Current Status and Future Direction

Open Grid Services Architecture

Hiro Kishimoto, Ph.D.
OGF OGSA-WG, co-chair
GFSG and OGF Board member
Research Fellow, Fujitsu
Visiting Professor, National Institute of Informatics
Agenda

• Open Grid Services Architecture Overview
• OGSA specification schedule
• OGSA and EGA reference model
• OGSA information/data modeling
The Open Grid Services Architecture

- An open, service-oriented architecture (SOA)
  - Resources as first-class entities
  - Dynamic service/resource creation and destruction
- Built on a Web services infrastructure
- Resource virtualization at the core
- Build grids from small number of standards-based components
  - Replaceable, coarse-grained
  - e.g. brokers
- Customizable
  - Support for dynamic, domain-specific content…
  - …within the same standardized framework
SOA & Web Services: Key Benefits

SOA
- **Flexible**
  - Locate services on any server
  - Relocate as necessary
  - Prospective clients find services using registries
- **Scalable**
  - Add & remove services as demand varies
- **Replaceable**
  - Update implementations without disruption to users
- **Fault-tolerant**
  - On failure, clients query registry for alternate services

Web Services
- **Interoperable**
  - Growing number of industry standards
- **Strong industry support**
- **Reduce time-to-value**
  - Harness robust development tools for Web services
  - Decrease learning & implementation time
- **Embrace and extend**
  - Leverage effort in developing and driving consensus on standards
  - Focus limited resources on augmenting & adding standards as needed
OGSA Capabilities

- **Execution Management**
  - Job description & submission
  - Scheduling
  - Resource provisioning

- **Data Services**
  - Common access facilities
  - Efficient & reliable transport
  - Replication services

- **Resource Management**
  - Discovery
  - Monitoring
  - Control

- **Information Services**
  - Registry
  - Notification
  - Logging/auditing

- **Self-Management**
  - Self-configuration
  - Self-optimization
  - Self-healing

- **Security**
  - Cross-organizational users
  - Trust nobody
  - Authorized access only

OGSA “profiles”

Web services foundation
OGSA Profiles

• The normative definition of OGSA
  • Styled on WS-I profiles to promote interoperability
  • Define specific usage patterns
    • e.g. execution management
  • Basis for claims of conformance
    • “My scheduler conforms to the OGSA Execution Management Profile…”

• Include specifications developed by OGF and by other bodies
  • Issue: How mature and widely adopted?
  • OGSA Profile Definition document provides guidelines

OGSA WSRF Basic Profile
OGSA Basic Security Profile – Core
OGSA Security Profile – Secure Channel
In the pipeline

Execution Mgmt Profile
HPC Profile
Data Profile
Others…
In the future

© 2006 Open Grid Forum
OGSA and Fellow WGs

- **OGSA-WG**
- **OGSA-AuthZ WG**
- **OGSA-Client WG**
- **OGSA-Naming WG**
- **OGSA-BES WG**
- **OGSA-ByteIO WG**
- **OGSA-RSS WG**

**Roadmap**
- **OGSI 1.0**
- **WSRF 1.0**
- **WSRF 1.5**

**Use Cases**
- **Arch 1.0**

**Roadmap**
- **2002**
- **2003**
- **2004**
- **2005**
- **2006**

- **OGSA-AuthZ WG**
- **OGSA-Naming WG**
- **OGSA-BES WG**
- **OGSA-ByteIO WG**
- **OGSA-RSS WG**
- **OGSA-Data WG**
- **OGSA-DMI WG**
- **OGSA-HPCP WG**
- **WSRF BP 1.0**
- **WSRF 1.0**
- **OGSA-DMI WG**
- **OGSA-Data WG**
- **OGSA-HPCP WG**

© 2006 Open Grid Forum
<table>
<thead>
<tr>
<th>Published OGSA Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base document</strong></td>
</tr>
<tr>
<td>✓ Use Cases (GFD-I.29)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Guideline</strong></td>
</tr>
<tr>
<td><strong>Service description</strong></td>
</tr>
<tr>
<td><strong>Profile</strong></td>
</tr>
<tr>
<td><strong>Specification</strong></td>
</tr>
</tbody>
</table>
OGSA Schedule

2005

Base document

Guideline

Service description

Profile

Specification

2006

Roadmap V1.1

Modeling guideline

EMS arch scenarios

Data architecture

Basic Security Profiles

HPC Profile

2007

GFD publication

Bytel O

Basic Execution Service

Container info-model

WS-Naming

Public comment start
## Document schedule (1)

