Mathematical Problems of Computer Science 30, 40--46, 2008.

Custom Software Installation and Availability Assurance of Computing Elements in LHC Grid [1]

Karen Mkoyan

Yerevan Physics Institute
karen@yerphi.am

Abstract:

This paper describes the process of custom software installation and validation in the LHC Grid. It also provides set of scripts developed by author for availability assurance and overall monitoring of computing elements prior to installation, and a script for the installation and validation process itself.

1. Introduction

In the LHC Grid, standard High Energy Physics (HEP) applications and libraries are available for all Virtual Organizations. It also provides Experiment Software Area for each VO, where the custom, experiment specific software could be installed. The Experiment Software Manager (ESM) is the member of the experiment VO entitled to install Application Software in the different sites.

ESM has permission to write into experiment area and add/remove software at any time without notifying the site manager. The installation is done on a per site base; no root access is required to install experiment software. The ESM can manage (install, validate, remove...) Experiment Software on a site at any time through a normal Grid job. The installed custom software, is also published in the Information Service, so that user's jobs are automatically sent and run on nodes where the user's required custom software is installed.

Since the installation jobs have no scheduling priorities and are treated as any other job of the same VO in the same queue, there is always a delay in the operation if the queue is busy. Also due to scheduled maintenance, or some other technical reasons not all Computing Elements of a Virtual Organization are always available for software installation. Therefore ESM needs to have a list of available Computing Elements prior to the installation.

In Chapter 2 the installation and validation mechanism in gLite 3.0 is described.

2. Custom Software Installation: General Procedure

All sites provide a dedicated space where each supported VO can install or remove software. The amount of available space must be negotiated between the VO and the site.

An environmental variable holds the path for the location of a such space. Its format is the following:

[1] This work has been supported by INTAS, Young Scientists Fellowship grant Ref. 05-110-4812

- *VO_<name_of_VO>_SW_DIR*

Different site configurations provide two different scenarios for software management:

1. A file system is shared among all WNs of the same cluster: the software is installed in the file system, and the jobs that want to use it just need to use the VO_<name_of_VO>_SW_DIR variable to access the software through POSIX calls.
2. The site does not provide a shared file system. There are in this case two different possibilities depending on whether the site provides the Tank & Spark [1] mechanism for automatic software propagation or not:
   - WITH T & S: the software gets permanently installed on each WN.
   - WITHOUT T & S: Bundles of software are stored in the closest SE in the site, and they get only installed in a WN when a job requires them. After the job finishes, the software is automatically erased by the batch system from the WN. This is done to avoid the exhaustion of disk space in the node.

In view of the fact that most sites in LHC Grid use shared file systems among Worker Nodes, this article discusses only the first case.

The general procedure for the custom software installation in LCG for gLite 3.0[2] middleware was implemented by so called Experiment Software Installation toolkit, and is described as following:

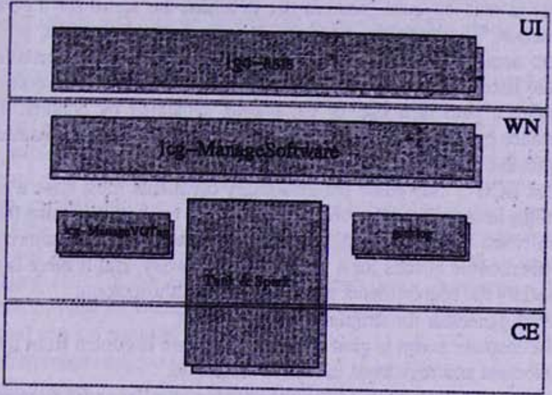


Figure 1: Multi layer structure of the general implementation of the Experiment Software Installation toolkit

The top layer, lcg-asis [3], is the friendly interface that can be used by the ESM to install/validate or remove software on one or more sites that are allowed for the VO the ESM belongs to.
The middle layer, installed on each WN, lcg-ManageSoftware [3] is the toolkit that steers any Application Software management process.
Finally, the low level layer is constituted by lcg-ManageVOTag[3], which is used to publish tags on the Information System. Tank&Spark is used to propagate software in case the site doesn't provide shared disk space.

