GFD-I.137

Ellen J. Stokes, IBM
Sergio Andreozzi, INFN
Michel Drescher, Fujitsu
Andreas Savva, Fujitsu

28 July 2008

Information and Data Modeling in OGSA® Grids

Status of This Document

This document provides information to the Grid community on the direction for information and data modeling of OGSA resources. It does not define any standards or technical recommendations. Distribution is unlimited.

Copyright Notice

Trademarks

Abstract

Resources in a grid need to advertise their capabilities, and activities in a grid need to consume those resources. This architecture paper defines the way to model resources' capabilities and requirements in OGSA grids. It builds on the wealth of existing systems management information already modeled and instantiated in systems today. Examples are included.

Contents

1. Introduction

Resources[1] in a grid need to advertise their capabilities, and activities in a grid need to consume those resources. This architecture paper defines the way to model resources' capabilities and requirements in an OGSA environment. It builds on the wealth of existing systems management information already modeled and instantiated in systems today. Examples are included.

2. Overall Model for Grid Resources

Modeling resources in a grid has several basic aspects: a reference model, an information model, and a data model. Users – suppliers of models and consumers of models – add their specific managed information to an information model and a data model. It is important to understand the relationship of these aspects to produce and maintain a coherent and consistent model of managed information for grids. Figure 1 depicts these aspects and their relationships.

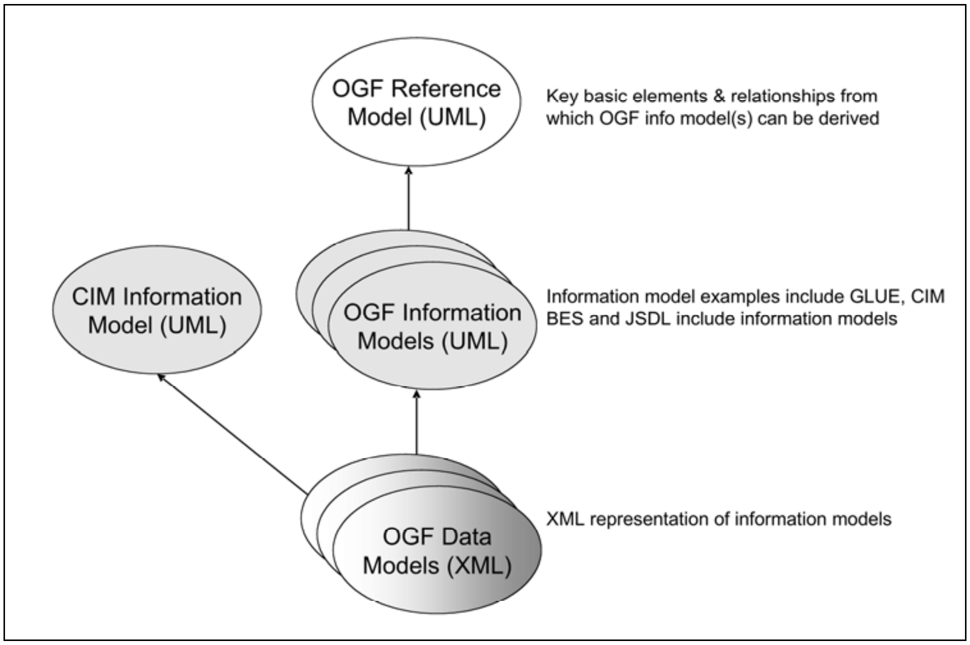
OGF Reference Model (UML)
Key basic elements & relationships from which OGF info model(s) can be derived
CIM Information Model (UML)
OGF Information Models (UML)
Information model examples include GLUE, CIM BES and JSDL include information models
OGF Data Models (XML)
XML representation of information models

Figure 1. Reference, Information, and Data Model Relationships

The reference model [RefModel] is a very general abstract model in UML that defines a small number of key basic elements and their relationships for grids from which all information model elements for grids can be derived. The reference model also describes a high-level lifecycle model for Grid Components.

The information model is a derivation of the reference model for specific useful, used, and managed grid resources. It is provided in UML. GLUE (grid compute and storage resources) is an example of an information model. Basic Execution Service (BES) and Job Submission Description Language (JSDL) include information models. Systems and network information models such as DMTF's Common Information Model (CIM) also describe elements that may be used in grids.

A data model is a concrete representation of an information model used to represent the data in real software systems. A data model may have one or more representations. Examples of data

[1] Although one can distinguish between a resource and a service, for brevity in this paper the term resource includes both resources and services.

model representations are XML and relational statements. OGSA prefers XML because of the service orientation and the grounding on Web Services related standards.

