Automatic Integration Testbeds validation on Open Science Grid

To cite this article: J Caballero et al 2011 J. Phys.: Conf. Ser. 331 062027

View the article online for updates and enhancements.

Related content
- The ATLAS PanDA Pilot in Operation P Nilsson, J Caballero, K De et al.
- Recent Improvements in the ATLAS PanDA Pilot P Nilsson, J Caballero Bejar, G Compostella et al.
- A method of searching LDAP directories using XQuery Ted Hesselroth
Automatic Integration Testbeds validation on Open Science Grid

J Caballero1, S Thapa2, R Gardner2, and M Potekhin1

1Brookhaven National Laboratory, PO BOX 5000 Upton, NY 11973, USA
25735 South Ellis Avenue, Chicago, IL 60637, USA
E-mail: Email: sthapa@ci.uchicago.edu

Abstract. A recurring challenge in deploying high quality production middleware is the extent to which realistic testing occurs before release of the software into the production environment. We describe here an automated system for validating releases of the Open Science Grid software stack that leverages the (pilot-based) PanDA job management system developed and used by the ATLAS experiment.

The system was motivated by a desire to subject the OSG Integration Testbed to more realistic validation tests. In particular those which resemble to every extent possible actual job workflows used by the experiments thus utilizing job scheduling at the compute element (CE), use of the worker node execution environment, transfer of data to/from the local storage element (SE), etc. The context is that candidate releases of OSG compute and storage elements can be tested by injecting large numbers of synthetic jobs varying in complexity and coverage of services tested. The native capabilities of the PanDA system can thus be used to define jobs, monitor their execution, and archive the resulting run statistics including success and failure modes. A repository of generic workflows and job types to measure various metrics of interest has been created. A command-line toolset has been developed so that testbed managers can quickly submit "VO-like" jobs into the system when newly deployed services are ready for testing. A system for automatic submission has been crafted to send jobs to integration testbed sites, collecting the results in a central service and generating regular reports for performance and reliability.

1. Introduction

The Open Science Grid (OSG)[1] provides middleware services for clusters providing grid computing resources to virtual organizations and users. The OSG provides a new release of the middleware every 4-6 weeks. However, OSG resources are in continuous production so these releases must be thoroughly tested before being released for deployment.

This testing is provided by an integration testbed (or ITB) which provides a distributed grid environment over a number of sites with a full complement of compute and storage resources running on a variety of hardware and software platforms. OSG release candidates are run on the ITB and undergo the usual installation, configuration and functional validation steps. However, full validation requires end to end testing by real VO jobs.

Given time and effort constraints present, conducting a full validation is difficult if incremental changes to isolated components need to be tested. In order to allow this testing to be done on the ITB, a new testing infrastructure, called the ITB robot, that leverages that existing capabilities of the ATLAS PanDA[2] system has been created. In addition to allowing for more extensive
testing with less manpower, this system allows for arbitrary payloads to be deployed on resources as well as web based facilities to automate tests and view statistics on previous tests run against resources.

2. Architecture

The ITB robot architecture (figure 1) consists of 3 major components: a client submit host, a web host, and the PanDA backend. The client submit host is used to be submitted tests to various ITB resources. The web host records test information, pending tests, and provides a location where testing details can be viewed. PanDA provides a pilot based infrastructure that schedules and runs tests on ITB resources. We describe briefly each of the components below.

2.1. Client Submit Host

This system can be a permanent server used to submit automated daily tests to the various ITB sites, or it can be an ad-hoc system that is used by an ITB participant to test a resource. The client submit host uses the PanDA client tools[3] to submit a given test to be run on a particular ITB resource.

For ad-hoc usage, an ITB participant can download the PanDA client tools, and the OSG client tools and use them to schedule tests on ITB resources. After installation of the client tools and the OSG client, an ITB participant can run the submission script and indicate the resource that should run the test, the particular test to run (with parameters for the test), and the number of times to run the test. After the tests have been successfully submitted to the
PanDA system, the client tools will return the PanDA ids that correspond to the scheduled tests. These ids can then be used to subsequently track the progress of the tests and review the results using the PanDA monitor website.

The ITB robot system has a permanent server that is used to send tests to ITB resources daily. This system uses the Python api provided by the PanDA client tools to programmatically schedule and submit tests to the PanDA system and then record the ids of the jobs for subsequent tracking.

2.2. Web host
The ITB robot uses a system based on Django[4] to schedule, track, and present test results. Django provides a reliable infrastructure that integrates a web server along with an object relational mapper and database abstraction that allows for rapid addition of new features, such as additional types of reports on test results, and tests to an existing website.

When new tests are scheduled using the automated testing facility, this is registered with the web host to record this pending test and the PanDA id associated with the scheduled job is also recorded. When scheduled the test is run on a site, the results of the test are submitted to the web host using a common XML format.

The web host provides reports on recorded test results, pending tests, and information on ITB resources that are being tested. For example, pilot settings, or PanDA queues for ITB resources are provided. Reports and information are accessible through a publicly accessible Django site so that ITB participants can review the current status of tests run on their site.

2.3. PanDA System
PanDA is a pilot based job scheduling system that is used for ATLAS to run its analysis and production computing worldwide. The ITB robot uses the job submission, scheduling, and
tracking capabilities of PanDA to schedule and run tests on ITB resources (figure 3). Each ITB resource that is being tested has a separate configuration created for it in PanDA. Pilots from PanDA are then sent to the resource. When a probe is scheduled to run on the resource, the pilot downloads the specified test script and then runs it. The current test scripts are python scripts that run a given workload and then report the results back to the web host.

2.4. Test scripts

The test scripts used by the ITB robot can be divided into two categories: resource functionality tests and job simulations.

The functionality tests check for features that should be present on every OSG site. This allows for things such as the existence of scratch space, data transfer areas, and scheduling to be tested. Each of these tests each only check for a given feature, allowing for basic site functionality to be tested.

Job simulations allow for a typical job to be simulated. This test downloads a file from a web site, uploads the file to a resource’s storage element using SRM[5], downloads the file and verifies the file integrity, and then does some matrix inversions to use some cpu time. The test records transfer rates for the various transfers and the time required to complete the matrix inversions and reports this to the web host. This test and similar tests allow for broad testing coverage of OSG software in the ITB.
| Name            | Test type    | Purpose                                                                 |
|-----------------|--------------|-------------------------------------------------------------------------|
| simulate_se_job | Job simulation | Simulate a job that runs on a worker node, reads and writes to a storage element (SE) using SRM client tools and does a matrix inversion on a random 100x100 matrix |
| cpu_burn        | Job simulation | Simulate a job that runs on a worker node and just does cpu intensive calculations |
| check_osg_data  | Functional test | Check for the existence of an OSG data directory and verify that it has the correct permissions |
| check_wn_tmp    | Functional test | Check for the existence of the worker node temp directory and verify that it has the correct permissions |

Table 1. Current job types

| Job Type     | Number of sub-missions | Run on UC_ITB | Run on OUHEP_ITB | Run on LBNL_ITB | Run on BNL_ITB |
|--------------|-------------------------|---------------|------------------|-----------------|---------------|
| check_osg_data | 3863                    | 1265          | 381              | 868             | 1349          |
| check_wn_tmp  | 3870                    | 1273          | 380              | 861             | 1356          |
| simulate_se_job | 5094                    | 2510          | 380              | 860             | 1344          |
| Total         | 12827                   | 5048          | 1141             | 2589            | 4049          |

Table 2. Job Statistics

3. Results
The current job types that are running on the ITB robot are listed in table 1. Typically an ITB resource has simulate_se_job, check_osg_data, and check_wn_tmp run on it ten times each day. The results are analyzed and presented through web reports (figure 2) outlining site information and success rates of tests run on that site over time.

Since the ITB robot started to be used for testing on September 12, 2010, over 12000 tests have been run on 4 different ITB resources (table 2). The resources were added to the system one by one resulting in different number of tests being run on the various resources. The majority of the tests run had runtimes on the order of 1 second or less although the simulate_se_job tests have a run time that varies from 300 seconds at BNL_ITB to 800 seconds at OUHEP. The simulate_se_job test includes a matrix inversion of a random 100x100 matrix as part of its workload so its runtime depends heavily on the processing power of the compute nodes at the resource that it is run on. Although it also transfers a 100MB file between an SE and the compute node, all the transfers are done with a SE local to that resource so transfer rates (~3-8 MB/s) are roughly the same on the different resources and don’t contribute significantly to the variation in runtime.

4. Conclusions
Using the ITB robot, integration and validation of new potential OSG releases has been significantly improved. The new system allows for quicker, more thorough testing of new releases and for detailed examination of test results from testing for previous releases. The ITB robot infrastructure allows for larger scale testing of existing ITB resources and for multiple tests to be run simultaneously on resources to allow scalability testing and to test concurrent job submissions. Since the ITB robot uses the same PanDA infrastructure used by ATLAS to do job submission, the ITB robot can potentially be scaled up to run hundreds of jobs from multiple users simultaneously.
Acknowledgments
This work was supported in part by the Office of Science, U.S. Department of Energy, SciDAC program under Contract DE-FC02-06ER41436 and the National Science Foundation Cooperative Agreement, PHY-0621704. We wish to acknowledge the assistance of the participating institutions and technical staff of the Open Science Grid, and the ATLAS PanDA development and operations teams.

References
[1] Pordes R and et al 2007 *J. Phys.: Conf. Ser.* **78**
[2] Maeno T 2008 *J. Phys.: Conf. Ser.* **119** 4
[3] Caballero J Panda client tools URL http://www.opensciencegrid.org/panda
[4] The django project URL http://www.djangoproject.com/
[5] Sim A and Shoshani A 2008 The storage resource manager interface specification version 2.2 Document in Full Recommendation GFD.129 Open Grid Forum