

Condition/configuration db

Demonstration using real example
of the ATLAS TGC cond DB

Conclusion

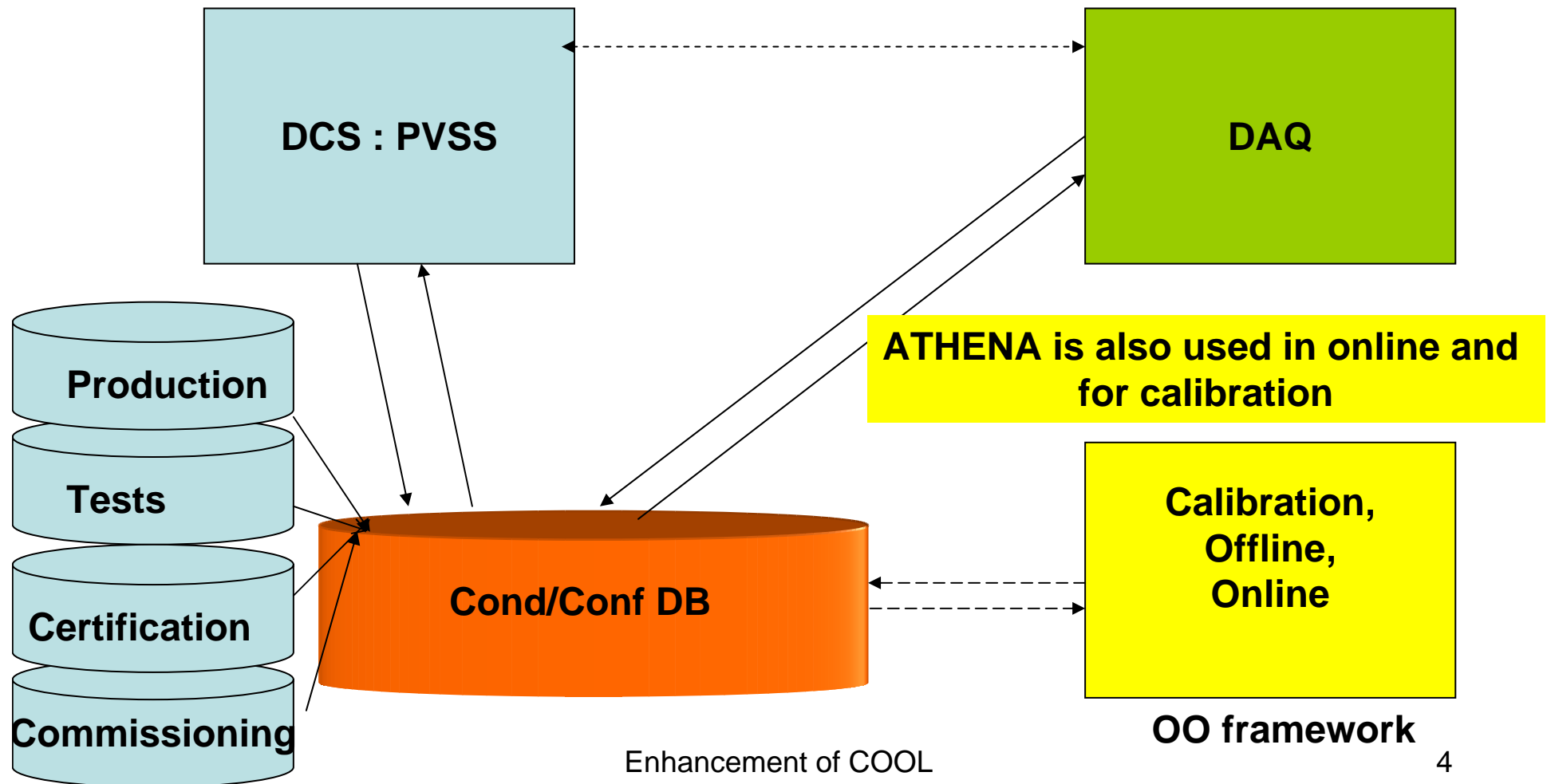
- **Requirements from COOL:**
 1. **Allow us to store non-COOL tables in the same DB**
 - These tables will generally be static and/or have small data volume (Detector description and hierarchies as well as Configuration information)
 - We will be able to do joins between our tables and COOL tables
 - This will enable using the static tables in ATHENA in diagnostics
 2. **Have our tables maintained together with the COOL tables**
 3. **Encode some payload fields as foreign keys**
 - This will enable quick extract, e.g. for HW configuration