The OGSA information and data model defines an architecture to maintain a coherent and consistent view among the various model work (at OGF and DMTF) to produce a usable implementation from the distinct piece-parts that are needed for grids, for example, GLUE, Basic Execution Service (BES), Job Submission Description Language (JSDL), and systems/network management models (e.g. CIM). The architecture was define according to the modeling guidelines [MODELguide].

3. Model for OGSA Resources

Resources are typically modeled for three reasons: (1) to manage resource information in a system, (2) to advertise the capabilities of resources in a system, and (3) to express a set of requirements such as those needed by a job that is to run in a system.

Today, a system's resources are typically modeled so systems (including network) management applications can manage (provision, configure, manage, and de-provision) those resources. Information modeled for systems/network management applications tends to be very granular and detailed. This type of granular, detailed, and administration focused information is '**managed information**. CIM is an example of an information and data model of managed information. Examples of CIM managed information are:

- The processor(s) for each computer with attributes like ProcessorFamily, Version, MaxClockSpeed, CurrentClockSpeed, DataWidth, AddressWidth, Load, CPUStatus, ExternalBusClockSpeed.
- The operating system for each computer (node) and processor with attributes like OSType, Version, LastBootUpTime, LocalDateTime, CurrentTimeZone, NumberOfProcesses, MaxNumberOfProcesses, MaxProcessesPerUser, TotalSwapSpaceSize, TotalVirtualMemorySize, FreeVirtualMemory, FreePhysicalMemory.
- Each computer (node) with attributes like Name and NameFormat (e.g. DNS style hostname), Load.

Relationships between elements of the above information to express, for example, that a given computer has *n* processors, has 1 or more installed operating systems, and one of those installed operating systems is actually the running operating system.

Values of attributes are typically discrete values and specific.

Figure 2 depicts this usage from both the 'manage' and the 'use' points of view.

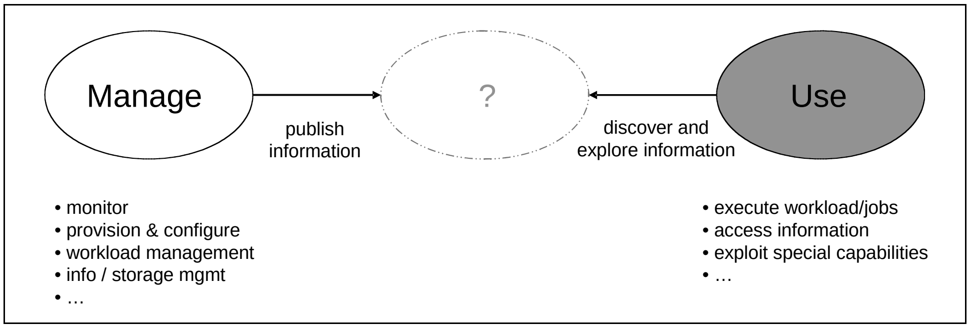

Figure 2. Resource Usage

In grids, a job (or in the more general sense – an activity) needs to run where its resource requirements are satisfied or can be provisioned to satisfy those requirements. To make that determination, systems in grids need to advertise the capabilities of their resources. Activities that are run in a grid may be, for example, parallelized applications and run on multiple nodes in the grid. So requirements may need to be expressed for multiple resources of the same type such as activity A needs a minimum of three processors with *n* CPU seconds or *m* seconds of wall clock time available as well as total CPU seconds or wall clock time for that activity. Therefore, these requirements - called **activity requirements** – and capabilities – called **advertised capabilities** – need to be expressed in terms that are meaningful to the activity's writer (e.g. user friendly). An activity requirement's values may be discrete or may be a range of values.

A study of the way in which managed information is modeled for systems management and how resources are modeled in grids to express requirements and capabilities leads one to conclude that the systems/network management type of managed information is not what is needed in OGSA grids. Instead, a higher level, less granular, and more user friendly information model is needed to express activity requirements and advertised capabilities. However, since managed information exists in systems today, it is beneficial to determine a resource's advertised capabilities from that existing managed information. Figure 3 shows the expanded data model from Figure 1 and proposes the data model relationships between a system's managed information, a grid's advertised capabilities, and an activity's requirements to consume those advertised capabilities.

