EPJ Web of Conferences **245**, 05033 (2020) https://doi.org/10.1051/epjconf/202024505033
CHEP 2019

Offline Software Management of the AMS experiment

Vitali Choutko[1,], *Alexander* Eogorov[1,], *Alexandre* Eline[1,], and *Baosong* Shan[2,*]

[1]Massachusetts Institute of Technology, Laboratory for Nuclear Science, MA-02139, United States
[2]Beihang University, School of Mathematical Sciences, Beijing 100191, China

Abstract. The Alpha Magnetic Spectrometer (AMS) is a particle physics experiment installed and operating aboard the International Space Station (ISS) from May 2011 and expected to last through 2028 and beyond. The AMS offline software is used for data reconstruction, Monte-Carlo simulation and physics analysis. This paper presents how we manage the offline software, including the version control, automatic integration, automatic deployment, and the distribution.

1 AMS Offline Software

As described in [1], the science data collected by the Alpha Magnetic Spectrometer [2] are transferred via the TDRS satellites to ground at White Sands, NM, and then relayed to NASAMarshall Space Flight Centre (MSFC), AL. At MSFC data are firstly transferred to AMS relay computers and finally to AMS Science Operation Centre (SOC) at CERN.

The AMS offline software is designed to serve the AMS production needs, and covers the whole data processing chain of AMS science data.

1.1 Physics Data Format

The AMS physics data format is based on CERN ROOT [3] package. Each reconstructed AMS event is represented by ROOT tree object and contains the event header, with general information of the event (around 100 bytes), 8-byte status word and (referenced) arrays (C++ STL vectors) of reconstructed objects like particle(s), track(s), cluster(s), etc., and parameters (typical total size of 8 kilobytes). On top of that, for each run, the time based information is available, such as average geomagnetic cutoff, detector live time, ISS position parameters, temperatures and voltages of AMS electronics, etc. The timing precision of such data is about 100 milliseconds. The reconstructed event files are grouped together in an event summary file, usually one file per run.

1.2 Data Production

The most important role of the AMS offline software is to server data production, which includes the preproduction, reconstruction, and simulation.

*e-mail: baosong.shan@cern.ch

1.2.1 Raw Data and Preproduction

The arrived data are raw events which are logically grouped in sequences called *runs*, each run having a unique 32bit identifier. Events inside one run are numbered in sequence. One run typically includes all events taken during a quarter of ISS Low Earth Orbit movements, i.e. $\sim$ 23 minutes, and contains in average 700,000 events. On orbits which are close to the Earth magnetic poles, the number of events in a run can exceed 2 million. Usually the AMS environmental characteristics are uniform during one run. On top of this, additional calibration information, like AMS electronics parameters, as well as various temperatures, pressures and other health and status data is recorded regularly. This calibration related information corresponds to less than 1% of the total data volume.

Preproduction de-frames the data, checks the consistency, indexes the raw events and reassembles them into the AMS *raw files*, one file per run. Resulted files are moved to permanent AMS disk storage, backed up through CERN tape system and registered in the AMS Oracle database.

1.2.2 Science Data Reconstruction

Reconstruction converts raw events recorded by AMS detector aboard the ISS to reconstructed events, containing all necessary parameters of the particle(s) crossing the detector.

Reconstruction includes two stages: first production runs continuously on the freshly arrived raw data from the Space Station and the produced data summary files are used by the Payload Operations Control Centre (POCC) for 7x24 detector monitoring. Second production runs semiannually on the same raw files, but with all the available calibrations, alignments and ancillary data from the ISS as well as monitoring values (temperatures, pressures, and voltages) to produce dataset ready for physics analysis.

Figure 1 is a screenshot of the AMS event display program, which shows the side and front view of the interactions of the passing particle with the sub-detectors.

1.2.3 Monte-Carlo Simulation

Specific data cards are given as the input of the offline software, to evaluate AMS detector performance using simulated events, taking into account the cosmic ray fluxes, the detector geometry and all relevant physics processes. The simulation uses modified GEANT-4 [4] library to generate raw events, the digitisation of the signals is simulated precisely according to the measured characteristics of the electronics, the trigger logic is equally simulated and digitised, and finally the reconstruction shares exactly the same process as in the science data reconstruction.

Figure 2 shows the event display of a simulated proton event produced by the Monte-Carlo production.

