UNICORE REGISTRY MANUAL

UNICORE Team

Document Version:	6.6.0
Component Version:	6.6.0
Date:	04 04 2013

This work is co-funded by the EC EMI project under the FP7 Collaborative Projects Grant Agreement Nr. INFSO-RI-261611.

Contents

The UNICORE Registry server provides information about available services to clients and other services. It is a specially configured UNICORE/X server, so please make sure to refer to the general UNICORE/X manual as well.

Multiple UNICORE/X sites can share a registry, greatly simplifying the use of your UNICORE Grid. Since such a registry is vital to the functioning of a UNICORE Grid, you can have more than one.

For more information about UNICORE visit http://www.unicore.eu.

1 Installation

1.1 Prerequisites

To run the Registry, you need the SUN or OpenJDK Java 6 (JRE or SDK). If not installed on your system, you can download it from http://java.oracle.com

The UNICORE Registry has been most extensively tested on Linux-like systems, but runs on Windows and MacOS as well.

Please note that

- to integrate into secure production environments, you will need access to a certificate authority and generate certificates for all your UNICORE servers.
- to make your UNICORE servers accessible outside of your firewalls, you should setup and configure a UNICORE Gateway.

1.2 A note on paths

The Registry can be installed either from a Linux package (i.e. RPM or deb), from a tar.gz or even from the UNICORE core server bundle package.

Note
Using the Linux packages, you can install only a single Registry instance per machine (without manual changes).

The following table gives an overview of the file locations for both tar.gz and Linux packages.

Table 1: Directory Layout

Name in this manual	tar.gz, zip	rpm	Description
CONF	<basedir>/conf/	/etc/unicore/registry	Config files
LIB	<basedir>/lib/	/usr/share/unicore/registry/lib	Jar libraries
LOG	<basedir>/log/	/var/log/unicore/registry/	Log files
BIN	<basedir>/bin/	/usr/sbin/	Start/stop scripts
—	—	/etc/init.d/unicore-registry	Init script

1.3 Updating a 6.5.x installation

Starting with the 6.6.0 release, UNICORE components use a new security library called CANL (Common Authentication Library), which offers many additional options related to keystores, truststores, certificate revocation etc. We have also harmonised many settings, making the property files easier to read and maintain.

Consequently many settings in the uas.config and wsrflite.xml files have new property keys, and new ones have been added. To update an existing installation, some effort will be required.

The general update procedure is presented below, with possible variations:

1. Stop the old Registry server and remove persisted data.
2. Update the server package. This step mostly applies for RPM/DEB managed installations. For Quickstart installation it is advised to place the new Registry to a separate directory and replace the existing *lib* folder by the new one.
3. Port the configuration of the pre-6.6.0 server to the new syntax. It can be done in two ways:
   - Manually by applying all old values to the new template configuration. There are quite a few properties to be ported so this requires several minutes of work. The advantage of manual porting is that the new template files with new options and updated comments are used. Note: for RPM installations the new files will be named `*.rpmnew`.
   - Automatically, using UNICORE configurator. You have to install the `unicore-configurator` package. It is included in Quickstart, for distributions install simply run appropriate command: `yum install unicore-configurator` or `apt-get install unicore-configurator`. Using the configurator you can update the old files automatically:
     - For RPM/DEB installations it is enough to run `unicore-config-update.py registry`.

- For Quickstart, specify the old config directory with the `-c` option: `./unicore-config-update.py -c REGISTRY_CONFIG_DIR registry`. Carefully read the output of the program - there can be some problems reported. Option `-h` provides help, with information about the usage: how to perform dry run, how to recover files from backup etc.

4. Delete the persisted service data.
5. Start the newly installed Registry.
6. Verify the log files and fix any problems reported.

2 Registry configuration

A Registry is running in a "normal" UNICORE/X container, however, you should use a dedicated UNICORE/X instance for the Registry, making sure no other services are running.

Thus, most of the UNICORE/X documentation regarding access control, keystores, etc also applies to the Registry. Please, make sure to read the UNICORE/X documentation as well.

2.1 Services configuration (CONF/wsrflite.xml)

Apart from hostname, port, and other properties, the wsrflite.xml file should contain the following service definitions.

```
<service name="ServiceGroupEntry" wsrf="true" persistent="true">
  <interface class="de.fzj.unicore.wsrflite.xmlbeans.sg. ↩
      ServiceGroupEntry" />
  <implementation class="de.fzj.unicore.wsrflite.xmlbeans. ↩
      registry.RegistryEntryHomeImpl"/>
</service>

<service name="Registry" wsrf="true" persistent="true">
  <interface class="de.fzj.unicore.wsrflite.xmlbeans.sg.Registry ↩
      " />
  <implementation class="de.fzj.unicore.wsrflite.xmlbeans. ↩
      registry.RegistryHomeImpl"/>
</service>
```

As soon these services are active, the container will operate as a registry.

2.2 Registry configuration (CONF/uas.config)

You can specify some properties, in addition to the usual configuration (attribute source settings, etc), to control certain aspects of the registry.

```
#switch off UDP multicast advertisement of the registry
container.registry.globalAdvertise=false
```

2.3 Starting and stopping

The registry is started and stopped like any other UNICORE/X container using the scripts in the "bin" folder. If running multiple UNICORE/X servers on the same host, make sure to check the container port, and possibly the JMX port (in the start script).

2.4 Enabling access control

To enable access control, set in CONF/uas.config

```
container.security.accesscontrol.Registry=true
```

This will check the security policy (CONF/xacml2Policies/*.xml) for each request. By default, this policy allows to add entries only for callers with the role "server".

If using an XUUDB or other attribute source, you will need to add the certificates / DNs of all servers wishing to publish into the registry as having the role "server". Please check the UNICORE/X documentation on how to do that.

2.5 Gateway configuration

Usually, you'll need to add an entry to the gateway's site list file (connections.properties) that points to your registry server. Another option is to use dynamic gateway registration. In the following, we assume the Registry VSite is named "REGISTRY".

2.6 Vsite configuration

To use a specific registry, configure the address of the registry in uas.config. The simplest is to use auto-discover the registry in your network using multicast (group 228.5.6.7 port 7700). Then you just have to set a single property in CONF/uas.config:

```
#switch on use of external registry
container.externalregistry.use=true
```

The entries in the global registry are updated at a specified interval. To control this interval, edit a property in CONF/wsrflite.xml:

```
<!-- default termination time for registry entries in seconds -->
<property name="container.wsrf.sg.defaulttermtime" value="1800"/>
```

2.7 Manual configuration

In case you cannot or will not use UDP multicast, you can also specify a fixed registry address. This is done in uas.config:

```
#switch on use of external registry
container.externalregistry.use=true

#switch off autodiscovery
container.externalregistry.autodiscover=false


# manually provide url and epr of the external registry
# url:
container.externalregistry.url=https://localhost:8080/REGISTRY/ ↩
    services/Registry?res=default_registry

# optionally you can have more registries
container.externalregistry.url.2=https://localhost:8080/REGISTRY- ↩
    BACKUP/services/Registry?res=default_registry
```

2.8 Client configuration

Make sure your clients use the global, shared registry.