A Dynamic System for ATLAS Software Installation on OSG Grid sites

X. Zhao¹, T. Maeno¹, T. Wenaus¹, F. Leuhring², S. Youssef³, J. Brunelle³, A. De Salvo¹ and A.S. Thompson⁵

¹ Brookhaven National Laboratory, Upton, NY 11973-5000, USA
² Indiana University, Bloomington, IN 47405, USA
³ Boston University, Boston, MA 02215, USA
⁴ INFN Roma, Piazzale A. Moro 2, IT-00185 Roma, Italy
⁵ University of Glasgow, Glasgow G12 8QQ, Scotland, UK

Email: xzhao@bnl.gov

Abstract. A dynamic and reliable system for installing the ATLAS software releases on Grid sites is crucial to guarantee the timely and smooth start of ATLAS production and reduce its failure rate. In this paper, we discuss the issues encountered in the previous software installation system, and introduce the new approach, which is built upon the new development in the areas of the ATLAS workload management system (PanDA), and software package management system (pacman). It is also designed to integrate with the EGEE ATLAS software installation framework. In the new system, ATLAS software releases are packaged as pacball, a uniquely identifiable and reproducible self-installing data file. The distribution of pacballs to remote sites is managed by ATLAS data management system (DQ2) and PanDA server. The installation on remote sites is automatically triggered by the PanDA pilot jobs. The installation job payload connects to a central ATLAS software installation portal, making the information of installation status easily accessible across OSG and EGEE Grids. The issues encountered in running the new system in production, and our future plan for improvement, will also be discussed.

1. ATLAS software installation on Grid sites

ATLAS Grid production, like many other VO Grid applications, requires the ATLAS software releases to be installed on all Grid sites in advance. A dynamic and reliable installation system is crucial for the timely start of ATLAS production jobs using new releases, as well as to reduce job failure rate.

ATLAS Grid production runs across several different Grids, among them two major ones are the Open Science Grid (OSG)[1][2], and the Enabling Grids for E-sciencE (EGEE)[3]. Due to historical reasons, there have been two different ATLAS software installation systems for OSG and EGEE sites.
2. Previous ATLAS software installation system for OSG

The old ATLAS software installation system is built on top of Globus[4] client tools and OSG information system. It’s composed of a set of shell scripts.

![Diagram of previous ATLAS software installation system for OSG sites](Image)

**Figure 1: Previous ATLAS software installation system for OSG sites**

2.1 Workflow

Figure 1 shows the work flow:

- From the installation host, installation scripts are constructed from templates, based on the information about the target sites, which is retrieved from the OSG information system, CEMon[5] and BDII [6];
- Globus GridFTP client is used to transfer to remote sites' gatekeepers the newly customized installation scripts, which are then invoked using `globus-job-run` command from the installation host;
- On the remote gatekeeper, when the installation scripts are invoked, they will first check whether the underlying software package management tool is installed or not. In ATLAS, we use `pacman`[7] as the major software package management tool. If it’s not available on remote site, installation scripts will firstly download and install it;
- The installation scripts get the ATLAS releases directly from the `pacman` cache at CERN;
- Once the `pacman` installation is finished, OSG BDII will be updated with the new releases on sites.

This is a pretty straightforward workflow, using the basic Globus client tools. In production environment, we run into several issues.

2.2 Problems with the previous installation system

2.2.1 Installation is done on the Globus gatekeepers of Grid sites. This causes several problems:
• On some sites, gatekeeper nodes run different operating system than what the worker nodes run, the installation environment is not the same as where the real jobs will run. This makes it difficult to validate the installation;
• It increases the load on the gatekeeper. Gatekeeper is already a heavy loaded server, with many other OSG services and Globus jobmanager processes running all the time. On most of the OSG sites, application software is installed on NFS area. A slowness of NFS server during the installation process will create high load on the gatekeeper server, stopping the site from serving any new job and file transfer requests.

2.2.2 Installation retrieves the release packages directly from the CERN pacman cache. This increases the load on the CERN web server. And more importantly, it slows down the installation process. On many USATLAS sites, one ATLAS release installation usually takes four or more hours to finish, a result of transferring thousands of tarballs over the network across the Atlantic ocean, sometimes combined with slow NFS response.

2.2.3 As mentioned in the first section, OSG and EGEE each have its own installation system, with different monitoring tools and archival system. A unification will certainly benefit the whole international ATLAS collaboration.

