Automatic database extraction for self contained and containerized Monte Carlo and analysis jobs

T. Harenberg, M. Sandhoff, M. Vogel, C. Zeitnitz

BERGISCHE
UNIVERSITÄT
WUPPERTAL

Wuppertal contributions to the Computing-Verbund

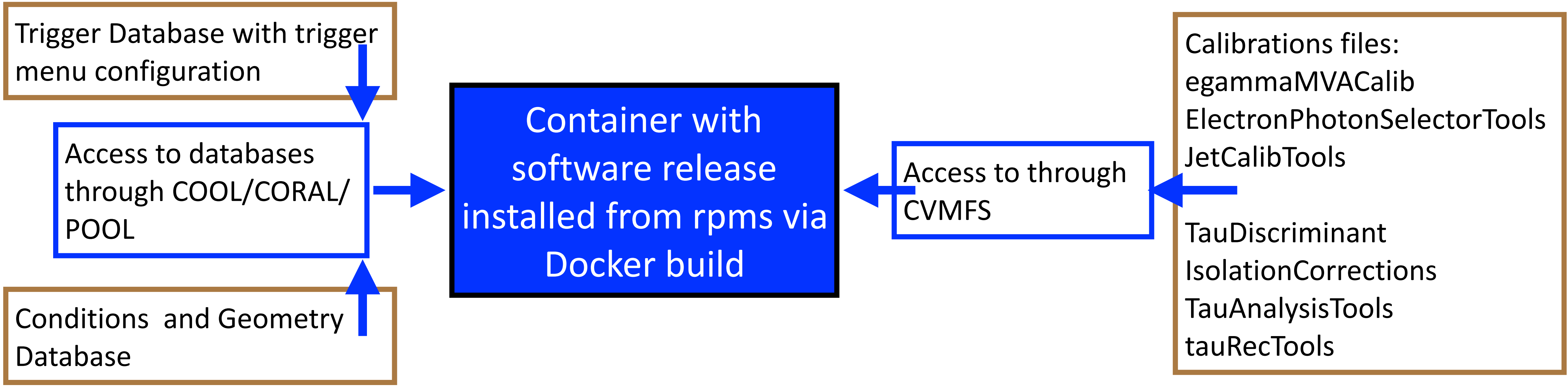
- The Wuppertal group is currently active in topic area A: work-package A1 (Containerization of user jobs, Containerization of services)
- Most recent activity has been in the development pipelines to produce images supporting local conditions database services
- The goal is to run typical production workflows without network connectivity. To this end, we are currently producing:
  - Docker/singularity images for Monte Carlo simulation
  - Docker/singularity images for MC digitization and reconstruction
  - Docker/singularity images for real data reconstruction

Current activities at ATLAS

- ATLAS is actively working on deploying workloads using containers
- The current consensus is that these containers will be single-release standalone images, which are favored over large (>200GB compressed) multi-release ones produced from de-duplicated CVMFS
- The software in single-release images is installed using package managers, so the content is well known and dependencies are automatically managed. No need to know what to copy, and what not to copy, from CVMFS
- Our goal is to make the images strictly standalone, which would be suitable for running at nodes with no outbound network connectivity (common at HPCs)

What do production workloads need from the outside?

- In order to make standalone images, we need to package the software releases with resources typically accessed through the network



Trigger Database with trigger menu configuration

Access to databases through COOL/CORAL/POOL

Conditions and Geometry Database

Container with software release installed from rpms via Docker build

Access to through CVMFS

Calibrations files:
egammaMVACalib
ElectronPhotonSelectorTools
JetCalibTools

TauDiscriminant
IsolationCorrections
TauAnalysisTools
tauRecTools

- ATLAS software releases access the backend databases through the so-called Persistency Framework

ATLAS software and the Persistency Framework

- Athena has various interfaces to fetch conditions data. These consists of the three main components of the persistency framework, used by most LHC experiments
  - **COOL**: tools for the handling of the time variation and versioning of the experiment conditions data (the database schema)
  - **POOL**: mapping between complex ROOT objects and relational database technologies (dropped by CMS in 2010). Pool file payloads are only referenced in COOL. Currrently used only by ATLAS for event and conditions data
  - **CORAL**: is an abstraction layer with an SQL-free API to access data stored using relational database technologies. CORAL plugin for Frontier (http queries). It is used directly by experiment-specific applications and internally by both COOL and POOL
- ATLAS releases also have tools that can use the same interfaces to access the databases, and make local SQLite copies: **AtlCoolCopy**

Copying conditions databases: AtlCoolCopy

- **AtlCoolCopy**: C++ tool for bulk copying and processing of COOL database data

- Copying COOL database data to a target database

```
AtlCoolCopy <sourcedb> <targetdb> <arguments>
```

- The data to be copied can be restricted by:
  - -folder, -exclude
  - -tag
  - -run, -runsince -rununtil
  - -runlumisince, -runlumiuntil
  - -timesince, -timeuntil
  - -channel -channel1 -channel2

