GRIP: the Evolution of UNICORE Towards a Service Oriented Grid

Roger Menday, Philipp Wieder
Forschungszentrum Jülich, Germany
{r.menday, ph.wieder}@fz-juelich.de
Outline

- Introduction
- Web/Grid Services
- Project GRIP
- UNICORE
- Architectural evolution
- Summary
Motivation

**Today:** “I need a Grid Service” -> “I deploy a Grid System”

**Future:** “I need a Grid Service” -> “I deploy a Grid Service”

- **Usage:** Deployment time vs. usage time ratio is unacceptable (days vs. Hours?)
- **Boundaries:** Virtual Organisations (VOs), within VOs (different CAs, ...), Grid systems (including protocols, ...)
- **Market:** Move from batch model to a service-market-oriented approach (Web Services, OGSA, Business Grid, ...)

Next Generation Grid Wishlist

- Transparent and reliable
- Open to wide user and provider communities
- Pervasive and ubiquitous
- Secure and provide trust across multiple administrative domains
- Easy to use and to program
- Persistent
- Based on standards for software and protocols
- Person-centric
- Scalable
- Easy to configure and manage

So does a Service Oriented approach help to fulfil this?
Integration of Web Services: Lots of talk about paradigm shift, but maybe it's true ...

- Currently: custom protocols, layered often closed architectures.
- With WS/GS: Distributed, open architecture - *Standardised Grid protocols, in a web services framework*

Challenges

Service detection, orchestration, dynamic federation, semantic grid ... and still the old: languages, interoperability, ...
Web/Grid Services

*Designed for loosely-coupled distributed computing*

**WSDL** – service description, **SOAP** – service invocation

*Extensible SOAP Header carrying supporting information*


**Routing** : WS-Routing, **Community security** : SAML, XKMS

**OGSI**

- *stateful* web services
- OGSI provides **service data** – remote instance variables for web services
- Construction of interfaces from other interfaces – *interface inheritance*
The **Grid Interoperability Project**

... to realise the interoperability of **UNICORE** and **Globus** and to work towards standards for interoperability at the Global Grid Forum

- Development of an interoperability layer between the two Grid systems
- Interoperable applications
- Contributions made to the Global Grid Forum
- UNICORE towards Grid Services

www.grid-interoperability.org

www.unicore.org
Partners

Two year project funded by the E.U. with the following partners:

- Forschungszentrum Jülich (DE)
- Pallas (DE)
- Deutschen Wetterdienst (DE)
- ICM (PL)
- Fujitsu (UK)
- University of Manchester (UK)
- University of Southhampton (UK)
- Argonne National Laboratory (US)

Project completes March 2004.
UNICORE today

• Full control over the jobs through a graphical user interface.
• Multi-system and multi-site jobs with UNICORE synchronising the jobs and staging data
• Secure & co-operates well with firewalls
• Abstraction of system functions, commands, and user actions to achieve system and installation independence. Software Resources. Plugins
• Retain full administrative autonomy at participating centres

What about the cons??

• More difficult to use it as the basis for exotic Grid applications/services.
• Lack of delegation → some restrictions
Technical points and issues

- Vertically integrated architecture
- Security based on X509 certificates and ssl. No delegation
- Java based, although Perl sometimes used for target systems
- Abstract Job Object (AJO)
  - Carries a workflow of jobs
  - Jobs described in an abstract form
  - Workflow can also contain some control constructs
Abstract

UNICORE Architecture

TSI – Target System Interface
NJS – Network Job Supervisor
IDB – Incarnation Database
UUDB – UNICORE User Database
Usite – UNICORE site
UPL – UNICORE Protocol Layer

Indicates SSL transport
Architecture proposed during the GRIP Project

TSI – Target System Interface
NJS – Network Job Supervisor
IDB – Incarnation Database
UUDB – UNICORE User Database
Usite – UNICORE site
UPL – UNICORE Protocol Layer

Indicates SSL transport
Grid/Web service <porttype>

TSI – Target System Interface
NJS – Network Job Supervisor
IDB – Incarnation Database
UUDB – UNICORE User Database
Usite – UNICORE site
UPL – UNICORE Protocol Layer
## Existing and proposed `<porttypes>` ...

<table>
<thead>
<tr>
<th><strong>UPL <code>&lt;porttype&gt;</code></strong></th>
<th><strong>IDB <code>&lt;porttype&gt;</code></strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>consignment of jobs</td>
<td>incarnation, site specific, software resources data</td>
</tr>
<tr>
<td>retrieve outcome</td>
<td>exposed as service data (?)</td>
</tr>
<tr>
<td>list of Vsites</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Broker <code>&lt;porttype&gt;</code></strong></th>
<th><strong>NJS-TSI (Job Manager) <code>&lt;porttype&gt;</code></strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>making functionality of the GRIP broker available as a Grid service</td>
<td>executing incarnated 'atomic' jobs including file staging functionality</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>UUDB <code>&lt;porttype&gt;</code></strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>read only authorisation decisions (using SAML?)</td>
</tr>
<tr>
<td>separate porttype for administration</td>
</tr>
</tbody>
</table>
Architecture comparison

**UNICORE**

vertically integrated architecture

'traditional' 3-tier architecture fits better for deployment in a 'regular' environment.

- grouping of Vsites arranged into Usites and a UNICORE grid consists of a collection of Usites.

**Globus**

dispenses with 'infrastructure' components. It operates on a “node-to-node” basis, and assumes that each machine is directly accessible.

- groupings of nodes to create VO structures.
Security

Biggest influencing factor on the design of Grid architectures?

Providing a delegation mechanism for a resource to access another resource *on behalf of a user* is both challenging and controversial.

Lots of **web service security** specifications:

- SOAP header is extensible to support security, routing, policy, etc.
- Message-level security
- Multiple signatures on a document describing a workflow
Virtualisation and Job Abstraction

Software resources

• Applications as services, not via scripts

Already exists in UNICORE. Implement as web services

• Derive a new 'software resource' porttype exposing the contents of the Incarnation Database (IDB) as service data

• Use operation extensibility of OGSI to parametrise the request of the software resource
A possible evolution

- A Grid comprising of distributed services
- **Virtualisation** - the UNICORE software resource concept maps to application specific web services
- Interoperability between different Service providers
- **Dynamic higher level services** built out of other services. Job workflow, for example, but also viewing security, authorisation, etc, as services, from which other aggregated services can be built
UNICORE Architecture from another perspective
Status of standardisation work

Most work taking place at the Global Grid Forum

Some interesting working groups:

**OGSA**: documentation of requirements, functionality, priorities, and interrelationships for OGSA services

**CMM**: define a Common Management Model and a set of OGSI porttypes for the standardised management of resources and services

**OGSI-Agreement**: agreement negotiation for the usage of services according to policy

**OGSA-Sec**: grid service security framework

**GridIR**: information retrieval system on the OGSA Grid - document collection management, indexing/searching, query processing
Summary

- Using web services is a natural direction for UNICORE project to take, and a sound move for the future.
- With the advent of OGSA, we view *interoperability in a broader sense and not just interoperability with Globus*.
- A Grid composed of services from multiple (including non-UNICORE) services.
- Service aggregation to build the functionality needed.

Forms the basis of the current work in GRIP.
Questions & comments?

www.grid-interoperability.org
www.unicore.org
www.unicorepro.com