DB Goals

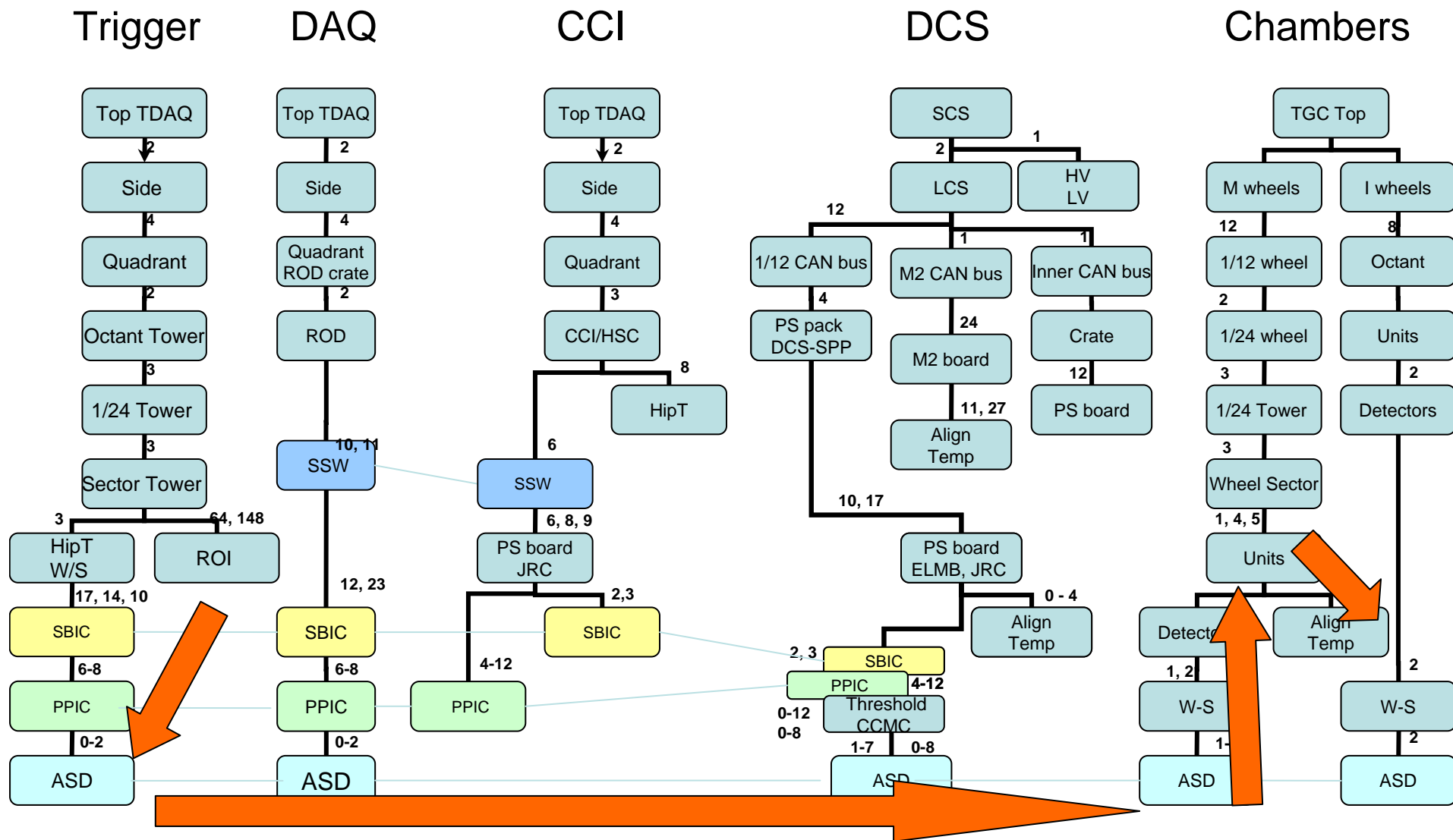
- Record the time varying (HW / SW) non event data : ==
Conditions
- Detector **Configuration** - used at the beginning of a run
(sends about 150,000 parameters in our case)
- Keep the detector structure, various constants, wiring maps
etc..
- Store calibration parameters: delays, thresholds,...
- Store alignment parameters
- Hold the history information starting from the **production**
through the **tests**, **certification** and **commissioning**
- The same data is used for :
 - Detector operation, maintenance and diagnostics
 - Analysis, Calibration, Trigger simulation

