Proc UK e-Science All Hands Meeting 2003, © EPSRC Sept 2003, ISBN 1-904425-11-9

Test and Deployment of ICENI, An Integrated Grid Middleware on the UK e-Science Grid

Nathalie Furmento William Lee Steven Newhouse John Darlington

London e-Science Centre, Imperial College London, South Kensington Campus, London SW7 2AZ, UK
Email: lesc-staff@doc.ic.ac.uk

Abstract

This paper presents a test and deployment infrastructure that has been developed to validate the ICENI Grid Middleware. Services have been developed to monitor the different steps in the life of the middleware from its compilation to its execution. We will show how these services can be composed to provide a full validation of the architecture and the functionalities of ICENI. We also present a MDS to ICENI Gateway used to register machines currently available in the MDS Server as ICENI launcher services.

1 Introduction

The Imperial College e-Science Networked Infrastructure - ICENI - is a service oriented/integrated Grid middleware that provides an augmented component programming model to aid the application developer in constructing Grid applications, and an execution infrastructure that exposes compute, storage and software resources as services with defined conditions of when and by whom these resources may be used. It utilises open and extensible XML schemas to encapsulate meta-data relating to resource capability, service availability and application behaviour.

Testing and validating a distributed infrastructure is not an easy task. In order to validate our development activity, and to facilitate the installation and deployment of ICENI, a build service as well as a deployment service have been provided. These services can be composed together to perform a complete test of ICENI from the initial compilation of the source code to the instantiation of the different services that it provides.

2 Test and Deployment Infrastructure

2.1 Build process

The ICENI software base is checked out of the CVS archive and automatically built every night. Some tests are performed locally on each module of the framework. These tests allow the developer to evaluate the individual functionalities of the different modules of ICENI. These unit tests are supplemented by a global test, that starts all the required services to submit a simple test application. That consists of a Private Administrative Domain, a Public Computational Community, a Scheduler Resource, a Launcher Resource, a Software Resource Repository and the Software Resources that will be composed to form the test application [6, 4]. Finally the application is submitted to the middleware and the output of its execution is evaluated.

On success, the API documentation and the binary modules are uploaded onto the LeSC web server. These two automatic processes send an email reporting success or failure to a developer's mailing list allowing us to keep track of the progress of the ICENI code.

2.2 Deployment process

Within the ICENI distribution, a script is provided to automatically install and deploy the latest version of the ICENI framework onto a local resource. This script will:

- Download the latest version of the binary modules.
- Update the configuration files (e.g. information on the private administrative domain or the public computational communities to connect to).

- Start the required resources.

This deployment process can be started automatically at frequent intervals (by using crontab), and can either start its own private administrative domain with specific resources, or start resources to connect to an already running administrative domain. It requires a CVS access to the ICENI repositories, as well as a valid X.509 certificate and a proxy allowing a Globus access to the machine `saturn.icpc.doc.ic.ac.uk`, and the `globus-url-copy` command line tool.

2.3 Composition

The build process and the deployment process of ICENI can be made available as services. These two services can then be composed to build and deploy ICENI, the deployment is triggered only when the build service executes successfully and can be run on different resources. The deployment service is being extended as an Application Deployment Service that will deploy any ICENI Application onto a resource. In the same way, an Application Status Service can be used to monitor and test an application, and reporting the status of the service (and therefore the application) into a registry (for example a MDS server).

Figure 1 shows how the build service and deployment service can be composed. The arrows represent the transfer of the ICENI modules that are pushed on the LeSC web server by the build service when the ICENI code compiles and runs its tests successfully, and then pulled down to the resources by the deployment service to install and run an ICENI application.

These mechanisms have been used to start a grid connecting resources in London, Cardiff, and Newcastle. This demonstration was shown at the London e-Science Centre Open Day with a Launcher Service in Cardiff, and another one in Newcastle, both connecting to a public Computational Community running in London. The applications submitted to the ICENI middleware were scheduled to run between these two launchers, having some components running in Newcastle, some others running in Cardiff.

Currently, build and deployment services are running on the following machines:

Build service is running on a Windows NT machine and a Solaris box inside the London e-Science Centre.

Deployment service is running on a Solaris box inside the London e-Science Centre, as well as on a Linux box and an IRIX box that are part of the Level 2 Grid activities. A deployment service involving machines from different e-Science centres is also running.

An Application Status Service has been implemented to report in a database the results of these different services. This database is publicly available on the LeSC web server [2] and can be queried according to different criteria such as the operating system on which the service was run, or the name of the test that was run. Extracts from the contents of this database are shown in Tables 1 and 2.

3 MDS to ICENI Gateway

ICENI provides a MDS [1] gateway that is going to connect to a specific MDS server to retrieve all the resources, by executing for example the following request:

```
grid-info-search -x -h giis.lesc.ic.ac.uk
 -b "mds-vo-name=lesc,o=grid"
 -s sub "(|(object=class=MdsHost)
           (objectclass=ComputingElement))"
```

For each host returned, a launcher is started with information such as the clock speed, or the memory available (information that will be used in a latter phase by the scheduler to efficiently map components on resources). The contract associated to the new launcher service will either be based on the list of authorised users provided by the resource in the MDS server, or if that information is not available, a default contract will be created.

That gateway can be used as part of the ICENI deployment to start as many launcher services as the number of machines currently available when the deployment is run. That allows to dynamically change the nature of the test, and check the capability of the scheduler to react to a different number of launcher with different configurations. Figure 2 shows machines from different ETF sites connecting to an UK e-Science Private Administrative Domain to form a federation of launcher services.