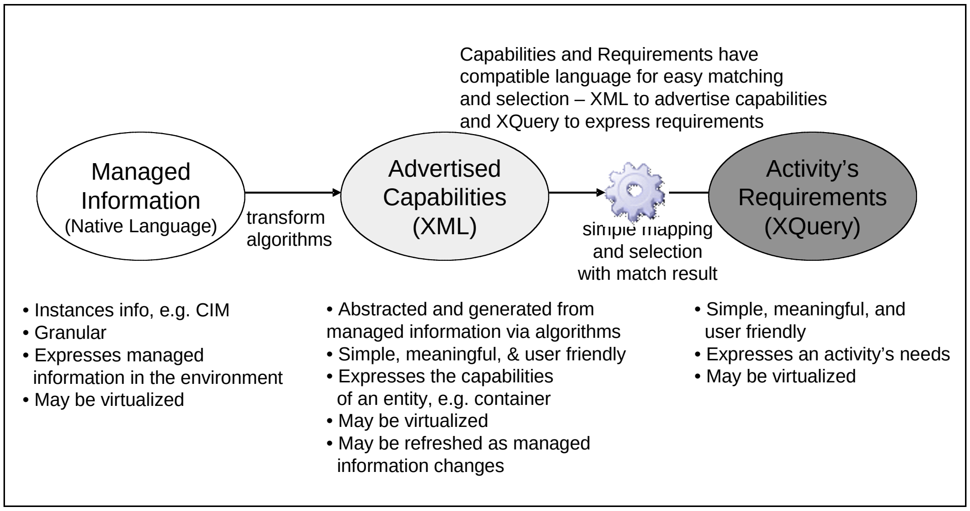
Capabilities and Requirements have compatible language for easy matching and selection – XML to advertise capabilities and XQuery to express requirements
Managed Information (Native Language)
transform algorithms
Advertised Capabilities (XML)
simple mapping and selection with match result
Activity's Requirements (XQuery)
• Instances info, e.g. CIM
• Granular
• Expresses managed information in the environment
• May be virtualized
• Abstracted and generated from managed information via algorithms
• Simple, meaningful, & user friendly
• Expresses the capabilities of an entity, e.g. container
• May be virtualized
• May be refreshed as managed information changes
• Simple, meaningful, and user friendly
• Expresses an activity's needs
• May be virtualized

Figure 3. OGSA Data Model Concept

Capabilities are abstracted and generated from managed information via algorithms. These capabilities are updated as the managed information in the system changes. An activity's requirements are matched against the advertised capabilities to determine which resources it will consume and hence where that activity can execute. These advertised capabilities and activity requirements use compatible languages for easy matching and selection of resources – XML to advertise capabilities and XQuery to express requirements.

There are two categories of capabilities that can be advertised. The first category is capabilities that are typically consumed by an activity such as an application. These capabilities typically have static values such as total physical memory or operating system type. The second category is capabilities that are typically consumed by some system component such as a job manager. These capabilities may change as activities are executed such as available physical memory or load.

4. Example

In this section, we describe an example that considers the modeling of a managed system, its advertised capabilities, and the expression of activity requirements. The managed system is modeled using the CIM information model, the advertised capabilities are described using the GLUE information model [GLUE], and the activity requirements are expressed as XPath/XQuery expressions on the GLUE-based XML rendering of the advertised capabilities.

Using the GLUE terminology, the example considers an administration domain ACME exposing a batch system to the Grid by using the BES interface. The batch system is locally managed by OpenPBS and is composed of two different subsets of nodes: 1) an old set of 50 machines based on Intel® Pentium® 4 2.0 GHz (SpecINT2000 640) and 1 GB RAM; 2) a new set of 50 machines of bi-processor dual-core Intel® Xeon® 5160 3.00GHz (SpecINT2000 3061) Two shares (blue and green) are configured in order to access either the old machine nodes or the new machine nodes. The Virtual Organization Blue signs an agreement with the administration domain ACME in order to access the BES endpoint to run activities on the old machine nodes. The Virtual Organization Green signs an agreement with the administration domain ACME in order to access the BES endpoint to run activities on the new machine nodes.

In Figure 4, we present a simple diagram showing the above scenario and the main concepts defined by GLUE which are relevant to describe this scenario. In particular, each VO maps to a UserDomain. The whole computing facility maps to the GLUE concept of Computing Service which is composed of a Computing Endpoint describing the BES interface instance, the Blue and Green shares that map to two instances of the GLUE Computing Share, and each type of machine that maps to an instance of the Execution Environment. The GLUE Application Environment is not used in this example, but it is a relevant concept to model software packages available in the various execution environments. The GLUE computing resource represents a local management scope for computing resources typically defined by a local resource manager and the managed resources (in this example OpenPBS and the 100 nodes).