Db client/server applications

Requirements from condition database



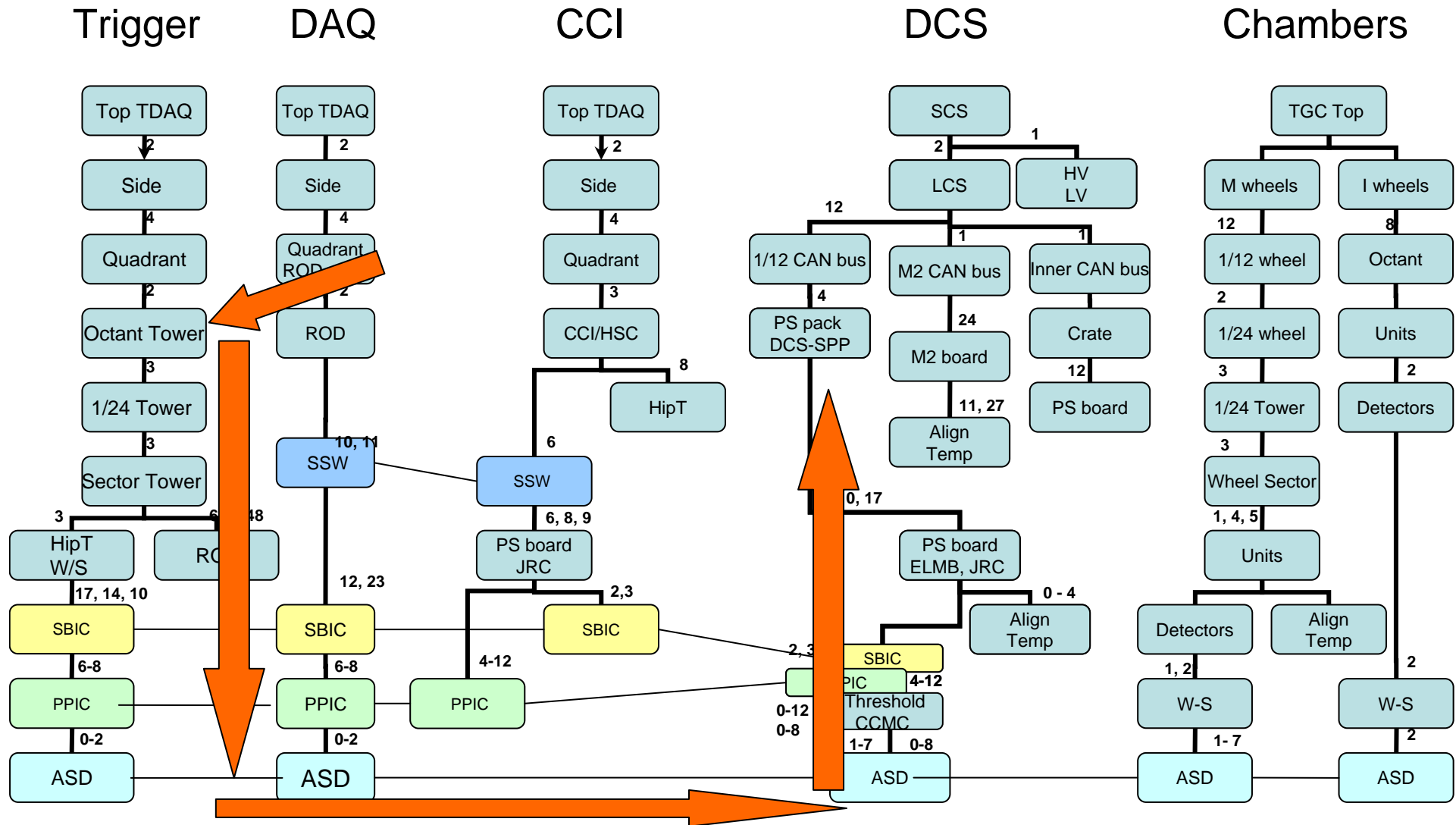
Each path divides the detector differently (Layers, towers, 1/8th, 1/12th,....)
 Example 1: search for all the ROI with a temperature > 30 degrees