<table>
<thead>
<tr>
<th>Document name</th>
<th>First draft available</th>
<th>Ready for PC</th>
<th>GFD publication</th>
</tr>
</thead>
</table>
| OGSA WSRF Basic Profile 1.0                        | May 2005
  May 2005                                           | Sept. 2005
  Sept 2006                                          |
| OGSA Architecture & Glossary 1.5                   | Oct. 2005
  Mar. 2006                                          | July 2006
  Sept 2006                                          |
| OGSA Security Profile 1.0 – Secure Channel         | Sept. 2005
  Sept. 2005                                          | Nov. 2005
  Feb. 2006                                          | Aug. 2006
  Oct. 2006                                          |
| OGSA Basic Security Profile 1.0 – Core             | Sept. 2005
  Sept. 2005                                          | Nov. 2005
  Feb. 2006                                          | Aug. 2006
  Oct. 2006                                          |

Plan date
Actual date
completed
delayed
## Document schedule (2)

<table>
<thead>
<tr>
<th>Document name</th>
<th>First draft available</th>
<th>Ready for PC</th>
<th>GFD publication</th>
</tr>
</thead>
<tbody>
<tr>
<td>OGSA EMS Architecture scenarios capability</td>
<td>TBD (3~6 month)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Document schedule (3)

<table>
<thead>
<tr>
<th>Document name</th>
<th>First draft available</th>
<th>Ready for PC</th>
<th>GFD publication</th>
</tr>
</thead>
</table>

These document should go together!!
OGSA and EGA reference model

- Glossary comparison
  - Identify similarities and differences between terminology.
- ERG Ref Model and OGSA
  - Establish mutual understanding
  - Working hypothesis: “OGSA is the SOA and Web service based instantiation of Reference Model”
### Glossary Of Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstraction</td>
<td>Abstraction involves changing the interface of an object and exposing it in a more useful or manageable form, software, or what have you.</td>
</tr>
</tbody>
</table>

#### 2. Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>Authentication, authorization and accounting.</td>
</tr>
<tr>
<td>Abstract name</td>
<td>See name.</td>
</tr>
</tbody>
</table>
| ACID   | Four properties that must generally apply to stateful resources used within the context of a transactional unit of work within a traditional, two-phase-commit-enabled transactional system. Briefly:  
- Atomicity: Updates must be made in an all-or-nothing fashion.  
- Consistency: Resources must be left in a consistent state, even in the event of failure. |
## Statistics

<table>
<thead>
<tr>
<th></th>
<th>EGA</th>
<th>OGSA</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terms</td>
<td>42</td>
<td>156</td>
<td></td>
</tr>
<tr>
<td>Trivial</td>
<td>8</td>
<td>41</td>
<td>Expands initials or refers to another definition</td>
</tr>
<tr>
<td>Overlap</td>
<td>11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Overlapping Terms (1)

<table>
<thead>
<tr>
<th>EGA</th>
<th>OGSA</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIM</td>
<td>CIM</td>
<td>Both trivial.</td>
</tr>
<tr>
<td>Enterprise grid, Grid computing</td>
<td>Grid</td>
<td>OGSA definition is formal; EGA is more explanatory.</td>
</tr>
<tr>
<td>GGF</td>
<td>GGF</td>
<td>Both trivial – deprecate! :0)</td>
</tr>
<tr>
<td>Grid component</td>
<td>Component</td>
<td>OGSA definition is brief &amp; formal; EGA explanatory, wider scope.</td>
</tr>
<tr>
<td>OGSA</td>
<td>OGSA</td>
<td>Both trivial!</td>
</tr>
<tr>
<td>Provisioning</td>
<td>Provisioning</td>
<td>Not identical, but no real conflict. OGSA was considering revising.</td>
</tr>
</tbody>
</table>
## Overlapping Terms (2)

<table>
<thead>
<tr>
<th>EGA</th>
<th>OGSA</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource</td>
<td>Resource</td>
<td>Similar.</td>
</tr>
<tr>
<td>Service</td>
<td>Service</td>
<td>OGSA is formal &amp; succinct; EGA more explanatory.</td>
</tr>
<tr>
<td>SLA</td>
<td>SLA</td>
<td>OGSA is a little more explanatory; OGSA mentions “level of service” vs. EGA’s “properties &amp; attributes.”</td>
</tr>
<tr>
<td>SLO</td>
<td>SLO</td>
<td>No conflict.</td>
</tr>
<tr>
<td>Virtualization</td>
<td>Virtualize</td>
<td>No conflict; EGA is more explanatory.</td>
</tr>
</tbody>
</table>
Observations