However the number of launchers started through this technique can be very high. We are looking at starting specific launchers according to the type of the host; several hosts can be grouped and managed by the same launcher such as a Condor launcher [5] or a Globus launcher [3].

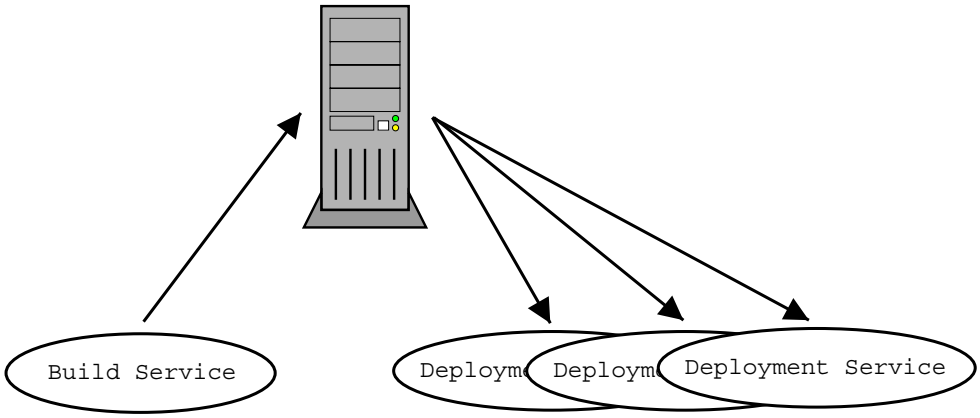

Figure 1: Composition of Build and Deployment Services

Hostname	Arch & OS	Date	Build result	Tests		
				Run	Failures	Errors
rhea.lesc.doc.ic.ac.uk	SunOS 5.8	2003-08-07 — 02:00	Build Successful	29	0	0
LeSCDemo2.doc.ic.ac.uk	WindowsNT 1	2003-08-07 — 00:02	Build Successful	29	0	0
rhea.lesc.doc.ic.ac.uk	SunOS 5.8	2003-08-06 — 02:00	Build Successful	29	0	0
LeSCDemo2.doc.ic.ac.uk	WindowsNT 1	2003-08-06 — 00:01	Build Successful	29	0	0
rhea.lesc.doc.ic.ac.uk	SunOS 5.8	2003-08-05 — 02:00	Build Successful	29	0	0

Table 1: Result of Build Services

Hostname	Arch & OS	Date	Release	Name	Result
bouscat.cs.cf.ac.uk	Linux 2.4.2	2003-08-07 — 11:23	-	Hello World	Pass
rhea.lesc.doc.ic.ac.uk	SunOS 5.8	2003-08-07 — 05:48	-	Hello World	Pass
bouscat.cs.cf.ac.uk	Linux 2.4.2	2003-08-07 — 04:23	-	Hello World	Fail, No output file
rhea.lesc.doc.ic.ac.uk	SunOS 5.8	2003-08-06 — 05:46	-	Hello World	Pass

Table 2: Result of Deployment Services

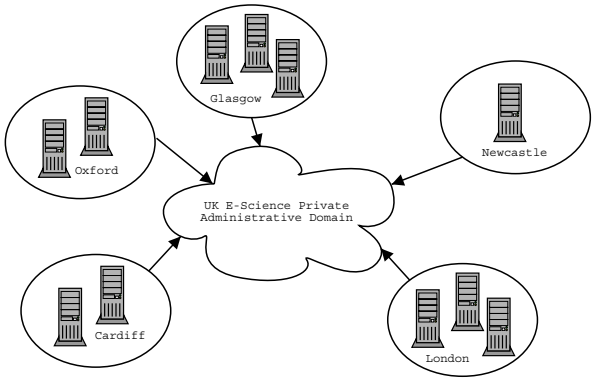

Figure 2: UK e-Science Administrative Domain

4 Conclusion

Testing and validating the behaviour of a distributed service oriented architecture presents many challenges. The build, test & deployment cycle within the ICENI nightly process represents an initial exploration of these issues. Such an infrastructure is essential in building the resilient, robust and reliable S.O.A needed to support the UK e-Science Grid.

References

[1] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman. Grid Information Services for Distributed Resource Sharing. In *Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10)*. IEEE Press, August 2001.

[2] The London e Science Centre. ICENI dashboard. `http://www.lesc.ic.ac.uk/iceni/dashboard.jsp`, 2003.

[3] I. Foster and C. Kesselman. The Globus Project: A Status Report. In *IPPS/SPDP '98 Heterogeneous Computing Workshop*, pages 4–18, 1998.

[4] A. Mayer, S. McGough, M. Gulamali, L. Young, J. Stanton, S. Newhouse, and J. Darlington. Meaning and Behaviour in Grid Oriented Components. In *3rd International Workshop on Grid Computing, Grid 2002*, volume 2536 of *Lecture Notes in Computer Science*, Baltimore, USA, November 2002.

[5] Condor Team. Condor Project Homepage. `http://www.cs.wisc.edu/condor`.

[6] L. Young, S. McGough, S. Newhouse, and J. Darlington. Scheduling Architecture and Algorithms within the ICENI Grid Middleware. In *All Hands Meeting, UK e-Science Program*, Nottingham, 2003.