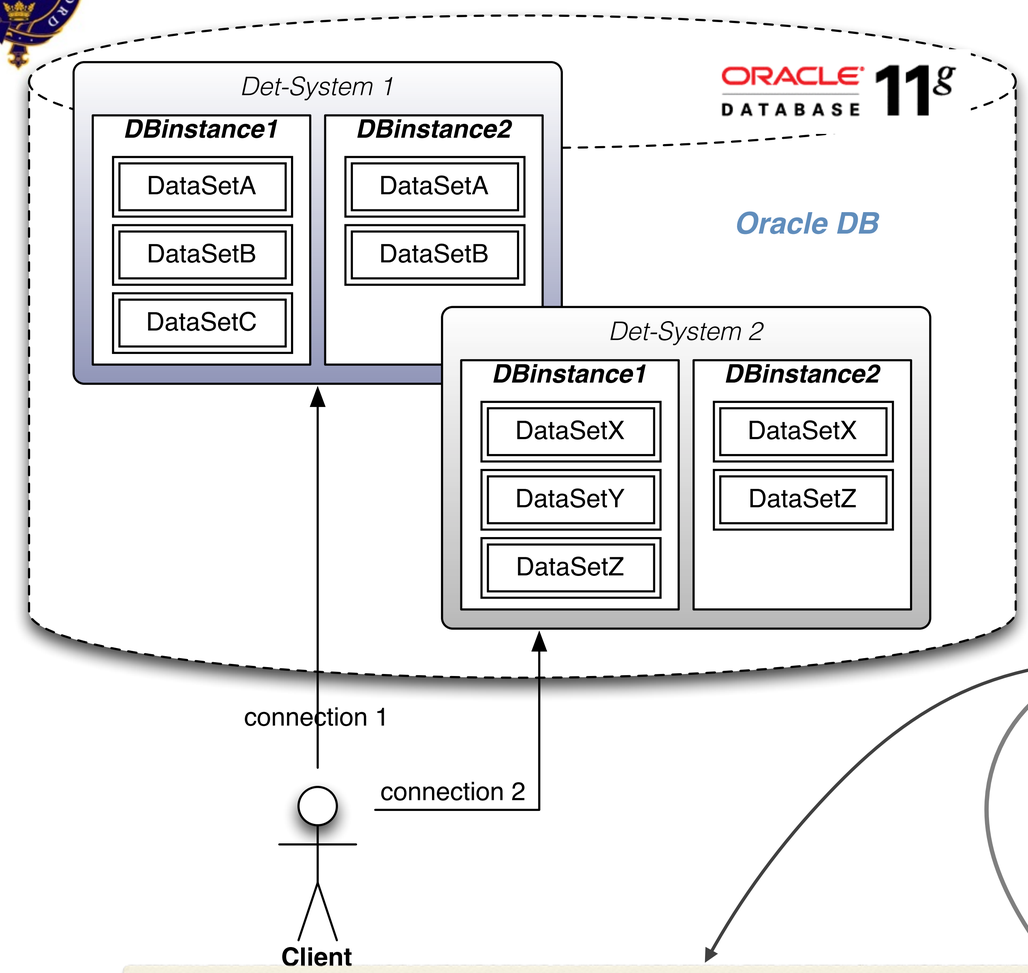
A JEE RESTful service to access Conditions Data in ATLAS

A.Formica[1], E.Gallas[2] on behalf of ATLAS collaboration
1: CEA/Saclay IRFU (SEDI)
2: Department of Physics, University of Oxford



ATLAS Conditions Data Infrastructure

Oracle : systems store their conditions data in separate schemas inside an Oracle DB administered by CERN IT

LCG-COOL : The C++ based API for interacting with an LCG Conditions Database was developed by CERN IT. Its methods are also available via a python bindings package PyCool. Interactions from any client are on a schema-specific basis.