- No violent disagreement
- EGA definitions tend to be longer and more explanatory
  - A few of the OGSA definitions are similar in style
  - Consider if styles should be merged
- Need to look for non-obvious overlaps
  - There might be some hidden conflicts
- May need to define some new terms
  - e.g., “enterprise”
OGSA HPC cluster usecase

• HPC cluster is set of computational servers connected high-speed network
  • It is not grid in general sense
• Administrator deploy multiple applications on computing nodes
• Users submit computational jobs to cluster (management host)
  • Each job runs on one or more computing nodes exclusively
  • More than one job can run on each node in turns
• EPS (scheduler) prioritize submitted jobs based on administers policy
• EPS monitors and logs note usage for accounting and billing
Workload

- Non interactive batch workload
- User may submit multiple jobs
  - E.g. workload consists of 100 jobs
  - Each job last several seconds to several days
  - Single job (run on single node) or parallel job (run on multiple nodes)
- Each job is “abstract application”: top level entity of Grid Component tree diagram
  - Job is different form “transactions” of online shopping application
Batch job: grid component mapping

Abstract Application Definition
- Paul’s BLAST
- Hiro’s BLAST
- Andrew’s BLAST
- Paul’s CHARMM
- Hiro’s CHARMM

Virtualized Operating Environment

HPC Cluster

Physical
- BLAST binary
- CHARMM binary
- Mgmt server
- Comp node

Each node is shared by multiple jobs, even though job runs exclusively.
OGSA HPC usecase diagram

EGA reference model defines these usecases

User

Provision node
Provision OS
Provision server pool
Deploy Application
Submit Job
Manage job
Undeploy application
Decommission server pool
Decommission OS
Decommission node

HPC cluster

Administrator

HPC profile, JSDL, and BES only covers here

© 2006 Open Grid Forum

www.ogf.org
Resource Usage

Manage

- monitor
- provision & configure
- workload management
- info / storage mgmt
- ...

Use

- discover and explore information
- execute workload/jobs
- access information
- exploit special capabilities
- ...

publish information
Resource Information Usage
(An Observation)

• Resources are modeled for 3 purposes
  • Managing resources/information in the system/network
  • Advertising existence & capabilities of resources in the system/network
  • Expressing (requesting) set of requirements to accomplish an activity in the system/network

• System/Network management is focused on managing…..Grid is focused on advertising and requesting
  • System/Network management: models are granular & detailed; examples are CIM, GLUE, Unicore, SDD, CMDB, TPM DCM, …
  • Grid: models similar but different focus…higher level, user friendly, simpler, capabilities, requirements, more abstract

• But Grid modeled information can and should build on and re-use System/Network information…
  • For example, to route workload successfully, you don’t need to know/specify lots of requirements
OGSA Information Model Concept

Managed Info Instances
- Instances info, e.g. CIM, GLUE
- Granular
- Expresses managed information in the environment
- May be virtualized

Advertised Capabilities
- Abstracted and generated from managed information via algorithms
- Simple, meaningful, and user friendly
- Expresses the capabilities of an entity, e.g. container
- May be virtualized
- May be refreshed as managed information changes

Activity’s Requirements
- Simple, meaningful, and user friendly
- Expresses an activity’s needs
- May be virtualized

Capabilities and Requirements have same language (name-value pairs) with ability to express range values for easy mapping and selection.
Moving from Concept to Concrete...

- Relationship of EMS model to systems management ("detailed") model
- Advertisement/Requirement (ClassAd-like) model
- Representation of resource capabilities
- Representation of requirements specification
- Basic set of resource capabilities and properties from which extend
Various “detailed” information models including Glue, CIM, SDD/IUDD, SDM and others

Managed Information Instances

Monitoring Instrumentation

Actual devices and components

advertisement

<Resource>
<Capabilities>
</Capabilities>
<Requirements>
</Requirements>
</Resource>

request

<Job>
<Capabilities>
</Capabilities>
<Requirements>
</Requirements>
</Job>

Matching advertisement request algorithms managed information instances
Advertisement/Requirement Matching

请求

<Job>
  <Capabilities>
  </Capabilities>
  <Requirements>
  </Requirements>
</Job>

<Job>
  <Capabilities>
  </Capabilities>
  <Requirements>
  </Requirements>
</Job>