7/9/2005

Enhancement of COOL
<http://atlas-proj-tgc.web.cern.ch/atlas-proj-tgc/doc/CondConfDBCOOL.pdf>

Example: need to change the threshold (via the ps pack) for a given octant

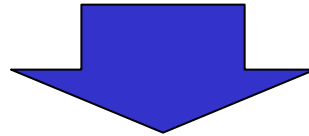


7/9/2005

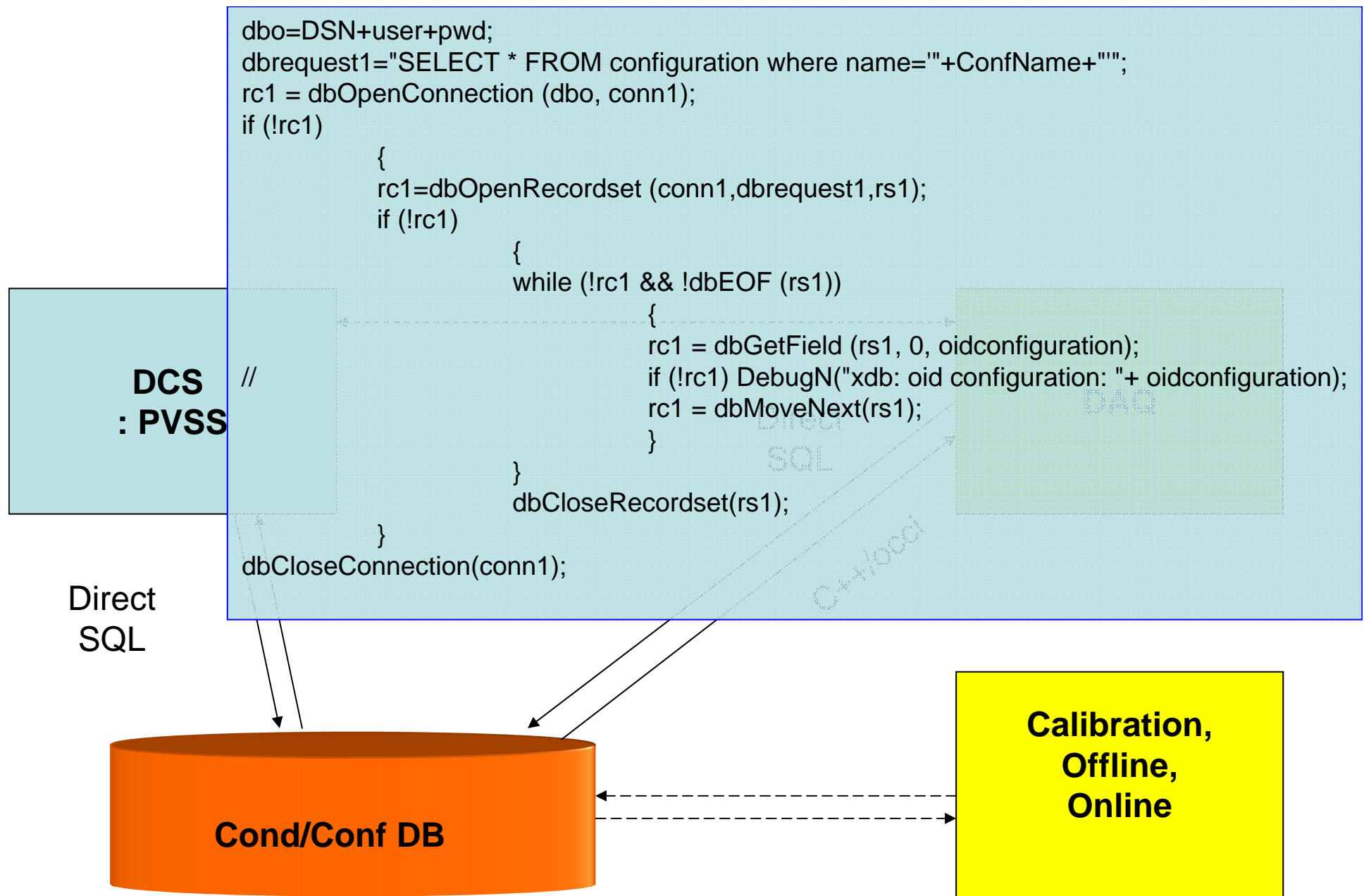
Enhancement of COOL
<http://atlas-proj-tgc.web.cern.ch/atlas-proj-tgc/doc/CondConfDBCOOL.pdf>