Work-Flow : system expert connect to its own schema/database instance to access the appropriate tables and manage its conditions

Conditions Metadata in COOL

The conditions data are accessed by reconstruction programs by means of a set of metadata:
Global Tag : a string identifying a set of leaf tags (this string is the same in every Det-System/DBinstance)
Node : a string identifying a DataSet type
Tag : a string identifying a set of IOVs (Interval Of Validity) for a given DataSet type
IOVs : time related parameters identifying the validity in time for a given payload inside a DataSet type

Drawbacks

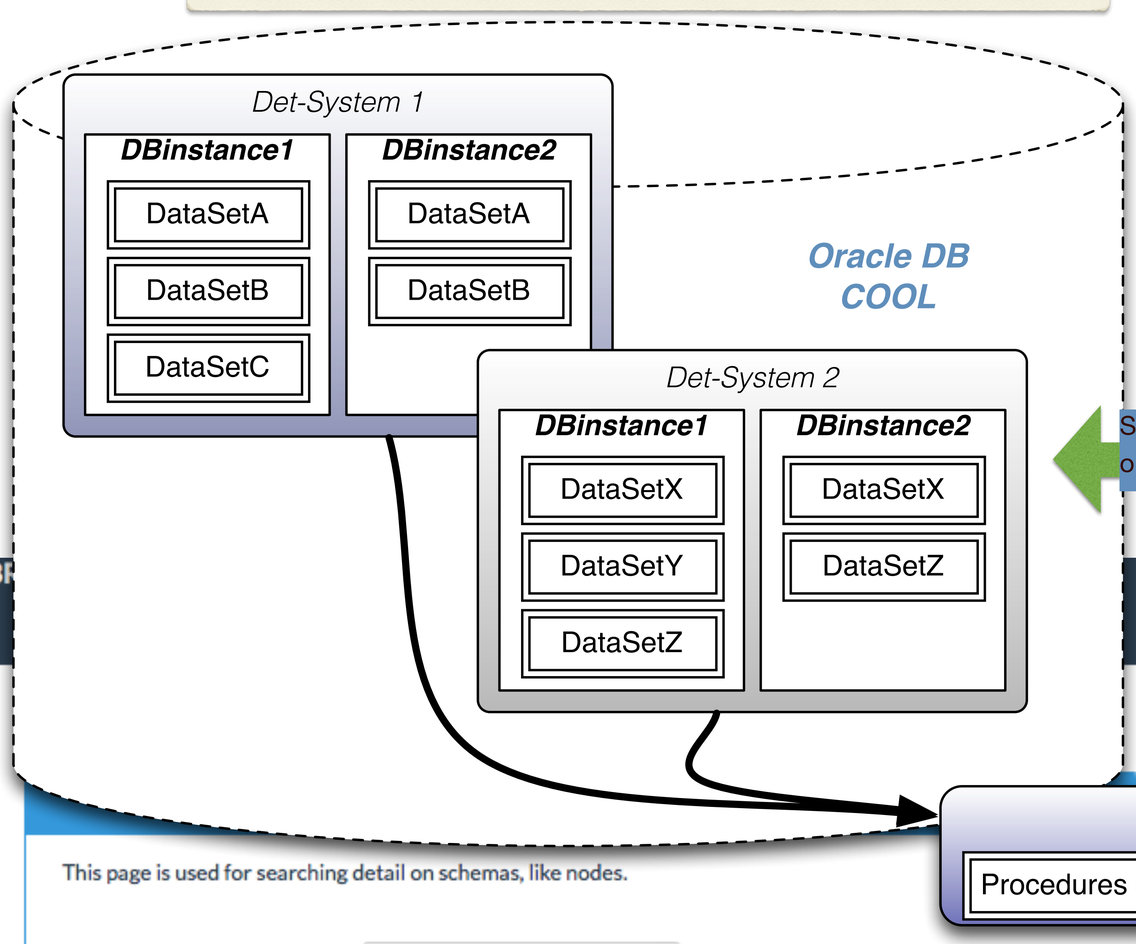
Any conditions or metadata retrieval are possible only via the COOL API, so retrieving information from several schemas requires an equivalent number of independent connections to the database