The ESM must prepare a tarball(s) containing the software to be installed and some scripts to install, validate, and remove the software. No constraints are imposed to the names of these scripts and tarballs. Afterwards, the ESM invokes (from the UI) the lcg-asis script by providing the name of the tarballs and/or the name of the scripts to be used for installing, removing or validating the software.

The lcg-asis script performs basically the following actions:

- Uploads the specified tarball on the grid (lcg-cr command).
- Loops over suitable sites for the given VO matching the requirements of the ESM.
- For each site checks whether other installation (or validation or removal) processes are running on the site.
- For each site composes the JDL with adequate parameters.
- For each site submits the installation job that invokes on the target WN the script lcg-ManageSoftware, which performs the installation process.

In the new release of gLite middleware, version 3.1 the Experiment Software Installation toolkit (lcg-asis and lcg-ManageSoftware) are not supported anymore and there is not any alternative script for that. Below, in Chapter 3, the "Custom-Installer" scripts - the replacement for Experiment Software Installation toolkit is introduced.

3. The "Custom-Installer" script

The "Custom-Installer" script contains the following modules:

- Configuration file

The ESM needs to edit the configuration file to adjust the script for a certain site and VO specific information. The variables needs to be edited are: VO name, preferred or default Storage Element name, Software repository path (the software to be installed), and the path to the file where listed all Computing Elements on which the software should be installed. This can be either a path to a file, or CE names, separated by comma. The list could be obtained in advance by CEAT script (also introduced in this paper in Chapter 3.1).

- Authorization and Authentication module

According to the LCG-2 user guide the grid proxy certificate must have a remaining time of at least 20 minutes in order the edg-job-* commands to work. Also, since this is not a regular grid job, but software installation/validation job the one should be authorized with a role of "lcgadmin". This module checks for a valid lcgadmin proxy, and if there is not any available or it does not satisfy the requirements prompts for the authorization.

- Installer script generator for template

In this phase the installer script is generated, and software is copied from the local repository to the storage element and registered in replica catalogue.

- Job Description Language (JDL) file generator for installer script from template

A JDL per Computing Element is generated. It is here mentioned that the job should run on specified CE.

- Job management module

Running the job(s), and activating real-time monitor subscript. The numbers of jobs are equal to the number of available computing elements (which are either added by ESM, or generated by CEAT script in advance)

- Software tags management module

As it was earlier described in the Chapter 2, during all phases of the installation (which are: installation is in progress, installation is done, validation is in process and finally validated), the software tags needs to be published in the Information System (IS). This module adds tags on all phases. During the installation the temporary tag format is *VO-*

$vo-$sw-in_preparation – where $vo is virtual organization name, $sw is the software name, that currently is being installed.
After the installation the temporary tag is removed, and a new tag: VO-$vo-$sw-to-be-validated is added.
After the validation the "to-be-validated" tag is removed, and the final tag is added to the Information System.

This script is currently used for VO "hermes" and VO "ilc" at DESY for the experiment software installation and validation, and is available at [4]. With minor changes it could be easily adjusted for other VOs.

4. Service Availability Monitor

Service Availability Monitoring (SAM) [5] is a monitoring tool conceived as a Site Functional Test (SFT) [6] extension, that was used to monitor the performance of a Grid site. SAM re-uses most of the SFT code, with the only exception of the web service API framework. The heart of the testing and publishing system is an Oracle database, which is connected to the SAM sensors (both the ones integrated into the framework and the standalone ones) through the Tomcat web services. These services include Grid Operations Centre Database (GOC DB) [7] query and publishing web services implemented in Java or using servlets. In addition to that, the Oracle DB interacts with the top level BDII using a Python script.
The monitoring procedures are carried out using sensors which regularly publish the results for all monitored service nodes and that are integrated within the SAM framework mainly in two different approaches:

- Creating a standalone sensor (daemon or cron job) which tests all essential service nodes and publishes the results to SAM using publishing scripts or web service APIs
- Integrating test script (sensor) with the SAM framework.

