Configuration management and monitoring of the middleware at GridKa

Dimitri Nilsen, Pavel Weber
Karlsruhe Institute of Technology Steinbuch Centre for Computing
Hermann-von-Helmholtz-Platz 1, D-76344 Eggenstein-Leopoldshafen
E-mail: dimitri.nilsen@kit.edu pavel.weber@kit.edu

Abstract. GridKa is a computing centre located in Karlsruhe. It serves as Tier-1 centre for the four LHC experiments and also provides its computing and storage resources for other non-LHC HEP and astroparticle physics experiments as well as for several communities of the German Grid Initiative D-Grid. The middleware layer at GridKa comprises three main flavours: Globus, gLite and UNICORE. This layer provides the access to the several clusters, according to the requirements of the corresponding communities. The heterogeneous structure of middleware resources and services requires their effective administration for stable and sustainable operation of the whole computing centre. In the presentation the overview of the middleware system at GridKa is given with focus on the configuration management and monitoring. These are the crucial components of the administration task for the system with high-availability setup. The various configuration tools used at GridKa, their benefits and limitations as well as developed automation procedures of the configuration management will be discussed. The overview of the monitoring system which evaluates the information delivered by central and local grid information services and provides status and detailed diagnostics for the middleware services is presented.

1. Introduction
The sustainable operation of the middleware system is crucial for the stable operation of the whole data centre. The reliable configuration management and effective multi-level monitoring system help to organize the effective administration of the heterogeneous middleware resources and efficiently detect possible malfunctions of the system. The implementation of the configuration management at GridKa for middleware services based on Puppet [1] tool provides clear view for the administration workflow. This workflow includes the integration, deployment and testing of the middleware services as well as their technical support and monitoring during the operation.

2. Middleware at GridKa centre
In the following the main flavours of middleware used at GridKa and general middleware configuration are briefly introduced.

2.1. Middleware flavours at GridKa
The main middleware flavours deployed at GridKa are depicted in Figure 1. gLite [2] is the main middleware used in world-wide LHC Computing Grid (WLCG) and one of the major
middleware flavours deployed at GridKa. It provides a uniform framework for collaborative usage of heterogeneous resources distributed around the world at many data centres. The main gLite components are:

- Grid security.
- Information services.
- Workload management.
- Job submission framework.
- User interface.
- Data services.

As GridKa serves as Tier-1 centre for the four LHC experiments, the gLite middleware plays the primary role in the middleware infrastructure at the data centre. Currently, the gLite 3.1 and gLite 3.2 versions are being used at the GridKa. Typically the middleware servers carry one gLite service per host.

UNICORE [3] is another middleware hosted at GridKa and KIT. UNICORE consists of five components:

- Gateway.
- Registry.
- Unicorex.
- Xuudn.
- Tsi.

These components could be combined on one physical machine. This possible configuration makes it important to describe each component as standalone to provide more efficient management of the UNICORE stack. Also Globus Toolkit 4/5 [4] gateways provide access to KIT clusters.

All flavours of middleware mentioned above are currently combined in the software platform developed by European Middleware Initiative (EMI) [5].

2.2. Middleware servers
Middleware components are unique for each middleware flavour. But at the job submission level packages like "pbs util" are the same at all gateways, independent of middleware installed. The distribution of the middleware flavours among the physical servers is not uniform. The modular structure of configuration management provides intelligent deployment for the components of all middleware flavours and simplify changes and maintenance of machines. As was mentioned in the previous section, most of the GridKa servers carry the gLite services and just a few UNICORE and Globus services. The exact distribution of the middleware services among hosts is given in Table 1.

3. Automatization of middleware deployment
The middleware deployment procedure basically consists of four steps, as depicted in Figure 2. The first step is an installation of a raw operating system (OS). Currently, the main OS used in WLCG environment is a Scientific Linux 5/64bit. The installation of OS is done with a
Preboot Execution Environment (PXE) and a CluClo (Linux cloning tool). All OS images and configuration files are located at one administrative server and can be easily modified or cloned.

The second step is the preparation of the OS for the middleware deployment. It includes the installation of the packages and the configuration layers like ssh, ntpd, networking, logging and puppet. All these packages are combined in a single rpm and installed via yum or other packaging tools during deployment. The goal of this step is to prepare the OS environment for the middleware deployment using configuration system.

