BUILDING THE LCG : FROM MIDDLEWARE INTEGRATION TO PRODUCTION QUALITY SOFTWARE

Marco Serra (INFN-CERN),Louis Poncet (LAL-IN2P3), Jean-Philippe Baud (CERN),
David Smith (CERN), Piera Bettini (INFN-CERN), Frederique Chollet (LAPP-IN2P3),
Gilbert Grosdidier (LAL-IN2P3) , Maarten Litmaath (CERN),
Carlos Osuna (University of Granada), Di Qing (Academia Sinica), Zdenek Sekera (CERN).

Abstract

In the last few years grid software (middleware) has become available from various sources. However, there are no standards yet which allow for an easy integration of different services. Moreover, middleware was produced by different projects with the main goal of developing new functionalities rather than production quality software. In the context of the LHC Computing Grid project (LCG) an integration, testing and certification activity is ongoing which aims at producing a stable coherent set of services. Here we report on the processes employed to produce the LCG middleware release and related activities, including the infrastructures used, the activities needed to integrate the various components and the certification process. Our certification process consists of a continuous iterative cycle that also involves feedback from the LCG production system and input from the software providers. The architecture of the LCG middleware is described in Ref.[3], including additional components developed by LCG to improve scalability and performance. Other associated activities include packaging for deployment, porting to different platforms, debugging and patching of the software. Functionality and stress tests are performed via a large testbed infrastructure that allows for benchmarking of different configurations.

CREATION OF THE CANDIDATE RELEASE

The Certification and Testing team obtains software from different sources, currently mainly from the EDG and VDT projects . We also develop middleware components that are urgently needed and cannot (yet) be obtained from other sources. We also use external software maintained by other groups. To create a release a strict release cycle is followed as described below and shown in Figure 1.

Define desired features.

Before beginning the integration of new parts of the middleware we define the scope of the next release. The requirements are defined by experiments and by bugfixes supplied by our internal debugging team and external groups. At any step in the certification process bugs can be reported and corrected before the release. We try to create incremental releases without too many changes at a time. The target is to permit experiments to adapt their code easily to new releases. This method permits us to have a new release about once per month. Each release is made available to the Infrastructure team, who decide if the release is useful to deploy. By this method the LCG functionality steadily increases without long pauses.

What are the different parts of the middleware?

The middleware of a grid is generic. We need job management, data management, an information system, accounting and monitoring. These functionalities will be provided by EGEE and VDT software. EGEE will replace the EDG software being used today. The target of the middleware is to be compatible with the most important batch systems and storage systems already installed in computing centres. We also have to provide open source solutions for centres without any batch or storage system. The VDT distribution of Globus still is the heart of the system. We collaborate with the VDT team, reporting and fixing bugs. We also try to increase the interoperability with other grids. A good connection between LCG and other grids would be an elegant way to increase the potential storage capacity and calculation power for each participating grid.

CERTIFICATION OF THE RELEASE ITSELF.

We use a certification test-bed supporting various configurations at the same time. A certification process is defined by five different series of tests:

- Installation.
- Basic functionality.
- Certification Matrix (intensive test).
- Special tests for new functionality, not yet included in the certification matrix).
- Experiment-specific tests (on the Experiments Integration test-bed).

All supported configurations are certified by the LCG Certification and Testing group.

Batch systems :

- OpenPBS.