<Job>
  <Capabilities>
  </Capabilities>
  <Requirements>
  </Requirements>
</Job>

<Job>
  <Capabilities>
  </Capabilities>
  <Requirements>
  </Requirements>
</Job>

<Job>
  <Capabilities>
  </Capabilities>
  <Requirements>
  </Requirements>
</Job>
Advertisement/Requirement Matching

document syntax is exemplary & not rigorous on these concept charts

advertisement

<Resource>

<Capabilities>

<Capability Name="Processor.Type" Value="Power4"/>
<Capability Name="Processor.Speed" Value="10"/>

</Capabilities>

<Requirements>

</Requirements>

</Resource>

request

<Job>

<Capabilities>

</Capabilities>

<Requirements>

{ Processor.Type = "Power4" 
  and Processor.Speed > 7 } 

</Requirements>

</Job>
Advertisement/Requirement Matching

document syntax is exemplary & not rigorous on these concept charts

advertisement

<Resource>
  <Capabilities>
    <Capability Name="Processor.Type" Value="Power4"/>
    <Capability Name="Processor.Speed" Value="10"/>
  </Capabilities>
  <Requirements>
    {  Processor.Type = “Power4”
        and  Processor.Speed > 7  }
  </Requirements>
</Resource>

advertisement

<Job>
  <Capabilities>
    <Capability Name="User.VO" Value="EGEE"/>
    <Capability Name="Processor.Type" Value="Power4"/>
    <Capability Name="Processor.Speed" Value="10"/>
  </Capabilities>
  <Requirements>
    {  User.VO="EGEE" or
        User.VO="NAREGI"  }
  </Requirements>
</Job>
Proposal: Use XML Document as Capability Specification

- Declarative format
- Name-value (capabilities-property) pair “basic model”
- Limited (2-level) hierarchy
- Can be expressed as XML document and either stored natively or in a relational table.
- Concrete rendering uses XSD schema and XML typing
XML Rendering of Resource Capabilities

Rendering: “Concrete”

```xml
<Resource Name="BESNode3">
  <Capabilities>
    <Processor>
      <Architecture>Power4</Architecture>
      <Speed>10</Speed>
    </Processor>
    <OS>
      <OSType>Linux</OSType>
      <PhysicalMemory>3000000</PhysicalMemory>
      <MaxProcessesPerUser>32</MaxProcessesPerUser>
    </OS>
    <SoftwareLicense>
      <SAP/>
    </SoftwareLicense>
  </Capabilities>
</Resource>
```

- Capitalize on tooling for generation and validation
- XML typing used; reuse of element names requires use of namespaces
- XSD can be used but must be open for extensibility, i.e. `<any>`
- Extensions can be derived from existing models
- Existing XML models can be included

<table>
<thead>
<tr>
<th>Resource</th>
<th>Capability</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BESNode3</td>
<td>Processor</td>
<td>Architecture</td>
<td>Power4</td>
</tr>
<tr>
<td>BESNode3</td>
<td>Processor</td>
<td>Speed</td>
<td>10</td>
</tr>
<tr>
<td>BESNode3</td>
<td>OS</td>
<td>OSTYPE</td>
<td>Linux</td>
</tr>
<tr>
<td>BESNode3</td>
<td>OS</td>
<td>PhysicalMemory</td>
<td>3000000</td>
</tr>
<tr>
<td>BESNode3</td>
<td>OS</td>
<td>MaxProcessesPerUser</td>
<td>32</td>
</tr>
<tr>
<td>BESNode3</td>
<td>SoftwareLicense</td>
<td>SAP</td>
<td>1 (True)</td>
</tr>
</tbody>
</table>
Proposal: Use XQuery (in some form) as a requirements specification grammar.

- XQuery Status:
  - Based on XPath 2.0
  - Spec is a Candidate Recommendation; implementation test suite exists
- “Candidate Set Generation” (CSG) becomes an XQuery over all the “advertised” capabilities/requirements between execution endpoints and a job.
- XQuery language is a fairly complete “algebra” with many comparison, arithmetic, operators.
  - XQuery is really an XML programming language including FLWOR [for, let, while, order-by, return] blocks, and IF/THEN logic etc.
  - Suggest development of a profile establish reasonable use subset (spec already lists optional parts...can/should we limit further?)
- User defined functions and prologues can be exploited to make XQuery “requirements” specifications simpler.
- Implementation of “resource repository” may either be native XML document store or data base (ODBC/JDBC)
Full Copyright Notice

Copyright (C) Open Grid Forum (2005, 2006). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.