Copying conditions databases: AtlCoolCopy

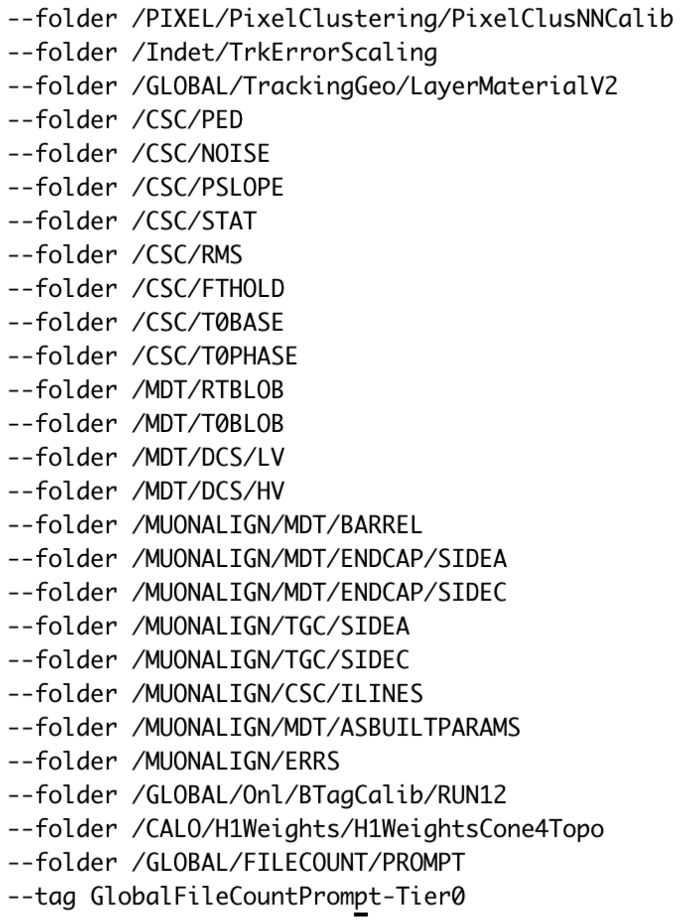
- **AtlCoolCopy**: C++ tool for bulk copying and processing of COOL database data

- Copying COOL database data to a target database

  AtlCoolCopy <sourcedb> <targetdb> <arguments>

- The data to be copied can be restricted by:
  - -folder, -exclude
  - -tag
  - -run, -runsince -rununtil
  - -runlumisince, -runlumiuntil
  - -timesince, -timeuntil
  - -channel -channel1 -channel2

→ COOL folders can be dumped from job options (conddb.dumpFolderTags())



```
--folder /PIXEL/PixelClustering/PixelClusNNCalib
--folder /Indet/TrkErrorScaling
--folder /GLOBAL/TrackingGeo/LayerMaterialV2
--folder /CSC/PED
--folder /CSC/NOISE
--folder /CSC/PSLOPE
--folder /CSC/STAT
--folder /CSC/RMS
--folder /CSC/FTHOLD
--folder /CSC/T0BASE
--folder /CSC/T0PHASE
--folder /MDT/RTBLOB
--folder /MDT/T0BLOB
--folder /MDT/DCS/LV
--folder /MDT/DCS/HV
--folder /MUONALIGN/MDT/BARREL
--folder /MUONALIGN/MDT/ENDCAP/SIDEA
--folder /MUONALIGN/MDT/ENDCAP/SIDEC
--folder /MUONALIGN/TGC/SIDEA
--folder /MUONALIGN/TGC/SIDEC
--folder /MUONALIGN/CSC/ILINES
--folder /MUONALIGN/MDT/ASBUILTPARAMS
--folder /MUONALIGN/ERRS
--folder /GLOBAL/Onl/BTagCalib/RUN12
--folder /CALO/H1Weights/H1WeightsCone4Topo
--folder /GLOBAL/FILECOUNT/PROMPT
--tag GlobalFileCountPrompt-Tier0
```

Copying conditions databases: AtlCoolCopy

- **AtlCoolCopy** can also give us the LFNs of POOL files referenced in the SQLite database

  ```
  AtlCoolCopy <sourcedb> -checkrefs
  ```

```
Total of 1 POOL files referenced

Ref 0D40E882-8E8D-3C45-B68F-E5ECC49F57B2 (43)
```

Copying conditions databases: AtlCoolCopy

- If a file catalog is provided, the PFNs are dumped instead

  ```
  AtlCoolCopy <sourcedb> -checkrefs -poolcat PoolFileCatalog.xml
  ```

  ```
  Total of 1 POOL files referenced

  *** Break *** segmentation violation
  ```