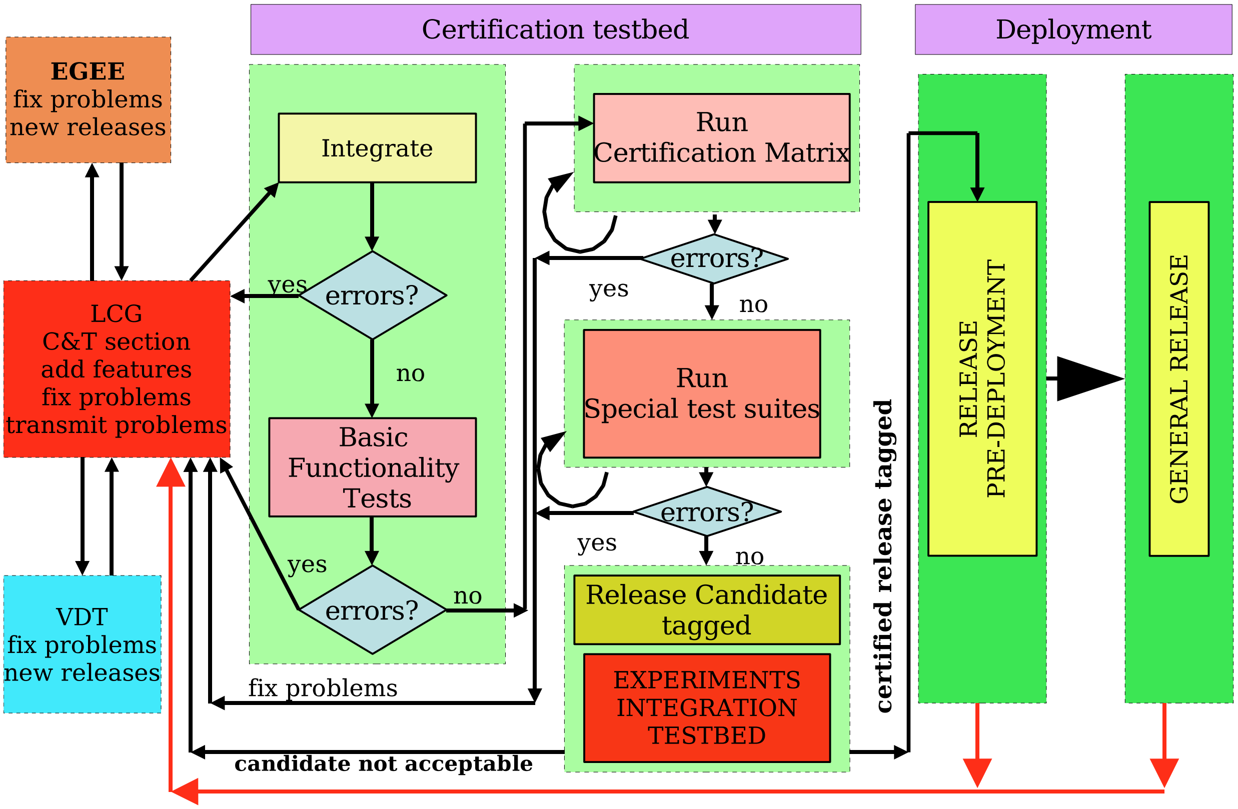

Figure 1: Certification process.

- LSF.
- Condor.
- Torque
- ...

Storage systems:

- Classic SE (simple GridFTP server).
- dCache SRM.
- CASTOR SRM.
- ...

Installation test

Currently the main platform still is RedHat 7.3, but we are moving toward other platforms, in particular Scientific Linux 3, compatible with RedHat Enterprise Linux 3. The RedHat 7.3 certification testbed is mostly installed through the LCFGng fabric management system. The LCFGng configuration files serve as a starting point for the manual and Quattor installation methods described later in this document. We ensure we can both upgrade nodes from the previous release and reinstall them from scratch with the new release. Currently the Scientific Linux 3 testbed is installed manually. We use the manual installation guide for the previous release and provide a document with the differences for the new configuration. The package lists are managed in cpp format, like for LCFGng. We create a set of scripts to download and install packages or manage an APT repository. As soon as possible we will switch to the Quattor fabric management system (http://quattor.web.cern.ch/) and share our experience with EGEE. LCFGng cannot easily be ported to Scientific Linux and cannot cooperate with other fabric management systems already in use in computing centres.

The certification matrix.

Every night 22 series of tests are run on the test-bed.

40% of them are functionality tests for various types of job submission, data registration, replication, copy, and removal. 60% are stress tests to assess the limits of the system. Today those tests are simple job storms, 3 types of data storms, a very intensive GridFTP storm, and MatchMaking storms for 2 protocols (letting Resource Brokers match jobs to sites, with constraints on input or output data). The certification matrix is regularly extended to test new functionalities and stress them. We may include special tests created by the developers of the software or by the Integration team. A web page is automatically produced showing the results of the tests. Examples can be found on the testing group website Ref. [4]. The test matrix is not fixed, but evolves with the middleware.

