Collecting Conditions usage metadata to optimize current and future ATLAS software and processing

D Barberis, A Formica, E Gallas, S Oda, L Rinaldi, G Rybkin, M Verducci on behalf of ATLAS Collaboration
Motivation and Outline

GOAL: improve understanding of Conditions data usage by event-wise processing
→ This helps us to develop improved tools and resources for future processing

• Introducing the ATLAS Conditions DataBase
• The need of gathering metadata information
• The ATLAS DB Release use case
  – Collecting Conditions DB metrics
  – Collecting Conditions DB access patterns by specific jobs
  – Realization of a custom-lite DB Release
'Conditions' data encompass a wide variety of information

- characterize the state of all ATLAS subsystems during specific intervals
- are not generally stored event-wise but during an interval of validity
- essential for data taking and/or event processing
- are grouped by logical or physical subsystem
DataBase Design

Driven primarily by the LCG-COOL DB structure

• Folder centric: Folders represent Conditions DB tables
  – Each Folder is owned by a specific Schema
    • Each has subsystem, instance, and if used offline or strictly online
  • Multi-version Folders have one/more FolderTags
    – For Conditions that allow different versions over time intervals
  • FolderTags may be included in one/more GlobalTags
    – When designated to be used in event-wise processing
Conditions metadata and usage information are spread over a large number of “sources”

- COOL DB: 17K tables distributed in ~30 schemas in Oracle
- GlobalTags, used to identify a set of conditions for every (sub)system, not exhaustive because information regarding single-version folder usage is not associated to any global tag
- Conditions may appear to be needed but then never really used by software

We have developed two new ways to gather metadata information from CondDB

- **CondDB metrics**: extract more information at DB level which are today not stored (added new tables containing number of IOVs, payload size, ...)
- **CondDB access patterns**: analyze log files for reprocessing or MC production

Some specific tasks for investigation:

- **DB Releases**: use case discussed in this contribution
- **Overlay Event simulation**: (heavy load on squid-frontier due to access to large number of Conditions)
The ATLAS DB Release Today

- ATLAS DB Releases contain:
  - file-based static SQLite replicas of the Conditions, Geometry and Trigger databases.
  - POOL ROOT payload data files for the Conditions DB (PoolCond)

- DB Releases used by the installations of ATLAS s/w when a network connection is not available on the grid node executing the job (HPC farms, offline laptops).

- DB Releases contains Conditions and Trigger data for Monte Carlo jobs only

- New DB Releases are created when a new GlobalTag is ready

- DB Releases are available on CVMFS and DDM
The present system allows to create only incremental versions of the ATLAS DB Releases.

DB Release size is steadily increasing (up to 9GB).

Not all information stored in a DB Release is needed.

**GOAL:** produce a custom-lite DB Release containing only information needed for dedicated production.
New Columns added in existing tables
- Date of latest changes
  - Helps with metadata sync
- Row counts, volumes
  - Helps to understand how much data is in each Folder and in each FolderTag
    → Useful for understanding relative scale of Athena access
- Count unique external references
  - Helps to know the number of POOL files are associated with
    - Each Folder
    - Each FolderTag
    → how many POOL files to in a customized DB Release
New schemas added: extension to Athena Log Conditions

Log files from specific jobs, retrieved from our distributed computing infrastructure, are analyzed in order to gather specific usage of conditions directly from the software side.
Several tests performed by running typical simulation transformations, accessing the default configuration

A postExec command has been used to dump all the Folders/Tags loaded during the job, by analyzing the Athena log files

**Result:** we observed that only a fraction of Folder/Tag relations were used (ranging from 10% to 20%)

This allowed to reduce the SQLite file size to a 10% - 20% of the original size

Tests with custom-lite DB Releases were successful

Ongoing analysis:

- Add information gathered by the CondDB metrics
- Reduce the number of PoolCond files
- Set up an automatic procedure
Summary and outlook

Information contained in the ATLAS Conditions DataBase is spread over a large number of sources

- Collecting such metadata is challenging in some very specific tasks
- The DB Release use case has been studied
- Custom-lite DB Releases have been successfully created by collecting metadata information with new tools based on the analysis of the CondDB access pattern
  - Results may be improved by using CondDB metrics

Future plans

- improve the procedures for metadata collection
- set up an automated procedure for DB Release slimming
- Investigate more use cases:
  - application on the Overlay Event simulation
Backup
New Columns added in existing tables

- **CB_Owner_Instances**
  - These are the schemas

- **CB_Nodes**
  - These are the folders
    - Branches (nodes) also here

- **CB_Ftags**
  - These are folder tags

- **New data is from**
  - Frontier cache consistency dates
    - Date of latest change to the folder
  - COOL Oracle Sequence tables
    - Date of latest IOV or TAG insert
  - COOL MONitoring tables
    - Count of rows, size, tags, chan
      - With associated dates
      - Updated just once per day

---

### CB_Owner_Instances

- **P** – CBOL_INDEX
- **U** – OWNER_NAME
- **FU** – CBI_NAME
- **F** – CBO_NAME
- **FI** – CBS_NAME
- **U** – COOL_SCHEMA
- **CBC_DATAMC**
- **CBC_FILE_DATE**
- **COMA_INS_DATE (t)**

### CB_Nodes

- **P** – CBF_INDEX
- **FU** – CBC_INDEX
- **U** – NODE_FULLPATH
- **I** – NODE_NAME
- **NODE_ID**
- **NODE_PARENTID**
- **NODE_ISLEAF**
- **NODE_INSTIME**
- **LASTMOD_DATE**
- **CBF_LASTMOD_DATE**
- **COMA_INS_DATE (t)**

### CB_FTags

- **P** – CBFT_INDEX
- **FU** – CBF_INDEX
- **U** – TAG_NAME
- **TAG_LOCK_STATUS**
- **TAG_DESCRIPTION**
- **SYS_INSTIME**
- **CBFT_INSTIME**
- **CBFT_NODE_ID**
- **CBFT_ROWCOUNT**
- **CBFT_LASTMOD_DATE**
- **COMA_UPD_DATE (t)**

### COOLMON Tags

- **COOLMON_TAGS**
- **COOLMON_ROWS**
- **COOLMON_BYTES**
- **COMA_UPD_DATE (t)**

### COOLMON Rows

- **COMA_INS_DATE (t)**
- **COMA_UPD_DATE (t)**

---

Collecting ConditionDB metrics
• Frontier cache consistency dates
  ▪ Date of latest change to the folder
    → BEST source of when the last data was entered
    → Use this to speed up loading of folder and global tags

• COOL Oracle Sequence tables
  ▪ Date of latest IOV or TAG insert
    → only changes when a new row is added

• COOL MONitoring tables
  ▪ Count of rows, size, tags, chan
    → With associated dates
    → Updated just once per day
  ▪ cross checking info: these are derived from Oracle dictionary tables filled by Oracle scheduler jobs!
    → Should I just use the direct source instead (?)
  ▪ found COOLMON tables count IOVs but not PAYLOADS
  ▪ Accounting for data volume in dblister is confusing (outdated?)
    → Since it is missing PAYLOAD tables