JRA1-Integration

Enabling Grids for E-sciencE

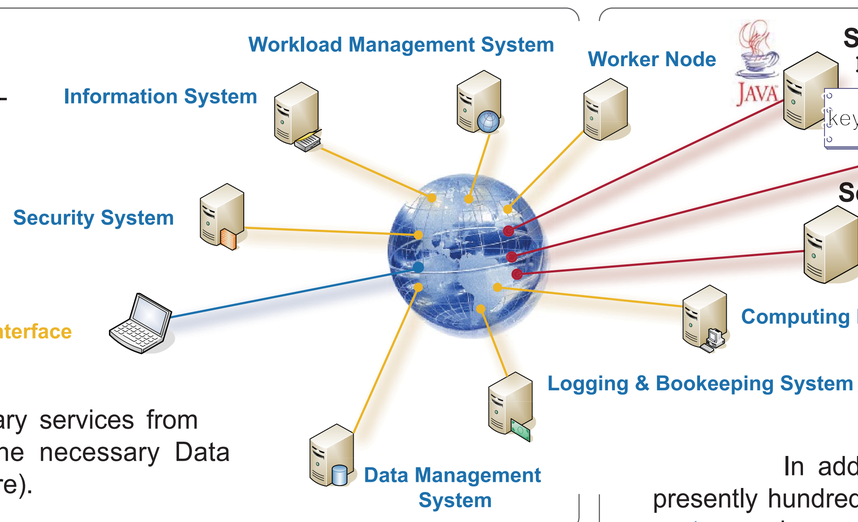
An XML-based configuration and management system for the gLite middleware

Joachim Flammer, Alberto Di Meglio, Guillermo Diez-Andino Sancho
Robert Harakaly, Lanxin Ma, Marian Zurek (CERN)

What is gLite ?

gLite is the next generation middleware for grid computing. Born from the collaborative efforts of more than 80 people in twelve different academic and industrial research centers as part of the EGEE Project, gLite provides a bleeding-edge, best-of-breed framework for building grid applications tapping into the power of distributed computing and storage resources across the Internet.

Currently, gLite is composed of more than 25 different services, spanning the full range of necessary services from Information and Workload Management services, the necessary Data Mangement services up to Security services (see picture).



Configuration and management challenges

Currently, gLite is composed of more than 25 different services, implemented in **different languages**, using **different technologies** and all coming with **individual configuration** needs. Most of the services have existed before gLite was written and come already with their **own configuration** or **logging systems**.

In addition gLite can be run in **multiple operational scenarios,** supports presently hundreds of configuration options and is run in a widely **distributed environment** spread over different sites and continents.

Past experience has shown that configuration and management are one of the biggest challenges of such a distributed system.

Our goal is to make the system as user friendly and manageable as possible. Operational questions are:

- How to configure a set of 1.000 distributed machines ?
- How to find out about the configuration of a specific service ?
- How to detect misconfigurations ?
- How to get log messages of a service ?
- How to restart a service without disturbing the other services ?

Implementation of a common configuration and management system is crucial

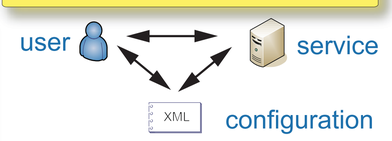
Design requirements

Modularity
Separate the installation from the configuration.

Guidance
Support user in understanding and selecting the correct value.

Documentation
Couple documentation directly to configuration.

Homogeneity
Unique way of storing configuration information for all services.



Common interfaces
Common interfaces to access configuration & control of service.

Flexibility
Allow user to arrange configuration according to their needs.

Security
Secure access to management and configuration interface.

Remote Instrumentation
Enable homogeneous remote management of services.

gLite configuration system

The gLite configuration system is trying to solve the first part of the challenges by basing itself on a common set of XML configuration files. While still being human readable, XML allows for a proper arrangement as well as automated treatment of the configuration information. The main features of the gLite configuration system are:

Separation of installation & configuration

- Each service has its own deployment module for installation
- Each service deployment module comes with its configuration script to configure and start the service

Common gLite wide configuration layer

- Storage of entire configuration information in xml
- Hiding underlying service specific configuration
- Common set of configuration files and file format