1.3 Production Workflow Management and Utilities

The offline software also contains tools for production workflow management and utilities for monitoring/analysis:

- *Job requesting*: Perl CGI [5] website for production job requesting;
- *Data validation*: Checksum verification and fast ROOT reading to validate the produced data;

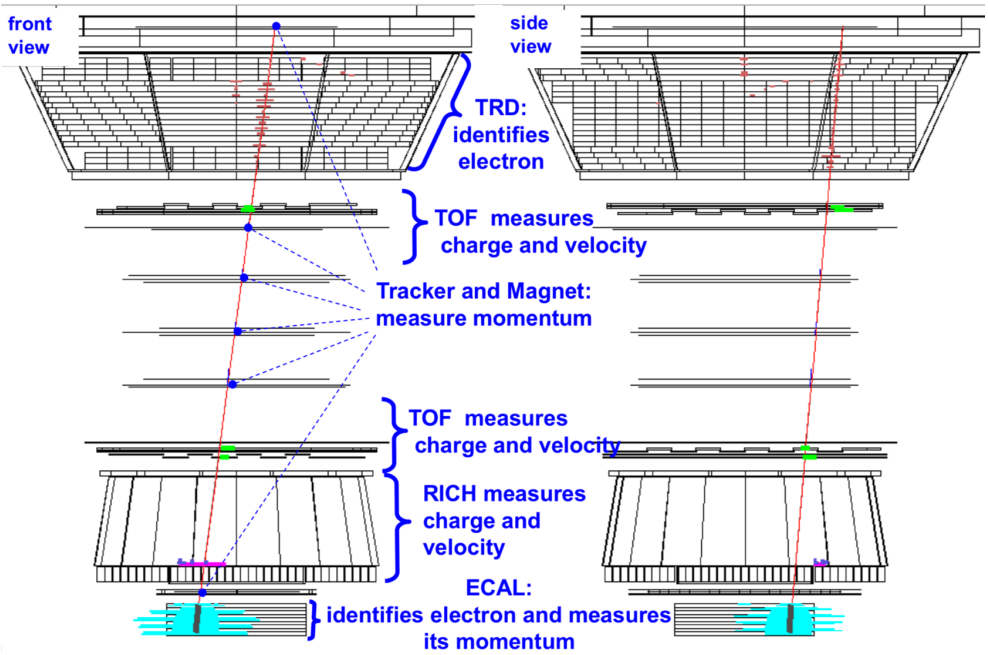

Figure 1. Example of reconstructed event.

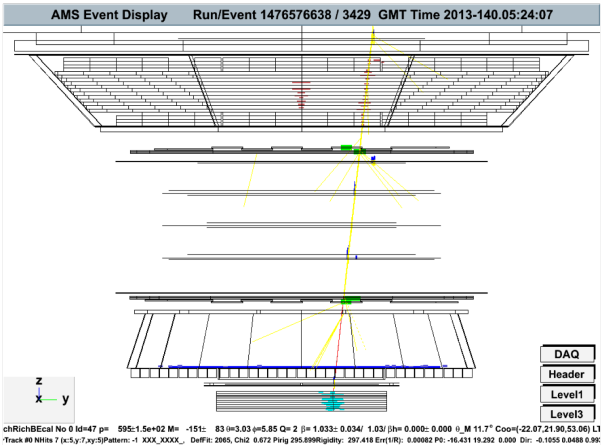

Figure 2. Example of simulated event.

- *Data moving*: Data moving between disk and tape storage, and among different computing centres;
- *Production server*: To dispatch production job to computing nodes and monitor the production processes;
- *Production Monitor*: To monitor the computing/storage facilities for production, and the production status;

- *Event display*: Visualisation utility for AMS reconstructed events.

2 Version Control

AMS uses Concurrent Versions System [6] (CVS) as the version control system for its offline software.

CVS is a central controlled version control system and the repository (CVSROOT) is located at a shared path where the clients can access. CVSROOT of AMS offline software is located on CERN AFS [7] storage and accessible by all LXPLUS hosts.

Among the features provided by CVS, there are two important ones which are used by us and must be considered during migration.

Access Control List (ACL): CVS has a simple access control mechanism relying on the ACL of the file system. AFS uses `pts` to define the group, so a `pts` group is defined to allow a group of users to be able to commit changes.

Branches: The reconstruction and simulation software of AMS has several different branches. The main branch `AMS` (also known as `vdev`) is for simulation, and the branches `BXXX_patches`, totally 23 of them up to now, are the stable versions for flight data reconstruction, to keep unique data properties for data analysis.