DB access methods

- DCS : read/write directly SQL queries in the cond/conf db. The DCS which knows the exact structure of the detector gives the guideline of the structure of the db
- DAQ : read/write directly SQL queries in the cond/conf db
- (missing) OO/Offline/Online/calibration : need a general interface to avoid very specific user/code to directly reach the db



Use of COOL tables and API is currently limited and difficult because some features are missing in COOL to solve the complexity of the problem:
Example: List channel-IDs of chambers whose radiation dose in the last machine-test was > X.



Possibility to declare and insert foreign key inside the tables

Conditions Table

Channel

Payload

IoV

•Possibility to query time, channel id (metadata) but also value (payload)

7/9/2005

configuration	
oid	
name	
rdate	
type	
comment	
ins_date	
ins_time	

confddb	
oid	
conf_id	
prod_id	
arg_id	
value	
ins_date	
ins_time	

Configuration Table

conddb	
oid	
conf_id	
prod_id	
arg_id	
value	
iov_since	
iov_until	
ins_date	
ins_time	

product	
oid	
device_id	
dcname	
label	
cciname	

each parameter (channel) needs his own record frequency in case of problem

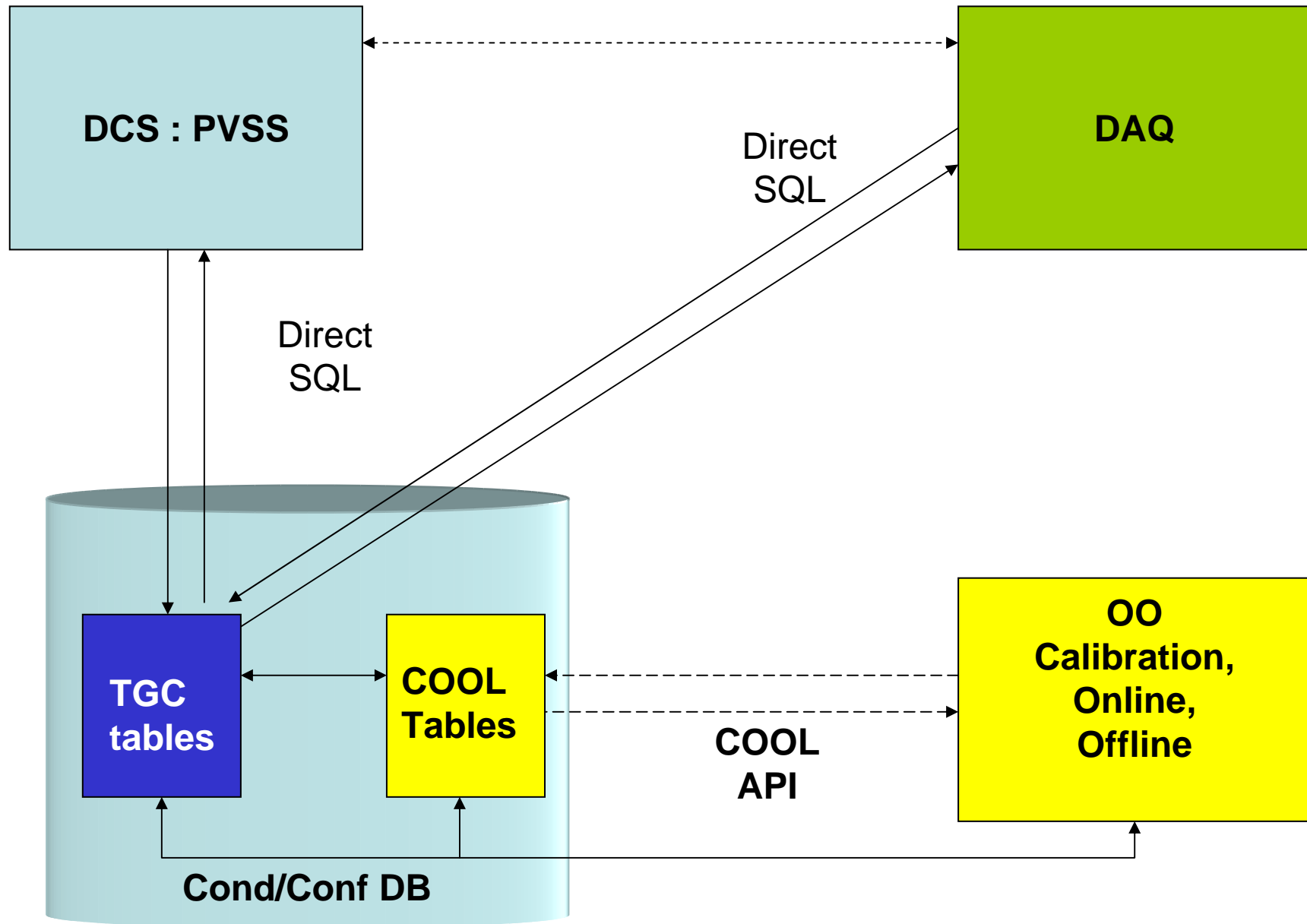
argument	
oid	
device_id	
arg_order	
type	
name	
comment	
pvss_cast	

Hierarchy Table

hierarchy	
oid	
prod_id	
in_prod_id	
hierarchy	

Channel

device	
oid	
name	
comments	



Conclusion

- **Requirements from COOL:**
 1. **Allow us to store non-COOL tables in the same DB**
 - These tables will generally be static and/or have small data volume (Detector description and hierarchies as well as Configuration information)
 - We will be able to do joins between our tables and COOL tables
 - This will enable using the static tables in ATHENA in diagnostics
 2. **Have our tables maintained together with the COOL tables**
 3. **Encode some payload fields as foreign keys**
 - This will enable quick extract, e.g. for HW configuration
- **Otherwise we are forced to use private DB outside COOL**