Configuration parameter classification

- Parameters are divided into three categories:
  - User-defined parameters: values have to be set by users
  - Advanced parameters: have always valid default values - can be changed by advanced user
  - System parameters: have always valid default values - should normally not be changed

```
<config>
    <xi:include href="http://glite.cern.ch/lxb30.xml" xpointer="//config"/>

    <vo name="EGEE">
        ...
    </vo>

    <node name="lxb1234.cern.ch">
        ...
    </node>

    <instance name="io-server" service="rgma-servicetool">
        ...
    </instance>

    <parameters>
        <!-- User defined parameter -->
        <voms.host.name
            description="VOMS server host name. [Type: string]
                         Example: kuiken.nikhef.nl"
            value="changeme"/>

        <!-- Advanced parameter -->
        <condor.daemonlist
            description="Array of the condor daemons to start."
            <value>MASTER</value>
            <value>SCHEDD</value>
        </condor.daemonlist>

        <!-- System parameter -->
        <webapp.location
            value="${GLITE_LOCATION}/share/webapps/"/>
    </parameters>
</config>
```

Site configuration

- Possibility of including other configuration files
- Configuring services from central point (web server)
- Combination of central and local configuration

Arranging configuration information according to user need

- Centralizing common information for one node or for one Virtual Organization
- Overwriting / Reusing of configuration information

Multiple instance handling

- Support for configuration of multiple instances
- Support for arrays

Proper documentation

- xml structure couples description, example values etc. to each configuration parameter
- Automatic documentation creation through XSLT transformation possible

Configuration scripts

- Set of platform independent, service specific and common configuration scripts (e.g. for tomcat)
- Implemented in python
- CLI to configure / start / stop / get status of a service

Configuration management system

In order to solve the second part of the challenges, we are working on a prototype to implement a gLite wide unique service instrumentation interface for direct configuration and management together with a central management service.

Management

Service Management

Central management
GliteManager

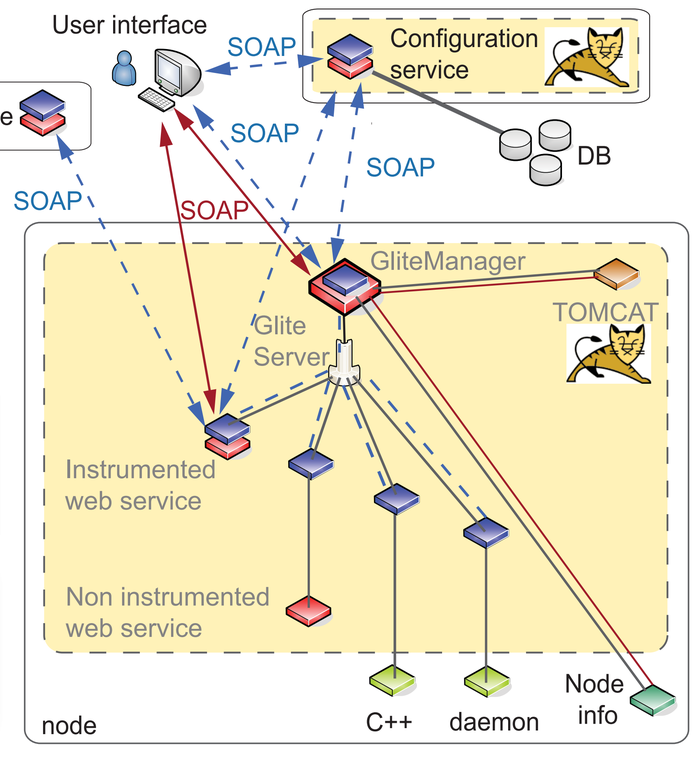
Tasks
- Collects information from different sources(node/tomcat/services)
- Enables management of different sources (node/tomcat/services)

Technology
- Java web based on JMX
- Inside communication via JMX
- Outside communication via SOAP
- Follows CIM standard

Service management
GliteService

Tasks
- Configuration of service
- Management of service sources
- Stub for non Java web services/ non instrumented services

Technology
- Implements common & extendable GliteService management interface
- Outside communication via SOAP



Configuration

Configuration exchange

Configuration service
- Configuration is served by one ore more configuration services to the gLite services
- Configuration service can have several backends DB, files, etc.

User
- User can define/query/update configuration via a user interface
- User interface contacts the configuration service

gLite service
- Service gets configuration from configuration service
- Build in interface for configuration in each service
- Service can get configuration from other gLite services