EGEE

R-GMA Service Tool

Document identifier:	**EGEE-JRA1-TEC-servicetool-manual**
Date:	**May 11, 2006**
Activity:	**JRA1: Middleware Engineering and Integration (UK Cluster)**
Document status:	**DRAFT**
Document link:	

Abstract: This document describes the R-GMA service publication tool

1 INTRODUCTION

The R-GMA servicetool enables a site to publish information about the services it knows about and their current status.

The tool consists of a daemon that regularly monitors configuration files containing information about the services a site has installed. At regular intervals, this information is published to the GlueService, GlueServiceData, GlueServiceAssociation and GlueServiceOwner tables. These files also contains information about a script to run to obtain status information for each service and a frequency to run this script. These scripts are run by the daemon with the specified frequency and the results inserted into the GlueServiceStatus table.

2 INSTALLATION

INSTALLATION FROM RPMS

1. Install R-GMA client library for Java

```
rpm -Uvh glite-rgma-base-4.x.x-1.noarch.rpm
rpm -Uvh glite-rgma-api-java-4.x.x-1.noarch.rpm
rpm -Uvh glite-rgma-stubs-servlet-java-4.x.x-1.noarch.rpm
```

2. Configure R-GMA client library

   To run R-GMA programs the environment variable `RGMA_HOME` must be set (alternatively if the variable `GLITE_LOCATION` is set this will be used instead). The main R-GMA setup script must then be run to configure the locations of the local server, schema and registry.

```
export RGMA_HOME=/opt/glite
/opt/glite/share/rgma/scripts/rgma-setup.py
```

3. Install Servicetool

```
rpm -Uvh glite-rgma-servicetool-4.x.x-1.noarch.rpm
```

3 CONFIGURATION

Servicetool uses a configuration directory which contains the following files:

servicetool.conf A `name=value` properties file. Currently the only recognized property is `site` which specifies the site name for all published services. The default site name is `changeme.invalid` which must be changed before services can be published. A site name should be a valid DNS entry, e.g. `lxb2029.cern.ch`, and must be unique to the site.

services A directory containing service files. Each file describes one service and has the following format:

```
Name=server.example.rl.ac.uk_org.example.ExampleService.
ReadableName=An example service
Endpoint=http://example.rl.ac.uk/serviceEndpoint
Type=org.example.ExampleService
Version=1.2.3
PublishInterval=600
StatusScript=/usr/sbin/check_service
StatusInterval=60
WSDL=http://example.rl.ac.uk/service?WSDL
Semantics=http://example.rl.ac.uk/service/semantics.html

VO=VOName1
VO=VOName2

AssociatedService=AssocServiceName
AssociatedService=AnotherAssocServiceName

RandomDataKey=Value
AnotherRandomDataKey=AnotherValue
```

Services may be published by adding a valid configuration file to the `services` directory.

SERVICE PROPERTIES

A service has the following properties which are specified in the service configuration file:

`Name` Globally unique name for the service. In general this should have the form `hostname_type` where `hostname` is the name of the host the service is running on, and `type` is the service type, as defined by the `Type` property. If the service is specific to a particular Virtual Organisation, the VO name may be inserted between the hostname and the type (separated by the _ character).

`ReadableName` Human-readable name for the service. Need not be globally unique. This property is optional - if it is absent the unique service name will be used instead.

`Endpoint` URL to contact the service at.

`Type` The service type. This should be uniquely defined for each service and should be a reversed DNS entry owner by the creator of the service type.

`Version` Service version in the form major.minor.patch, e.g. 1.2.3

`PublishInterval` How often to publish the service details, in seconds

`StatusScript` Script to run when determining service status. This script should return an exit code of 0 to indicate the service is OK, other values should indicate an error. The first line of the standard output should be a brief message describing the service status (e.g. 'Accepting connections')

`StatusInterval` How often to check and publish service status, in seconds.

`WSDL` URL of a WSDL document for the service (Leave blank if the service has no WSDL)

`Semantics` URL of a document containing a detailed description of the service and how it should be used.

`VO` Name of a Virtual Organization this service is considered part of. This property is optional and may also be used multiple times for services that are part of more than one VO.

`AssociatedService` Name of another service associated with this one. This property is optional and may also be used multiple times.

Properties specified in the service configuration file that do **not** match any of the above properties will be treated as service-specific key/value data and inserted into the GlueServiceData table appropriately. **N.B. this means you cannot have service data with a key that is the same as one of the standard service properties.** This is a deliberate limitation to avoid confusion between standard and service-specific properties.

4 COMMAND USAGE

DAEMON

```
rgma-servicetool-daemon (--config <config dir>)
                        (--check-interval <seconds>)
```

`--config` specifies the location of the configuration directory. If this parameter is absent, the default configuration directory will be used, which is:

```
$RGMA_HOME/etc/rgma-servicetool
```

`--check-interval` specifies how often to check for changes in the configuration, in seconds.

An init.d script is provided for Unix systems. The servicetool daemon may be controlled by:

```
/etc/init.d/rgma-servicetool <start|stop|restart|status>
```

This daemon uses the default configuration directory. Logging output is written to the file:

```
/var/log/glite/rgma-servicetool.log
```

Logging configuration may be changed by editing the file:

```
$RGMA_HOME/etc/rgma-servicetool/log4j.properties
```