3. The new installation system

To address the above issues with the previous installation system, we come up with a new design, which leverages recent advances in two distinct areas: (1) ATLAS production workflow management system, PanDA[8], and (2) ATLAS software package management system, pacman[7].

The main design points are:
• Installation is done locally on the worker nodes of remote Grid sites, using PanDA pilots;
• Install from a local pacman cache from the worker nodes, using pacball, which is transferred via the ATLAS data management system (DQ2)[9];
• Unify the two installation systems between OSG and EGEE, by sharing the same installation scripts, web portal and installation database, which are originally from EGEE.

The workflow of the new system is shown in Figure 2. Below we will go through each of the main components in more details.
3.1. Installation on worker nodes using PanDA pilots

The ATLAS Production ANd Distributed Analysis system (PANDA) is now the workload management system for the worldwide distributed production of the international ATLAS collaboration. It implements just-in-time late binding of real job payloads with CPU slots, using pilots.

Figure 3 shows the pilot-based job flow in PanDA. PanDA keeps a steady pilot streams to all Grid sites, using Condor-G[10] job scheduler. Pilots themselves are regular jobs on the local batch system of the remote sites. Pilots provide sandboxes for the real job payload. Once a pilot lands on a CPU slot, it will ask PanDA server for a real job payload, and run it. If there are no real jobs to be assigned, pilot will exit immediately, releasing the CPU slot. The real production and user analysis jobs are submitted and defined in PanDA server using a standard PanDA API.

In our new installation system, instead of running the installation commands through Globus clients ourselves, we define installation tasks as job payloads, and submit them to PanDA using the standard API. PanDA will then assign the jobs to an available pilot from the target sites. This way we can make sure the installation and validation of ATLAS software releases are done in the real production job environment, i.e. pilots on the worker nodes.

![Diagram of PanDA job flow](image)

**Figure 3: Job flow in PanDA**

3.2. Pacman cache localization

Pacman allows users to wrap a pacman cache in a script, called pacball[7]. Pacball is machine-independent, self-installing and self-contained script, easy to be moved around and executed without other pacman tools. Pacball is not updatable, each pacball has an md5 checksum attached, making it a uniquely identifiable and reproducible software environment.
In the new installation system, ATLAS software pacball is treated as the input data file for the installation jobs. DQ2 makes each pacball into its own dataset, transfer them to the storage element (SE) of all Tier1 sites. In OSG, pandamover[8] jobs transfer the pacballs from BNL Tier1 to all Tier2 and some large Tier3 sites. When installation job starts, pilot will copy the pacball dataset from local site SE to local disk on worker node. This way we can run the installation out of a local pacman cache, no need to always access the CERN web server through wide area network.

3.3. Unification

We share the same installation scripts, software installation portal and database between OSG and EGEE Grids. This provides a unique interface for the international ATLAS collaboration to query for software installation status across OSG and EGEE Grid sites. The same information is also available on the OSG and EGEE BDII, respectively.

Another important aspect of the unification is that, the new installation system is built on top of the application level ATLAS workflow management (PanDA) and dataflow management (DQ2) systems. These systems hide the details of the various Grid infrastructures, have fault tolerant and robust software mechanisms built in to cope with common Gird middleware failures and Grid site issues. Wherever ATLAS Grid production runs, these systems will work. Therefore they provide an ideal and uniform platform to build the installation system on, makes it interoperable and usable for all Grid favors inside ATLAS collaboration.

4. Implementation of the new installation system

We will go through some of the details of implementing the major components of the new installation system.

4.1. Installation queues in PanDA

A new job type, installation job, is defined in PanDA, to distinguish it from production and user analysis jobs. And for each OSG production queue in PanDA, a new corresponding installation queue is added.

The installation queue is defined in such a way that the installation pilots will share the same job slots on Grid sites with the production pilots, but at a much lower rate. Installation pilots are submitted to Grid sites using voms proxy with ATLAS software role, which will be assigned to software manager’s account (usatlas2) on OSG sites. The software manager account has the permission to write to the VO application software area on all OSG sites.

4.2. Pacball creation and transfer in DQ2

Starting with 14 release series, every new ATLAS software base kit and production cache will have a corresponding pacball created. Then following the same data flow model[2] as production and user analysis input data does, pacballs will be replicated to all production sites via DQ2 and made available on worker nodes for local installation. DQ2 site service uses LFC[11] as file catalog, LFC ACL is modified to give write permission of software manager’s voms proxy, allowing log files of installation jobs to be saved to local SE, and eventually transferred back to BNL Tier1 SE.