The test workflow goes across two stages called test phase and publishing phase. During the test phase, simple sensors perform the tests, then publish the results, while more complex sensors carry out basic tests, publish some results and submit some long term test jobs. The publishing phase is optional for the simplest sensors. During this stage, more complicated sensors check the status of long term test jobs and publishing the results. Within the SAM testing infrastructure a special package called SAM Submission Framework has been developed. The latter allows collecting service sensors and providing them a uniform way to:

- discover list of service nodes to test
- schedule and execute tests
- publish results to the central SAM Oracle database

Within the SEE-GRID project a Standalone version of SAM client, that can be used to test sites without the need for a dedicated SAM server has been developed [8].

5. CEAT script: Computing Element Availability Tests

Originally the CEAT script was meant as addendum to Custom-Installer script. The main purpose was to ensure the availability of CE before experiment software installation. However it can be also used as a standalone monitoring tool for maintaining an up to date list of currently available CEs for certain VO and publish it on a Web.
CEAT script is using lightweight and slightly modified, standalone version of SAM (service availability monitor), which is a modified version of SAM client than can be used to test sites

without the need for a dedicated SAM server. From its source same.sfx file is built, which is a self-extracting shell archive (tar.gz file wrapped in a shell script that extracts it and runs the predefined command). In this case the predefined command is same/same.sh, a small shell script that in turn calls SAM testjob sensor. This will run the tests but instead of publishing them to the SAM server, it will create an HTML page with a report.
The source is available at: [9], so a new test can be added, but the same.sfx should be rebuilt.
Current tests are:

- *CE-sft-lcg-rm-cr* - Copy and register a short text file to the default SE using lcg-cr command. Retrieve list of replicas with lcg-lr command.
- *CE-sft-lcg-rm-cp* - Copy the file registered in test CE-sft-lcg-rm-cr to the WN using lcg-cp command.
- *CE-sft-lcg-rm-rep* - Replicate the file registered in test CE-sft-lcg-rm-cr to the chosen "central" SE using lcg-rep command.
- *CE-sft-lcg-rm-del* - Delete replicas of all the files registered in previous tests using lcg-del command.
- *CE-sft-vo-swdir* - Detect the software directory for selected VO. If the variable is set and directory VO_${VO}_SW_DIR exists then the test is successful.
- *CE-sft-vo-tag* - Check tag management for selected VO using lcg-ManageVOTag command. If there are no installed software for selected VO test is in warning status. If command retrieves the list of installed software then the test is successful.

There would be real-time and up to date information if CE is available and ready for software installation after running this series of test.
Tests are run on CEs of defined VO, after completion the output is being analyze. There could be 3 states of test-job:

1. Aborted – when the job is aborted
2. Failed – when job reaches the WN but one of tests has failure, so the SW could not be installed on the entire CE
3. OK – when jobs reaches WN, and all tests passed

In general terms, the block diagram for the controller can be drawn as in Fig. 2.

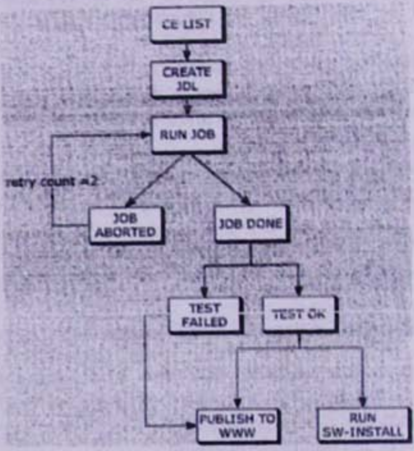
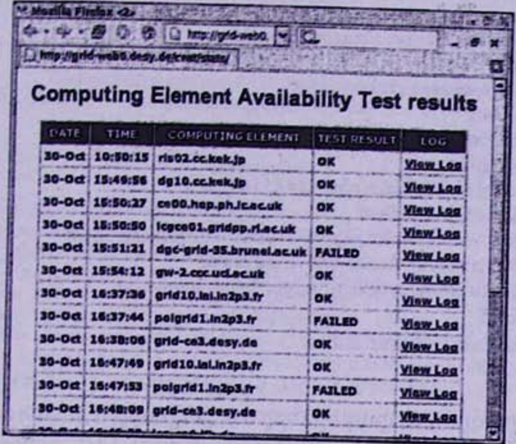


Figure 2. CEAT block diagram

After analyzing the output, CEs are sorted to two files, FAIL.list (if the tests are failed), and OK.list (if tests are passed). The OK.list can be "feed" to Custom-Installer script, to automatically install SW. Also, OK, and failed jobs are published in the web: [10] with detailed information on which tests are failed.