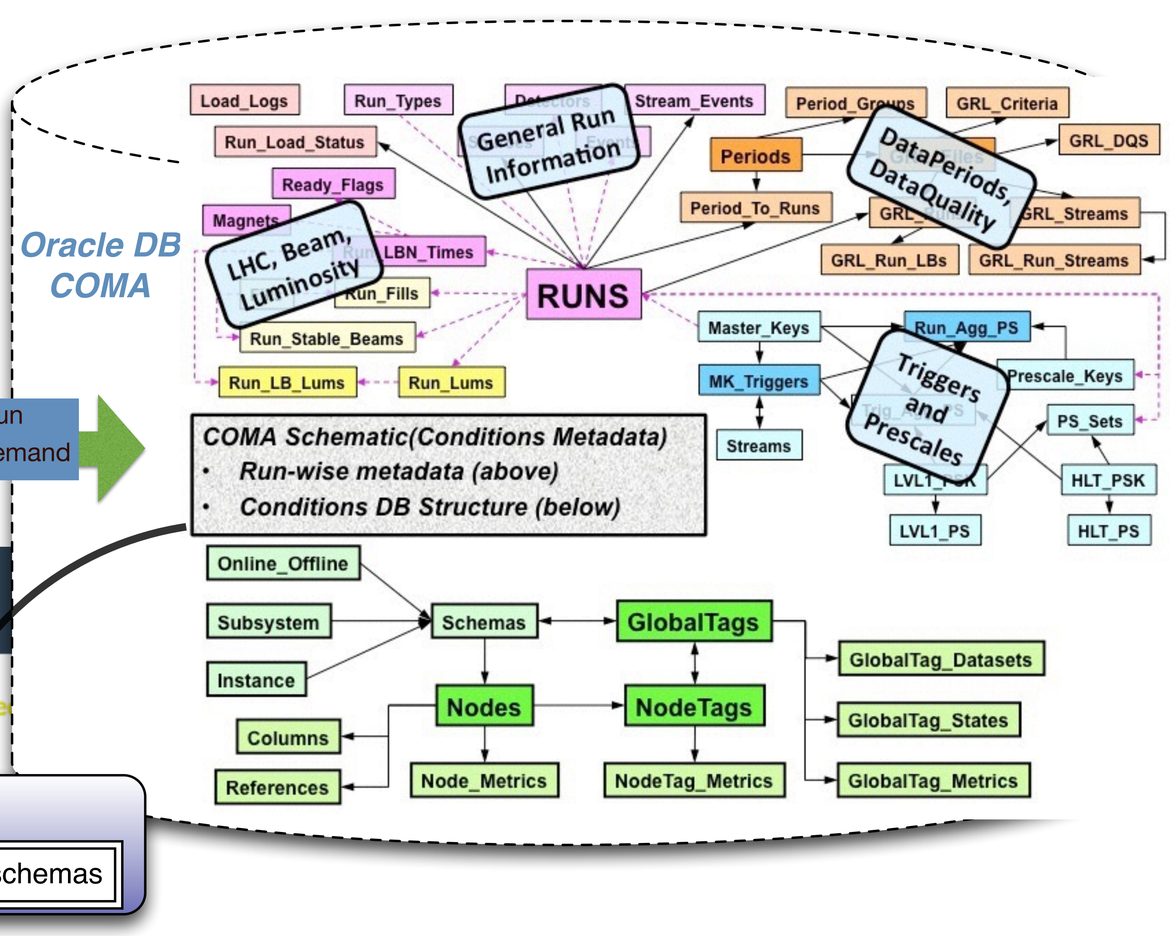
Simplify COOL schema access

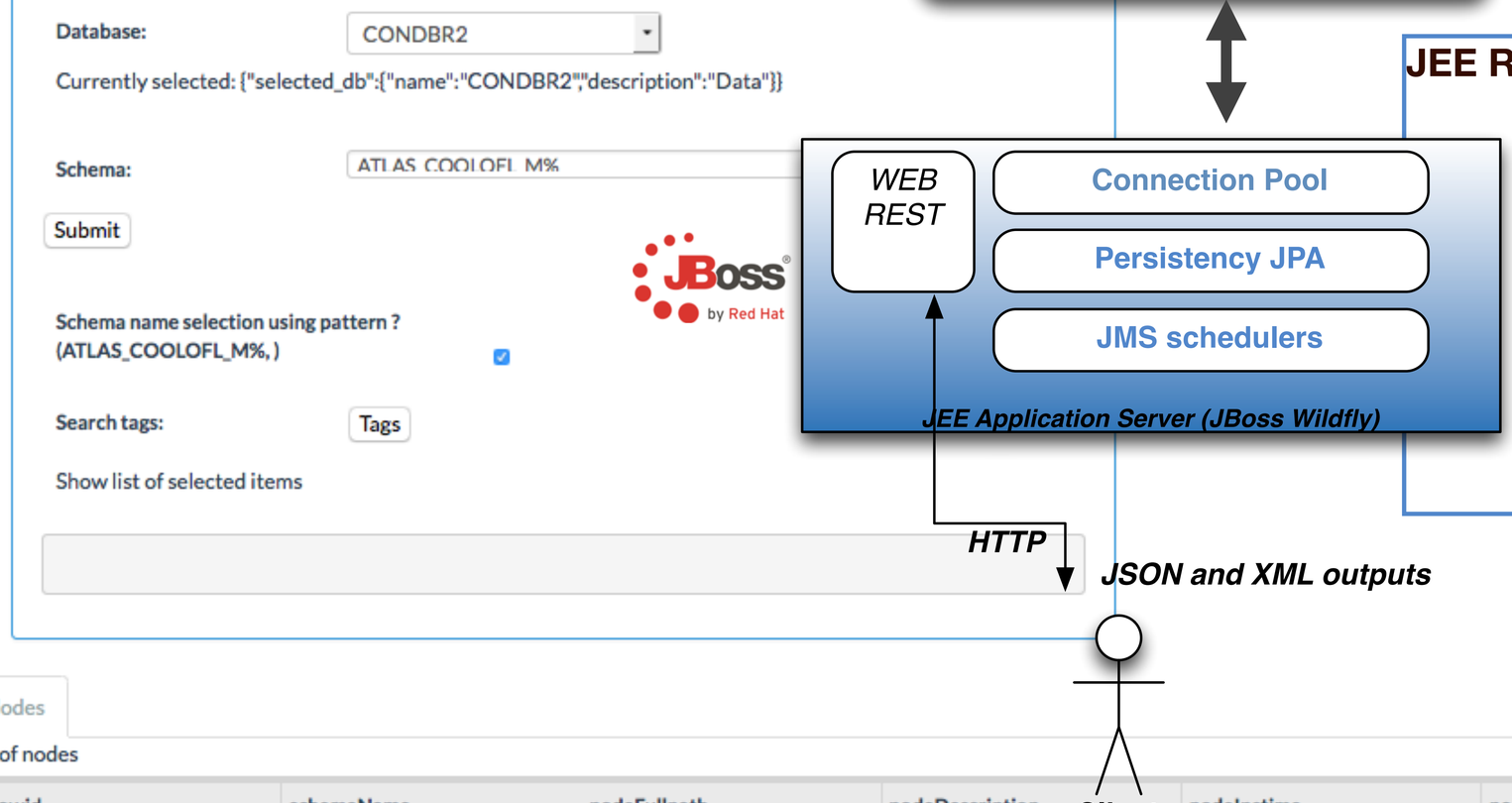
Usage of single schema with select privileges on all COOL schemas
Data retrieved via PL/SQL (loop over all schemas implemented at that level)

META-DATA extraction in unique set of tables (COMA DB)

Conditions meta-data (with the exception of IOVs and payload data) have been extracted and merged into an independent well normalised relational schema: The COMA Database includes all ATLAS Conditions Database Node and Tag metadata (along with other operations metadata) which provides information from several schemas (and systems) in the same query.







JEE RESTful services

1) Access to multiple data sources.
2) Web services implemented via REST ((REpresentational State Transfer)
   - ✤ provide client language independent services
3) PL/SQL performance faster than COOL API methods
   - ✤ provides tools for quick comparison of COMA and COOL content
   - ✤ enables faster synchronisation from COOL to COMA
4) Provides methods to clients not available via the COOL API
