Stager daemon and utilities: status and ongoing development

Dario Berzano

INFN & Università – Torino, Italy

ALICE Offline Week – Nov 18, 2010

Dataset management utilities (AFDSUtils)

AFDSUtils – *set of utilities to create and manage PROOF datasets and the storage for both AAF user and admin*

They are available to users as AAF package:

```
gProof->EnablePackage("VO_ALICE@AFDSUtils::0.4.3")
```

What you can do:

- Create datasets directly from AliEn, either dynamic or static, with one command *(see next slide)*
- Repair the datasets: find corrupted files and take actions on them (delete, remove from list, re-fetch)
- See for each dataset which files are staged for real, *i.e.* not only marked as such but immediately available for analysis

AFDSUtils: create datasets from AliEn

Using function `afDataSetFromAliEn()`

Three kinds of datasets can be created:

- AliEn datasets – `setstaged` option
  ⇒ *Data is accessed directly from AliEn and never cached on your AAF*
- Dynamic datasets – `setstaged:cache` options
  ⇒ *Data is not staged, but at the first (slow) access it is cached on the PROOF pool to make subsequent accesses quicker*
- Static datasets – no options
  ⇒ *Data is staged asynchronously by the stager daemon*

Exhaustive guide with examples can be found here:
aaf.cern.ch/node/160

AFDSUtils: fast verification of datasets

Verifying datasets consists in:

- Checking file integrity (*i.e.*, can I `TFile::Open()` it?)
- Counting events in default tree and update PROOF info

Counting events with the **old method**:

- Open file, get the tree, do `tree->GetEntries()`

Counting events with the **new method**:

- Search for `Run<RUN>.Event0_<NUM>.ESD.tag.root` in same chunk dir of ESD on AliEn
- `<NUM>` is the number of events
- Fall back on old method if not applicable

New method is ∼25x faster

AFDSUtils: manual verification and real staging status

Manual verification of datasets can be done:

- When creating a dataset with `afDataSetFromAliEn()` with the option `verify`
- With `afFillMetaData()` – option `fast` may be used too

Automatic verification of staged data:

- Performed by the stager daemon: it must check integrity too, so slow check occurs

Show the contents of a dataset, with event number and files URLs:

- Use `afShowDsContent()`
- With options `"Rrn"` checks if each file is online, *i.e.* available on AAF disks (`R`), or offline (`r`)

AFDSUtils: repair datasets

Files unavailable or broken are marked as corrupted by the stager daemon: they can be repaired using function `afRepairDs()`

Actions that can be taken on files marked as corrupted, passed as options to the function:

- `unlist`: files entries are removed from the dataset
- `unstage`: files are deleted from the storage and marked as unstaged, to force re-fetch from the daemon – *available only to AAF admins*
- `uncorrupt`: files are marked as not corrupted

Users are responsible for the integrity of their own datasets

Stager daemon: status report

Daemon tasks:

- Used to stage and verify static datasets
- Processes files marked as non-staged and non-corrupted
  ⇒ *To prevent staging, dynamic datasets must be pre-listed as staged*
- Synchronizes datasets through different AAFs using AliEn
- Makes its status available on the web through MonALISA
  ⇒ *For CAF: alimonitor.cern.ch/stats?page=CAF2/datasets*

Daemon stability status report:

- In production on new CAF since early tests (April 2010)
- Finally solved problems in reporting status to MonALISA
- On CAF: over 50 TB staged and verified

Stager daemon is stable

Stager daemon: ongoing development

- Different staging queues with priorities
- Separate stage and verify processes

  ⇒ *Sometimes the verification of a file right after its download is slow*
- Verifier daemon updates file locations in datasets

  ⇒ *A file may go offline because older data is removed automatically by xrootd on each node when disk space runs out*
- Staging statistics

  ⇒ *Interesting to have daily, weekly, monthly reports on the amount of data staged, corrupted files and network speed*
- Integration of daemon in the codebase of ROOT

  ⇒ *It is a ROOT application and it has become the PROOF dataset manager, but for now it is distributed with xrootd and aaf-installer*