Computing Element Availability Test results

DATE	TIME	COMPUTING ELEMENT	TEST RESULT	LOG
30-Oct	10:50:15	ris02.cc.kek.jp	OK	View Log
30-Oct	15:49:56	dg10.cc.kek.jp	OK	View Log
30-Oct	15:50:27	ce00.hep.ph.ic.ac.uk	OK	View Log
30-Oct	15:50:50	lcgce01.gridpp.rl.ac.uk	OK	View Log
30-Oct	15:51:21	dgc-grid-35.brunel.ac.uk	FAILED	View Log
30-Oct	15:54:12	gw-2.ccc.ucl.ac.uk	OK	View Log
30-Oct	16:37:36	grid10.lal.in2p3.fr	OK	View Log
30-Oct	16:37:44	polgrid1.in2p3.fr	FAILED	View Log
30-Oct	16:38:06	grid-ce3.desy.de	OK	View Log
30-Oct	16:47:49	grid10.lal.in2p3.fr	OK	View Log
30-Oct	16:47:53	polgrid1.in2p3.fr	FAILED	View Log
30-Oct	16:48:09	grid-ce3.desy.de	OK	View Log

Figure 3. Screenshot of CEAT web report.

6. Conclusions

In this paper two scripts were introduced: "Custom-Installer" script the replacement for Experiment Software Installation toolkit; and CEAT script - Computing Element Availability Tests for real-time monitoring of Computing Elements and publishing reports in the web.
Since in the new release of gLite middleware, version 3.1 the Experiment Software Installation toolkit (lcg-asis and lcg-ManageSoftware) are not supported anymore and there is not any alternative script for that, the introduced scripts are useful replacements which covering that gap. It has to be mentioned that "Custom-Installer" script is currently used for VO "hermes" and VO "ilc" at DESY for the experiment software installation and validation and CEAT script is used as a one of main monitoring tools at DESY.

7. References

[1] Tank and Spark *http://grid-deployment.web.cern.ch/grid deployment/eis/docs/internal/chep04/SW_Installation.pdf*

[2] Lightweight middleware for Grid Computing *http://glite.web.cern.ch/glite/*

[3] Experiment Software Installation in LCG-2 *http://grid-deployment.web.cern.ch/grid-deployment/eis/docs/ExpSwInstall/sw-install.pdf*

[4] "Custom Installer" script *http://grid-web0.desy.de/ceat/custom-installer.tgz*

[5] Service Availability Monitor (SAM) *http://sam-docs.web.cern.ch/sam-docs/*

[6] Site Functional Tests *http://goc.grid.sinica.edu.tw/gocwiki/Site_Functional_Tests*

[7] Grid Operations Centre *http://grid-it.cnaf.infn.it/index.php?id=853&type=1*

[8] Standalone version of SAM client *http://wiki.egee-see.org/index.php/SEE-GRID_standalone_SAM*

[9] Slightly modified version of standalone SAM client *http://grid-web0.desy.de/ceat/lightweight-ssam_source.tgz*

[10] CEAT at DESY *http://grid-web0.desy.de/ceat/stats/*

Հատուկ ծրագրային փաթեթների տեղադրումը և հաշվիչ սարքերի ֆունկցիոնալության հավաստիացումը LHC գրիդում

Վ. Մկոյան

Ամփոփում

Հոդվածում նկարագրված է հատուկ ծրագրային փաթեթների տեղադրման ընթացքը LHC գրիդում: Մշակված են ծրագրային փաթեթներ տեղադրման ընթացքի ավտոմատացման և հաշվիչ սարքերի ֆունկցիոնալության հավաստիացման և վերահսկման համար: