The Second-Generation Grid-Ireland Deployment Architecture with Quattor Centralised Fabric Management

Stephen Childs, Brian Coghlan, David O'Callaghan, Geoff Quigley and John Walsh
{childss, coghlan, ocalladw, gquigle, walshj1}@cs.tcd.ie

Department of Computer Science, Trinity College Dublin, Ireland

The Grid-Ireland infrastructure is unusual among national grids in that all Grid service nodes at our eighteen sites are remotely managed from a central Operations Centre (OpsCentre) which controls the configuration of the Grid middleware and the standard OS components. The Grid middleware is validated at the OpsCentre before being deployed to the remote sites: any troublesome issues can be resolved before the new software is rolled out. As all aspects of the OS configuration are controlled from a central location, the OpsCentre can ensure that all of the infrastructure machines are up-to-date with security patches and that their configuration is consistent.

It is clear that such an architecture requires a well-thought out management system: ad-hoc methods that may suffice for a single site do not scale to an entire national Grid comprising many sites. A comprehensive fabric management system driven from a central configuration database is essential. In the first generation of our deployment architecture we used LCFGng [1] to manage the configuration of Grid service nodes running RedHat 7.3. After a period of evaluation, we recently migrated our entire infrastructure to use the Quattor [5] fabric management system. We now present the results of this evaluation and our experiences with the transition.

The goal of Grid computing is to create a highly robust and stable infrastructure that enables scientists to access globally distributed computing and storage resources in a straightforward way. The Grid will have succeeded when it becomes a largely invisible network service that researchers can take for granted as they do Internet connectivity. We believe that a centrally managed architecture like ours enables a realistic roadmap to creating this infrastructure.

1 Introduction

In most geographically dispersed grid infrastructures, both the service nodes and compute clusters at the member sites are managed by the system administrators at those sites. The model adopted by Grid-Ireland is somewhat different: a skilled central team manages service nodes at all sites, based on configurations stored in a central repository, site admins manage local clusters, and the central team supports them in integrating these clusters with the Grid infrastructure.

There are a number of advantages to this approach. Firstly, the core infrastructure is kept consistent. Secondly, thorough validation of new software is performed centrally before deployment to the sites. Thirdly, uniform monitoring tools can be deployed, and finally, a uniform upgrade cycle can be implemented across the infrastructure. We believe that this is a more sustainable model in the long term; it makes Grid infrastructure a resource for all and liberates local researchers to focus on science. Also, a centrally-managed infrastructure is a good fit for a non-competitive funding model as applied to National Research and Education Networks (NRENs), paving the way for Grid infrastructure management to become an NREN service in the future.

At the time of writing (January 2006), Grid-Ireland includes eighteen sites located at institutions throughout the island of Ireland. Each site hosts a standard set of service nodes running LCG middleware (Computing Element, Storage Element, User Interface and test Worker Node). Further details of the Grid-Ireland architecture are found in [2].

The Grid infrastructure is remotely managed from the Operations Centre based at Trinity College Dublin, where approximately 4.5 FTE staff are allocated to manage daily operations. Grid-Ireland also runs central services to support national VOs. These services include a information server (BDII), a resource broker, a MyProxy server, a VOMS server and a replica location server. Auxiliary services such as a Certification Authority, web server and software repository are also managed by Grid-Ireland.

2 Deployment model

All configuration details and software packages are stored on a repository server located at the Operations Centre. For any new release or update, we follow these steps (shown in Figure 1):

1. Updated software is downloaded and any configuration changes are made and committed to CVS
2. This new configuration is validated in our test environment (a comprehensive replica of the national infrastructure)
3. If the validation is successful, the configuration is tagged in CVS
4. This tagged set of profiles and any new software packages are deployed transactionally to repositories at the sites from whence real deployment takes place. This ensures efficiency and resilience.

Conceptually, there are three levels of configuration that influence a grid site: Grid-wide configuration common to all sites worldwide, region-wide configuration common to all sites in a particular country or federation, and site-level configuration specific to a particular site. Our configuration database is structured in a way that reflects this hierarchy, allowing configuration changes to be made efficiently at the appropriate level. Standard Grid-wide configuration templates are imported unchanged and are only overridden when necessary.

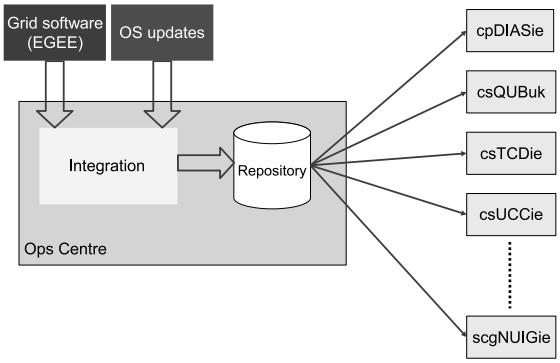

Fig. 1: Grid-Ireland deployment model

3 Evaluation of fabric management tools

In order to successfully implement a centrally-managed Grid infrastructure, it is essential to choose fabric management tools that provide sufficient control over the configuration of managed nodes. As part of the preparation for upgrading our infrastructure, we identified the requirements our model imposed and used them to evaluate the various solutions available.

The three main contenders we evaluated were LCFG, Kickstart/YAIM and Quattor. LCFG is a fabric management tool that was developed at the University of Edinburgh and used extensively for configuring Grid middleware (the first generation of the Grid-Ireland deployment architecture used LCFG). In the summer of 2005 it was phased out of LCG Grid middleware releases. LCFG is composed of an installation server and client software. The installation server hosts configuration templates which are compiled into XML profiles and a repository of software packages for installation. The client software includes a daemon for retrieving profiles and a suite of objects that configure OS and grid middleware components based on parameters read from the profiles.

Kickstart is a tool for installing machines from scratch over the network. It uses PXE network booting and TFTP to retrieve a kernel image and configuration file from a server, and then installs the OS via HTTP or NFS using the parameters specified in the configuration file. YAIM is a set of shell scripts that install and configure Grid middleware components according to parameters specified in a site configuration file. While Kickstart and YAIM are not an officially-supported combination, they are often used together (e.g. at the majority of UK grid sites). YAIM can be run automatically as part of Kickstart's post-installation phase, allowing Grid nodes to be installed without intervention.

Quattor is an ambitious project for developing a complete fabric management system. It includes its own language, PAN, [4] for describing the configuration of nodes, and a large suite of components for managing various OS components. Its design is similar to that of LCFG, but it was specifically designed to overcome

some of LCFG's shortcomings. A full Quattor system includes a configuration database server for managing versioned templates (and serving them to clients over HTTP) and a software repository for managing installation packages. However, Quattor can be profitably used without installing the full set of servers.

3.1 Requirements

In order to help us decide on the best solution for managing the Grid-Ireland infrastructure, we listed the requirements for the fabric management system. The first requirement is that the system should allow for flexible, incremental configuration of all significant operating system components. It is not sufficient to allow configuration at install-time: the daily operation of the sites necessitates a fine-grained, continuous control over OS paramters. The tool must also provide a good coverage of the various OS components: any residual manual configuration steps result in an unmanageable system when running a large number of geographically distributed nodes.

A second requirement is for validation of configurations before deployment. This means ensuring that profiles compile correctly before deploying to site installation servers and that there are no typing errors in syntactically correct profiles. These are ambitious goals, and are not fully met by any configuration system in widespread use today. However, there are degrees to which these goals can be met. The use of a typed language in configuration templates allows scope for building in validation checks on variable values, and a flexible compilation scheme that allows changes to be verified on the user's workstation allows for better testing of changes.

A third requirement was for a declarative package management system. By declarative, we mean that for each node, a list of packages to be installed is generated from a configuration file, and the software package management client ensures that only these packages are installed. This contrasts with update-based systems such as apt or yum, which have no real concept of a desired package list for a particular node, and simply upgrade packages to a later version.

A fourth requirement is good integration with Grid middleware. This has two dimensions: that there are configuration templates available for standard Grid middleware components, and that there is a pool of users within the Grid community to provide support and develop new components.

A fifth requirement is support for building a hierarchical configuration that will allow multiple levels of configuration to be defined (in our case three levels are necessary: grid-wide, country-wide and per-site).

3.2 Comparing against requirements

Table 1 summarises the requirements for a centrally managed infrastructure. As you can see, LCFG met almost all of our requirements in the past, although it had a few frustrating limitations. For instance, there were some significant components (e.g. PBS scheduler) that LCFG didn't configure, which necessitated manual post-configuration of notes. Another problem was that profiles could

only be compiled on the installation server where they would be deployed (due to dependencies on the hostname of the machine). This made it almost impossible to verify that configuration changes would produce profiles that compiled. The deciding factor against future use of LCFG was lack of support within the Grid community: it would have been impractical to singlehandedly port and maintain LCFG components for Grid configuration.

	LCFG	**KS/yaim**	**Quattor**
OS configuration	yes	partial	yes
Ongoing (not just install time)	yes	no	yes
Comprehensive configuration	partial	partial	yes
Central validation of profiles	no	n/a	yes
Declarative package management	yes	no	yes
Integration with Grid middleware	phased out	yes	yes
Active user community on Grids	no	yes	yes
Multiple levels of configuration	yes	no	yes

Tab. 1: Requirements on a fabric management system

Kickstart and YAIM, while well-suited to the installation of a single site, do not meet our requirements. Kickstart provides for install-time configuration of OS components, not day-to-day reconfiguration. In many environments this is not a problem as minor changes can be effected using scripts and SSH keys, and major changes can be effected by reinstallation. This is not a viable option for us: remote reinstallation of nodes at distance can be a perilous operation. In any case, we require our configuration to be located in a single database rather than dispersed through a set of scripts. In addition, this solution provides no support for validation: YAIM uses simple key-value pairs with no typing. Finally, in this model packages are usually managed with apt or yum, which do not provide declarative package management.

Quattor is the solution that best meets our current and future requirements. It provides a large set of components for configuring OS parameters, and the use of a typed language and the ability to compile profiles anywhere make it easier to detect configuration errors before deployment. Declarative package management allows for strict control of package installation on target nodes, ensuring uniformity. Finally, the set of templates distributed by the Quattor Working Group provides good coverage of the standard LCG Grid middleware components, and there is a significant number of sites now using Quattor, thus creating a mutually supportive user community.

4 Configuration hierarchy

The organisation of our configuration database reflects the three-level hierarchy describe earlier, and has been designed to reduce the number of edits required to implement configuration changes. In our model, it would not make sense to edit configuration files for each site whenever a high-level configuration change is necessary. Figure 2 shows our configuration structures for LCFGng and Quattor. For LCFG, the standard configuration came from the Grid De-

```
common/
|-- rpmlist
`-- source
sites/
|-- cpDIASie
|   |-- cluster-nodes
|   |   |-- gridwn01
|   |   |-- ...
|   |   `-- gridwn07
|   |-- gateway-nodes
|   |   |-- gridgate
|   |   |-- ...
|   |   `-- gridui
|   |-- rpmlist
|   |   |-- BASE-rpm
|   |   |-- ...
|   |   `-- local-rpm.h
|   `-- source
|       |-- CE-extras.h
|       |-- ...
|       `-- site-cfg.h
VOs/
|-- CosmoGrid
|-- ...
`-- gitut
```

```
shared/
|-- LCG-2.6.0-15/
|-- VOs
|   |-- ...
|   `-- webcom
|-- grid-ireland
|   |-- profiles
|   |   |-- ...
|   |   `-- profile_gi_wn.tpl
|   |-- rpmlist
|   |   |-- ...
|   |   `-- pro_sl305_default.tpl
|   `-- source
|       |-- ...
|       `-- pro_vo_egee_users.tpl
`-- repository
    |-- ...
    `-- repository_grid_ireland_i386_xen.tpl
sites/
|-- cpDIASie
|   |-- profiles
|   |   |-- ...
|   |   `-- profile_gridwn07.tpl
|   `-- source
|       |-- ...
|       `-- pro_site_ui_users.tpl
```

Fig. 2: LCFG and Quattor configuration hierarchies

ployment Team at CERN; for Quattor, it comes from the LCG Quattor Working Group [6].

The basic concept is the same in both cases: we include a particular release of the standard distribution templates (e.g. `shared/LCG-2.6.0-15`) unmodified, and override variables as necessary at a lower level. We have chosen this approach rather than patching the standard distribution because it allows our configuration to be validated against a reference release. Also, for small configuration changes, a new release of the standard templates can be dropped into place and new profiles generated without any changes to the Grid-Ireland-specific configuration.

We also aggregate configuration that is common across our infrastructure (in `shared/grid-ireland`). We can rapidly make changes to all of our sites by editing a single file. Individual node profiles inherit configuration from three levels (grid, regional, and site). Using the ant configuration developed by Cal Loomis [7], include paths are generated that pull in the correct templates for a site. The template for a particular node is normally a simple inclusion of a standard template for that node type.

5 Grid-Ireland deployment architecture using Quattor

Quattor is a modular toolsuite comprising clients and servers for managing different aspects of nodes' installation, configuration and maintenance. All configuration is described in PAN, a typed language with support for hierarchical structures, and is implemented by component objects that manipulate the various OS components on the client nodes.