2.1 Migrating to Git

Although CVS is a stable, simple, and efficient version control system, it has many limitations/issues compared to the modern competitors. Git [8] is a distributed version control system which was designed to be an opposite to CVS, which means, to take CVS as an example of what not to do. Git has a lot more advantages compared to CVS.

- *Atomic operations*: CVS is based on the per-file Revision Control System [9] (RCS) version control system, so commits (and other operations) are not atomic, and an interrupted operation may lead to an inconsistent state and a corruption. While in Git all operations are atomic, in this way, the consistency is always assured.
- *Changesets*: Changes in CVS are per file, and they cannot be grouped. This made it very difficult to track the changes of the whole project. Git always refers to the whole project for all commits. This is very important and makes it very easy in to revert or undo whole change and to do partial checkouts.
- *Commit before merge*: Since Git is designed to be the opposite of CVS, unlike CVS uses merge-before-commit Git uses commit-before-merge. CVS forces users to first update the working directory and resolve conflicts before allowing committing. On the contrast Git allows saving the state first, then merging the changes from others or having other developer resolve conflicts.

Based on the advantages above and the fact that CERN provides GitLab Service officially, we decided to migrate from CVS to CERN GitLab.

2.1.1 Access Control List Migration

As mentioned the CVS ACL is based on the current hosting file system AFS, and the migration to Git requires a mapping between the membership of the `pts` group to the user system of CERN GitLab. We developed a script to retrieve the user IDs from the `pts` group, use the IDs to get the information of the users from `phonebook`, and write into the mapping file.

Table 1. Access Control Mapping

Role in CVS	Role in GitLab	Comments
Read-only users and most read-write users	Reporters	View/Clone/Submitting merge requests
Experienced frequent contributors	Developers	Committing/Review merge requests

Compared to CVS, Git provides finer granularity of access control. To adapt this, we re-designed the access control strategy for offline repositories.

The repository's visibility is set to be "private". The read-only users/groups in CVS are migrated to be "reporters", while the read/write users in CVS are divided into two groups, "reporters" and "developers". As shown in Table 1, experienced frequent contributors are "developers" who can commit their changes directly to the repository, and the others, together with the previously "read-only" users, are "reporters" who can fork the code, make their changes, and submit "merge requests". The "developers" can review the commits and decide if the merge request can be approved.

2.1.2 Importation

The importation is done by the Git tool `cvsimport`, which scans the CVS repository root directory, retrieves all the contents, including code, commits, users, tags, branches, etc., and writes into a local Git repository. Below is an example of the importation commands.

```
git cvsimport -v -d :local:$CVSROOT -A cvs2git.map -o AMS -C AMS AMS
```

This command reads the generated user mapping file `cvs2git.map` described in Section 2.1.1 and set `AMS` as the default branch for `HEAD`.

Running the importation for the first time on a big repository may take hours. After it completes, we need to push it into a created CERN GitLab project. This command reads the generated user mapping file `cvs2git.map` and set `AMS` as the default branch for `HEAD`.

```
cd AMS
git remote add origin https://:@gitlab.cern.ch:8443/AMS.dev/AMS.git
git push -u origin --all
git push -u origin --tags
```

The first importation will generate a default branch `master` which is a duplication of the main branch `AMS`. So we removed the branch `master` and make `AMS` as the default branch in Git.

Following the same procedure, we migrated all the other CVS repositories related to AMS offline computing, including the active ones (e.g. modified GEANT4 and ROOT code) and archived ones (e.g. AMS-01 code).

2.1.3 Periodic Synchronisation

The importation is done by a set of periodic synchronisation instead of a one-time operation. Before the GitLab repository is finally announced to start operation, many incremental importations have to be done to synchronise the CVS commits. The incremental importations use the same `cvsimport` command followed by the `push` command discussed in Section 2.1.2. However, the subsequent synchronisation focuses only on the new commits and costs much less time than the initial importation.

3 Continuous Integration and Continuous Deployment

Another advantage of migrating to Git is to enable Continuous Integration (CI) and Continuous Deployment (CD).

3.1 Automatic Build

The most important role of AMS offline code is to provide tools for data reconstruction and Monte-Carlo simulation. Upon the arrival of a new commit or merge request, the first step for CI is to build the producer programs for reconstruction and simulation.

For this, we define the building rules in the file `.gitlab-ci.yml`. Since the repository contains various components of AMS offline computing, we first define which changes will trigger the building of the producers. Once triggered, we use a standard Scientific Linux CERN 6 docker image, install the Open SSH client, copy repository code to a specific host for building, and finally perform the building on that host. By this way, we can easily use all the available shared "runners" from CERN GitLab service without bringing too much load.