5) Monitoring functionalities implemented as scheduled tasks
   - ✤ IOV coverage for system tags
   - ✤ Detect missing Condition information in run ranges
   - ✤ Compute total number of IOVs per tag

Nodes

List of nodes

rowid	schemaName	nodeFullpath	nodeDescription	nodeInstime	nodeIovBas
1	ATLAS_COOLOFL_MUONALIGN	/MUONALIGN/MDT/BARREL	<timeStamp>time</timeStamp><ad...	2014-08-11_13:44:15.162017000 ...	time
2	ATLAS_COOLOFL_MUONALIGN	/MU			
3	ATLAS_COOLOFL_MUONALIGN	/MU			
4	ATLAS_COOLOFL_MUONALIGN	/MU			
5	ATLAS_COOLOFL_MUONALIGN	/MU			
6	ATLAS_COOLOFL_MUONALIGN	/MU			
7	ATLAS_COOLOFL_MUONALIGN	/MU			
8	ATLAS_COOLOFL_MDT	/MDT/RTBLOB	<timeStamp>run-lumi</timeStamp>...	2014-08-11_13:22:47.350940000 ...	run-lumi
9	ATLAS_COOLOFL_MDT	/MDT/T0BLOB	<timeStamp>run-lumi</timeStamp>...	2014-08-11_13:23:28.187909000 ...	run-lumi

Multi language support

Client can be implemented in any language because the data are exchanged in ascii (JSON or XML formats supported). We have implemented a simple **python client** and a **javascript** based WEB application as examples (using AngularJS). Access also possible via *curl*.