Tagging of the release candidate

When the certification matrix results confirm we have reached the targets for a new release, we tag the middleware and configuration template versions that have just been certified on the test-bed. A release is a set of LCFGng files listing all RPMs to be used, configuration file templates for LCFGng modules, and examples. On our website we create download pages for the RPMs.

DISTRIBUTION PROCESS

All the middleware is distributed through CVS and the Web. Our website contains all the required information for deployment of the middleware.

- The release page provides CVS commands to download all required files for each release.
- Download pages provide all required packages for all releases.
- Documentation pages provide information on how to use our tools in general. Specific documentation for the middleware installation is in the tag given by the release page.
- The CVS repository allows browsing of RPM lists, configuration templates, and source code.
- Problems can be reported in the CERN Savannah tracking system for bugs, patches and tasks.
- There are links for grid users, experiments, and computing centre system administrators.

We provide a ready-to-use solution for centres without any tools to do fabric management. Today we are using LCFGng for RedHat 7.3 and we will use Quattor for Scientific Linux. We also provide documentation for manual installation, to permit centres to adapt the installation to their local fabric management software.

CREATION OF THE FINAL RELEASE

Experiments Integration test-bed.

The Experiments Integration team installs the candidate release on the experiments integration test-bed, thereby first testing the installation procedure itself. Next they test their latest tools for the installation of experiment software and invite the experiments to test the release candidate. Experiments certify that their own software runs with the new middleware, and report bugs that must be corrected before the new release is put on the production system. Only on the production system will the middleware be subjected to fully realistic conditions, and any problems observed are reported back to the Certification and Testing team for correction in future releases.

Creation of the final release

After corrections, if any, the candidate release is given to the Infrastructure team, who will remove all CERN-specific settings and packages, and adapt the middleware to the experiments' requirements (VO settings). Then they create manual installation scripts, documentation for LCFGng and manual installations. They test themselves the manual installation procedure.

MANAGEMENT OF THE PROJECT

Bug reporting, task management and software maintenance

We use the CERN Savannah tools and central CVS service. Bug reports are tracked by the bug report system of Savannah. We manage all our middleware tasks with the Savannah task manager. The distribution is done through a deployment CVS server supplying the LCFGng configuration templates and various scripts. The use of the central CVS service and tracking system allows the certification and deployment teams to focus on issues with the middleware itself.

OUR MAIN PROBLEMS

What is the boundary between the system and the middleware ?

This is an important problem for the distribution and the certification. The middleware has important dependencies on standard tools and libraries provided with the operating system. Do we have to provide a ready-to-use OS or a list of required packages? Either solution has shortcomings. An LCG worker node may not be fully dedicated to LCG, but used for other purposes as well. LCG software may then require a version of a particular library that is incompatible with other software that needs to be able to run. We provide a ready-to-use configuration for LCFGng with operating system and standard tools included besides the middleware. A manual installation is supported providing only the middleware, in which case it is the responsibility of the computing centre to ensure compatibility with their OS and standard tools.

How to support multiple architectures and operating systems on the same grid?

A binary that runs on RedHat 7.3 is not sure to run on Scientific Linux 3, and for completely different architectures or operating systems it is not even an option to take a binary not specifically compiled for the worker node where it shall run. The binary could be compiled on the fly, but that requires the presence of a compiler and set of libraries and tools that have been certified by the experiments. Such tools could all be installed in a dedicated area per experiment, but that may pose maintenance problems. These issues are under investigation.

How to convince people to change their code to fit grid requirements ?

Generic interfaces to grid services are still being developed. In the meantime complex and obsolete mechanisms have to remain supported. Experiments are not yet prepared for the new types of failures that can occur on a grid, compared to local batch systems and storage facilities. For example, they may submit jobs assuming that each job will at most run once, whereas on a wide area grid it is hard to guarantee such desired behavior.

CONCLUSION

The LCG grid deployment group tries to merge all requirements in incremental releases, steadily creating a very large grid able to satisfy the data processing needs of the experiments. Today LCG has around 8 000 CPUs connected.

REFERENCES

[1] Grid Deployment website
http://cern.ch/grid-deployment/

[2] LCG website
http://cern.ch/lcg/

[3] LCG-2 architecture
https://edms.cern.ch/cedar/plsql/\
doc.info?document_id=498079

[4] Testing process results
http://cern.ch/grid-deployment/cgi-bin/\
index.cgi?var=tstg/validation