The middleware deployment step manages all middleware specific components. Puppet was chosen as a most effective tool for configuration management by complex middleware system with a huge number of modules. The desired state of the host or service is described in so-called Puppet manifests. Based on these manifests, Puppet performs configuration changes for each host. It installs, configures and starts the services in one configuration run. After the deployment run the machine is ready to be taken into production.

The last step starts the live cycle management that constantly updates the machine with configuration parameters during service life. The typical examples for these parameters are user mapping, certificates etc. Puppet also provides a possibility to run scripts and to monitor the system processes, which is used to prevent failures and to improve the availability of the service.

All results are published to the central databases and can be used for further investigation in case of system malfunctions.

![Diagram of middleware deployment steps](image)

**Figure 2.** Middleware deployment steps.
Figure 3. Schematic view of the middleware configuration management framework at GridKa.

4. Middleware configuration management

The schematic view of the middleware configuration management framework developed and implemented at GridKa is depicted in Figure 3. The framework consists of several blocks:

- Subversion [6] repository.
- Puppet master server.
- External data source.
- Middleware stack.
- Dashboard.

The central and complete configuration management system for heterogeneous middleware resources at GridKa can simplify the daily administration work make it more transparent and better documented. On the other hand, any small typing error in the configuration can affect the whole middleware system.

Therefore the configuration management system should be safe to any change done in configuration. This means that a faulty change in configuration or a buggy update for the given service will not affect the current state of the system. This requirement can be achieved by using the release management, which provides the possibility to prepare different versions of the configuration. The new configuration is thoroughly tested before its implementation in production instance. As a basis for the release management the Subversion repository is used. It stores the configuration data required for middleware services as well as configuration modules for Puppet server itself. After the changes committed to Subversion the post-commit hook mechanism updates the puppet master server if it is required.

Each middleware configuration release stored in Subversion is a set of three versions: development release, test release and production release. Any change in the configuration is implemented in the development release. When all changes are implemented, the test release is built. The configuration changes implemented in test release are applied to pre-production system. The pre-production system consists of several servers, which are used explicitly for the testing purposes. After the new configuration has been successfully tested at pre-production system, the changes in configuration propagate to production release and are applied to production system.

The Puppet master server is the central place of the configuration system where currently the Puppet tool version 2.7 is running. Puppet server holds the configuration data and the rules for all middleware hosts. It also accepts additional external configuration data from external sources.
data source. The typical example of the data coming from external source are the certificates with a limited lifetime issued by Certificate Authorities.

During each configuration run, Puppet obtains the status from each host provided by the Facter [7] and performs the required configuration based on description of the host or the service in the corresponding Puppet modules. The results of the configuration runs are stored in the Foreman dashboard database.

Foreman dashboard comprises the real time status information on hosts as well as inventory information for each host, history of configuration runs for each hosts. The dashboard board acts as a central place for monitoring of the current configuration on all middleware hosts. In the next section the dashboard and its usage will be described in more detail.

5. Monitoring

For the monitoring of the puppet performance and the host states the Foreman dashboard is used. In Figure 4, the table view of a few middleware hosts, their configuration state and additional system information is shown. Foreman also builds a system inventory (based on Facter) and provides real time information about host status based on Puppet reports and history of configuration runs. An example of the page with complete information about the configured host is depicted in Figure 5. The notification mechanism in log files is configured to report any troubles at the middleware stack, e.g. by errors in puppet runs. With an integrated dashboard, the administrator has entire overview of the middleware configuration.

6. Conclusions

In this paper, the configuration management and monitoring of middleware resources at GridKa data centre are discussed. The implemented configuration management based on Puppet tool together with the release management provides the efficient and reliable administration of the middleware servers. The developed configuration system allows to perform full control and accounting on any change in the middleware resources and roll back to the stable configuration state in case faulty configuration change. The monitoring dashboard provides the entire overview of the middleware system and allows to perform the detailed diagnostics for the middleware services.
References
[1] Puppet, http://puppetlabs.com/
[2] gLite, http://glite.web.cern.ch/glite/
[3] UNICORE http://www.unicore.eu/
[4] Globus, http://www.globus.org/
[5] EMI, http://www.eu-emi.eu/home
[6] Subversion, http://subversion.apache.org/
[7] Facter, http://projects.puppetlabs.com/projects/facter