The current automatic build procedure is just the first step, which is subject to further optimisation and expansion. The commits to other components than the producers, like production utilities written in Perl/Python, should also undergo similar procedures.

3.2 Automatic Test

Successful building will be followed by further testing on the functionality and performance of the producers. Testing includes several jobs covering data reconstruction, Monte-Carlo simulation, and Monte-Carlo reconstruction, and the jobs have to be run on all the supported platforms. Current supported platforms include Scientific Linux CERN 6 and CERN CentOS 7.

In future, more testing procedures will be added to cover more parts of the production and new platforms will be added as well.

3.3 Automatic Deployments

Once the building and testing is completed, the new producer executables will be deployed on AFS.

4 Distribution

In this section, we will discuss the distribution of the AMS offline data, which includes not only the software, but also the databases, the auxiliary data, etc.

The main storage for AMS offline data is AFS, supported by CERN IT department. Inside the `Offline` directory, there are the offline software code, the executables and libraries built from the code, the calibration and alignment data, the time-dependant-variable databases, the ISS position parameters, etc. Although AFS is a reliable shared file system, it is not designed to handle high Input/output operations per second (IOPS). To avoid the high amount of requests to AFS, we allow only the production accounts to access the offline databases on AFS. The offline contents are synchronised to CVMFS [10] periodically for other users' access.

4.1 CVMFS Synchronisation

After the AMS CVMFS Tier-0 node has been updated the the new CVMFS server software, we found that the previous syncing mechanism has to be modified accordingly to avoid filling up the root disk of the node. So we re-designed the synchronisation procedure by separating the scanning and the syncing processes. First a dry run of `rsync` is performed to find out all the new/updated/deleted files, directories, and links, and after this is done, only the changes are synced to CVMFS. We've tried different granularities for the syncing pass, and found that syncing all the changed directories (instead of files) is not practical due to significantly increased time and disk usage (factor 10x or even more).

A full synchronisation process can finish within 4 hours with the new procedure, of which the dry run is 3 hours and the syncing is usually within 1 hour.

The performance boost makes it possible to increase the frequency of the synchronisation. We start to apply a daily CVMFS syncing with tags, and totally around 30–40 tagged snapshots will be kept: daily for the past 7 days, weekly for the past 8 weeks before the dailies, monthly for the past 12 months before the weeklies, and yearly for the rest.

5 Conclusion

The AMS offline software is used for data reconstruction and Monte-Carlo simulation, being the fundamental tool for physics analysis works. We migrated the source control system from CVS to GitLab, and the main repository counts nearly 1.4 million lines of code and more than ten thousand commits. The automatic integration and automatic deployment is also started to be employed for a better software management. And a new procedure is developed to efficiently synchronise the offline software/data from AFS to CVMFS.

References

[1] V. Choutko, A. Egorov, A. Eline, B. Shan, *Computing Strategy of the AMS Experiment*, in *Journal of Physics: Conference Series* (IOP Publishing, 2015), Vol. 664, p. 032029

[2] S. Ting, Nuclear Physics B-Proceedings Supplements **243**, 12 (2013)

[3] I. Antcheva, M. Ballintijn, B. Bellenot, M. Biskup, R. Brun, N. Buncic, P. Canal, D. Casadei, O. Couet, V. Fine et al., Computer Physics Communications **182**, 1384 (2011)

[4] S. Agostinelli, J. Allison, K.a. Amako, J. Apostolakis, H. Araujo, P. Arce, M. Asai, D. Axen, S. Banerjee, G.. Barrand et al., Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment **506**, 250 (2003)

[5] S. Guelich, S. Gundavaram, G. Birznieks, *CGI Programming with Perl: Creating Dynamic Web Pages* (" O'Reilly Media, Inc.", 2000)

[6] B. Berliner, J. Polk, *Concurrent versions system (cvs)* (2001)

[7] J.H. Howard et al., *An overview of the andrew file system* (Carnegie Mellon University, Information Technology Center, 1988)

[8] D. Spinellis, IEEE software **29**, 100 (2012)

[9] W.F. Tichy, Software: Practice and Experience **15**, 637 (1985)

[10] C. Aguado Sanchez, J. Bloomer, P. Buncic, L. Franco, S. Klemer, P. Mato, *CVMFS-a file system for the CernVM virtual appliance*, in *Proceedings of XII Advanced Computing and Analysis Techniques in Physics Research* (2008), Vol. 1, p. 52