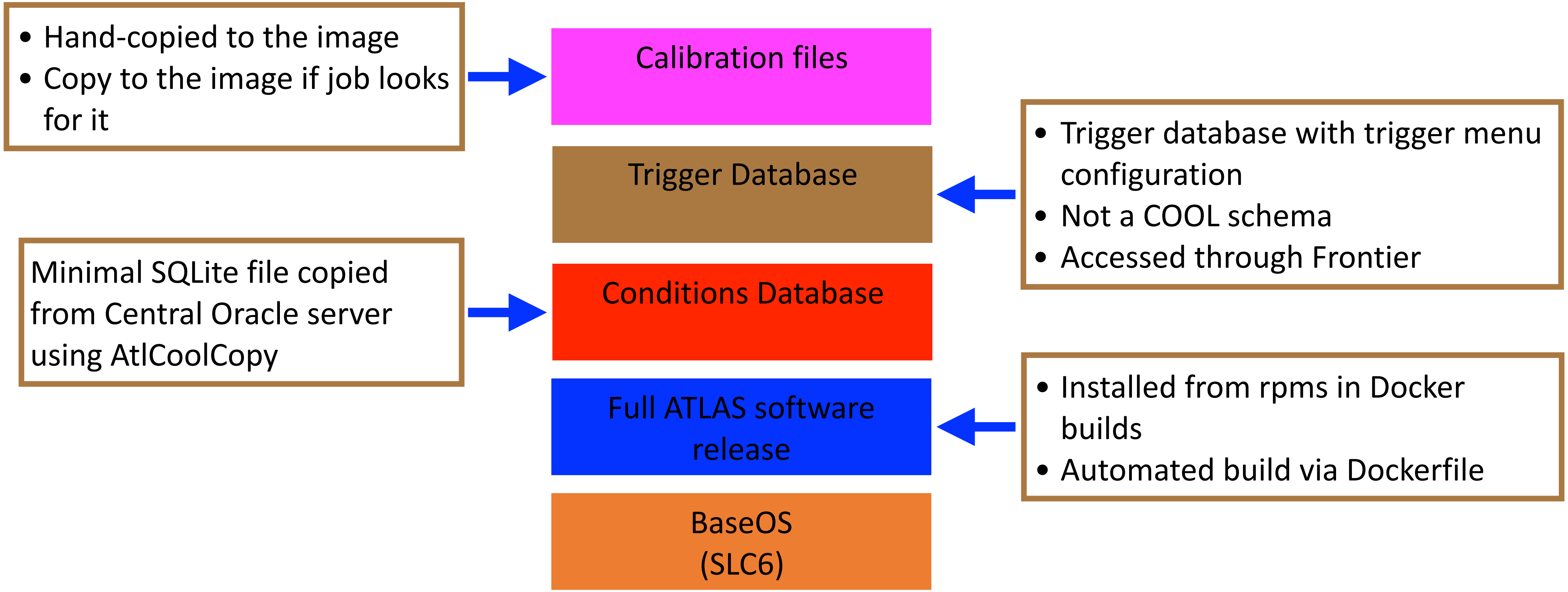
- AtlCoolCopy was used in pre-Frontier/CVMFS times and it is not currently maintained by ATLAS

- Once identified, the POOL files can be copied to the image, and the local POOL file catalog updated

- **What has AtlCoolCopy done for us?** Allowed us to create a minimal detector conditions database necessary for processing data identified by run number, lumi-block, time, etc.

Image composition and current status

- Most components are or could be packaged with the image: this is proof of concept!



- Hand-copied to the image
- Copy to the image if job looks for it

Calibration files

Trigger Database

- Trigger database with trigger menu configuration
- Not a COOL schema
- Accessed through Frontier

Minimal SQLite file copied from Central Oracle server using AtlCoolCopy

Conditions Database

Full ATLAS software release

- Installed from rpms in Docker builds
- Automated build via Dockerfile

BaseOS (SLC6)

Available images

- We have produced three types of images depending on the target workload:

- **Images for event simulation (particle-material interaction)**. tested successfully at Theta (HPC at Argonne National Laboratory
  - Single release + database release packaged for simulation by ATLAS (405M)

- **Images for MC digitization (detector read-out) and reconstruction (clustering, tracking and physics objects)**
  - Single release + database for simulation. Original SQLite file replaced by custom-made file (293M). The total image is 40GB

- **Images for real data reconstruction**
  - Single release + database for simulation. Original SQLite file replaced by custom-made file (329M) + additional POOL files (40MB). The total image is 39GB
  - The current blocker is the trigger database, which is only available through Frontier

Next steps

- Solve the **“calibration files”** problem. These files are stored in CVMFS under GroupData and amount to 500GB. We need to know which calibration files are needed for a given workflow

- Deploy/develop a tool for copying the **trigger database** to SQLite, or ask experiments to always support this option.

- Develop a generic application analogous to **AtlCoolCopy**, for copying COOL schema databases. The fact that most experiments at the LHC use COOL/CORAL as abstraction layers to the backend database solutions, motivates the need for having such a tool

Summary and Conclusions

- We have managed to run most ATLAS productions workloads in standalone containers: simulation, digitization, reconstruction
- Data reconstruction poses a greater challenge due to the need for correction and calibration data, and the additional trigger database
- Experiment software is commonly plagued by hard-coded database calls and an over-reliance on file systems served over the network. Ideally all resources necessary should be fully specified from the metadata
- We are presenting our work at CHEP 2019 in Adelaide, Australia (poster)