The main servers in a full Quattor installation are the Configuration Database (CDB) and software repository (SWrep). The CDB stores and compiles the configuration templates needed to describe the system, providing version control and validation facilities. We decided not to use the CDB, opting to use our existing CVS system for version control and `ant` to compile the profiles. We made this decision because of limitations in the CDB control interface, difficulties with configuring multiple sites, and the extra management overhead of maintaining the CDB servers. Similar considerations applied with the SWrep server. We didn't feel that the features offered by this server (access control, replication, multi-platform support) justified the overhead of learning how to configure and manage it. Our "repositories" are simply directories on a web server, with packages grouped according to their function or origin (e.g. OS, middleware, CA).

Our Quattor deployment model is essentially the same as the LCFG-based model. The most significant difference is that admins can now edit profiles on their own desktop machines, and verify compilation before committing their changes to CVS. Site software repositories are replicated using rsync, and the templates describing the repositories' contents are generated from the central repository and the hostnames of the site repositories are substituted in when compiling node profiles.

6 Migration from LCFG to Quattor

We first modified our transactional deployment tool to support Quattor, and then used this tool to migrate the nodes themselves (including the install servers). The Grid-Ireland transactional deployment (TD) tool [3] allows for new configurations to be rolled out transactionally to multiple sites with a single action. The original version was based on LCFG, and so modifications were necessary to make it compatible with Quattor. These did not require significant changes to the architecture of the tool. In fact the code was simplified: as Quattor allows off-site profile compilation, a single compilation could be executed on the TD server, and the results propagated to the relevant sites.

The first node we reconfigured was the install server at each site: we ran the TD tool to rsync across the new packages and profiles. We then reconfigured the service nodes. First, we added the Quattor client RPMs (RedHat 7.3 versions) to the LCFG package list and ensured that LCFG installed them. We then ran a script on each node that configured the Quattor configuration manager for the first time, downloaded the new profile and ran the Quattor package manager to upgrade the packages. This procedure proved to be remarkably smooth, if

somewhat slow (an upgrade typically took around an hour, although this was on Xen virtual machines with only 200 MB of memory).

7 Conclusion

Our most recent upgrade (December 2005) provided a stern test of our deployment architecture, as it included three major transitions: an OS upgrade from RedHat 7.3 to Scientific Linux 3.0.5, a fabric management system upgrade from LCFGng to Quattor, and a Grid middleware upgrade from LCG 2.4.0 to LCG 2.6.0. To prepare, we carried out extensive certification testing within our TestGrid environment over a period of three months. We were then able to complete the live upgrade of national servers and sites within a couple of weeks.

We believe that our experience validates the use of centralised fabric management within a hierarchical deployment architecture for a national grid; the relative ease of transitioning to a new fabric management system has confirmed our approach. Based on our evaluation, we would recommend Quattor for use in managing Grid sites, with the following qualifications: i) as the learning curve is steep, it is only worth migrating to Quattor if you are managing multiple sites or a very large single site; ii) some care is necessary in deciding which modules of the Quattor tool-suite are appropriate for your configuration.

We would encourage national and regional grids to consider adopting a centrally-managed Grid infrastructure. The benefits include an efficient use of personnel, a greater uniformity of site configuration, faster upgrade times and easier deployment of region-specific software (including monitoring tools).

This work was supported by the European Union under the EGEE project. We would like to thank the Quattor developers for their help.

References

1. Paul Anderson and Alastair Scobie. LCFG — the Next Generation. In *UKUUG Winter Conference*. UKUUG, 2002.
2. B.A. Coghlan, J. Walsh, and D. O'Callaghan. Grid-Ireland Deployment Architecture. In *Advances in Grid Computing - EGC 2005*, LNCS3470, Amsterdam, The Netherlands, February 2005. Springer.
3. B.A. Coghlan, J. Walsh, G. Quigley, D. O'Callaghan, S. Childs, and E. Kenny. Principles of Transactional Grid Deployment. In *Advances in Grid Computing - EGC 2005*, LNCS3470, pages 88–97, Amsterdam, February 2005. Springer.
4. Lionel Cons and Piotr Poznanski. Pan: A high-level configuration language. In *LISA 2002: Sixteenth Systems Administration Conference*, pages 83–98. Usenix, 2002.
5. R. Garcia Leiva et al. Quattor: Tools and Techniques for the Configuration, Installation and Management of Large-Scale Grid Computing Fabrics. *Journal of Grid Computing*, 2(4), 2004.
6. Charles Loomis. Installing LCG Software with Quattor. https://svn.lal.in2p3.fr/LCG/QWG/web/lcg-with-quattor.pdf, February 2005.
7. Charles Loomis. Replacing the CDB with Subversion and Ant. https://svn.lal.in2p3.fr/LCG/QWG/web/svn-ant-db.pdf, April 2005.