4.3. Installation job payload
The installation job payload is adopted from the original EGEE ATLAS installation scripts, modified to cover OSG Grid environment and use cases. PanDA team implements the job definition API to submit the installation jobs to PanDA.

The original EGEE ATLAS installation portal and database are extended to be multi-grid enabled, now serving both OSG and EGEE sites. The information of existing ATLAS software installed on OSG sites is migrated to the new portal and database, providing a complete picture of software installation for ATLAS collaboration across the two major Grids.

5. Issues and future plan

We have been running the new installation system on OSG USATLAS production for a couple of months now, the current status can be found at the ATLAS installation portal[12]. Below we will discuss some issues encountered in production and our future plan.

5.1. Integrity check

One long-standing problem in installation system is how to check the integrity of the software installed on all sites. In the distributed production environment, sites go up and down all the time, so there are always sites that fall behind in installation of new software releases, which will cause a large number of job failures if not caught quickly.

In the new installation system, we plan to address this issue by automating the job payload scripts. Every time an installation job starts, it will scan existing releases against a target list, and go ahead to install the missing ones. Since all the available release pacballs are registered in DQ2, we should be able to construct the target list by querying the DQ2 central catalog. Another advantage of this approach is that, the creation of a new pacball will automatically trigger its installation on all sites, requires less manual operation.

5.2. Operational issues and improvements

5.2.1. Latency in pacball dataset replication to Tier1

ATLAS releases are the first thing sites need in order to run the production and user analysis jobs. Production managers and users usually want a new release to be deployed on the Grid very quickly, so new jobs can start to produce new physics results. In the early days of operating the new system in production, one major issue is DQ2 can’t transfer the pacballs to destination sites promptly, we then have to go back to manually start the installation.

To address this issue, we plan to: (1) make the creation and registration of new pacballs part of the standard ATLAS release procedure, managed by ATLAS Software Infrastructure Team (SIT); (2) work with ATLAS Data Management team to increase the priority of pacball datasets replication to Tier1 sites.

5.2.2. Independent installation queue in PanDA

Right now the installation queues are virtual, attached to a production queue. They basically share the same site configuration parameters in PanDA. One problem of this model is that, whenever a production site is set offline, the corresponding installation queue will be disabled as well. Well sometimes the reason for setting a site offline is to fix a software release problem. Coupling them together prevents new installation jobs from going. So our plan is to create real installation queues
inside PanDA, which will allow us to control their behavior independently from other production and user analysis operations.

References
[1] Open Science Grid (OSG) web site: http://opensciencegrid.org
[2] The Open Science Grid, Ruth Pordes etc. J.Phys.:Conf. Ser. 78(2007)012057
[3] Enabling Grids for E-sciencE (EGEE) web site: http://public.eu-egee.org
[4] Globus Toolkit. http://www.globus.org; I.Foster and C. Kesselma. Globus:A metacomputing infrastructure toolkit. In Proc. of Workshop on Environments and Tools, SIAM., 1996
[5] CEMon project web site: http://grid.pd.infn.it/cemon/field.php?n=Main.AboutCEMon
[6] Berkeley Database Information Index V5 (BDII) twiki page: https://twiki.cern.ch/twiki/bin/view/EGEE/BDII
[7] Pacman. http://atlas.bu.edu/~youssef/pacman/index.html
[8] T. Maeno, PanDA:distributed production and distributed analysis system for ATLAS, Journal of Physics, Conference Series, Volume 119, Issue 6, pp. 062036(2008)
[9] M. Branco, D. Cameron, B. Gaidioz, V. Garonne, B. Koblitz, M. Lassnig, R. Rocha, P. Salgado, T. Wenaus. Managing ATLAS data on a petabyte-scale with DQ2. Journal of Physics, Conference Series, Volume 119, Issue 6, pp. 062017(2008)
[10] J. Frey, T. Tannenbaum, I. Foster, M. Livny, and S. Tuecke, “Condor-G: A Computation Management Agent for Multiple-Institutional Grids”, Proc. of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC10)
[11] CERN LFC web page: https://twiki.cern.ch/twiki/bin/view/LCG/LfcWlcg
[12] URL for ATLAS Installation portal: https://atlas-install.roma1.infn.it/atlas_install/