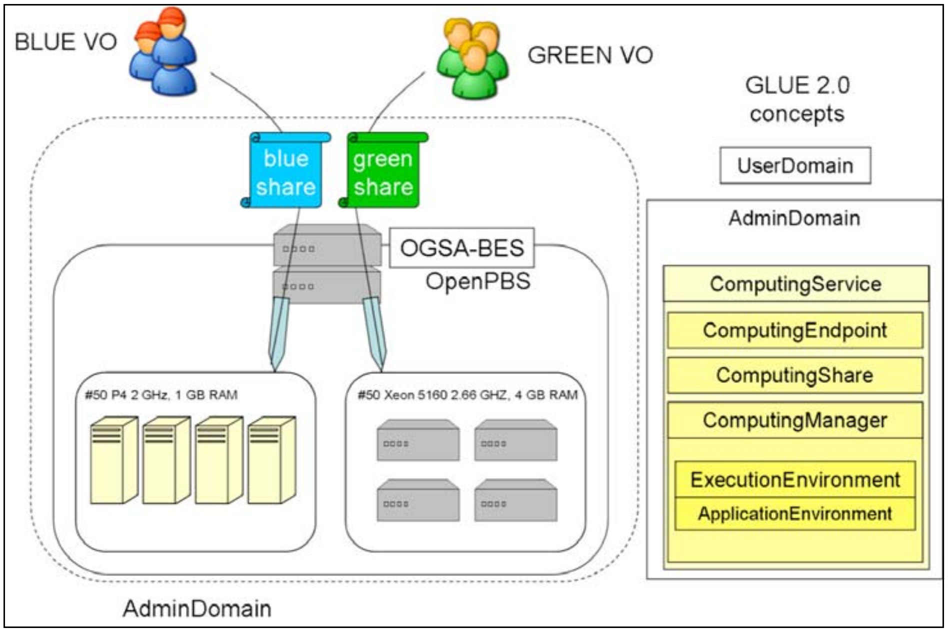

Figure 4. Example (picture form)

This example shows the managed information in a system (using CIM), the capabilities advertised (using GLUE which can be exposed via the BES interface) from that system that an activity can consume, and the activity's requirements (using JSDL) that are matched against the advertised capabilities. The activity is some application with a JSDL document, and the capabilities advertised are those that may be typically matched against a JSDL document. Capabilities that a system component may want to consume are not advertised in this example.

This example is provided below in English text for ease of use by the reader. In practice, it would be implemented in XML/XQuery format consistent with OGF information/data models. A subset of the complete environment (for ease of use) is presented that is further explored and rendered in XML/XQuery in Section 5.

4.1 Managed Information

As stated in the previous part, the managed information of a system is captured by models such as CIM. Some values of the managed information are static for the life of the instance (e.g. hostname, operating system type), and some values change over time (e.g. processor load, free memory (physical, virtual, page space)).

The managed information consists of the following:

There is an administration domain named ACME and it has a DNS-style name, Name=acme.com. It has 2 execution environments (in CIM, an execution environment is represented by a ComputerSystem). ComputerSystem A has a DNS-style name, Name=xeon.acme.com. It is classified as a non-dedicated system (ability to run applications, store data, act as a route or gateway, etc) with attribute Dedicated=0. Likewise, ComputerSystem B has a DNS-style name, Name=pentium.acme.com and the same characteristics as ComputerSystem A. In the following sections, execution environment Pentium 4 is modeled as ComputerSystemA, and execution environment Xeon is modeled as ComputerSystem B.

ComputerSystem A has 50 machine nodes / processors. Each machine node/processor has the following managed information:

- ProcessorFamily=178 (Pentium(R) 4)
- MaxClockSpeed=2000 MHz
- CurrentClockSpeed=2000 MHz
- DataWidth=8 bit
- AddressWidth=32 bit
- Load=each processor has its load expressed as a percentage
- CPUStatus=1 (CPU enabled)
- ExternalBusClockSpeed=800 MHz (front side bus)
- Characteristics=2 (64-bit capable)

ComputerSystem A is configured and running Scientific Linux operating system. Managed information for the current running Linux operating system is:

- OSType=1 (Other)
- OtherTypeDescription=Scientific Linux
- Version=4.0.5 (major.minor.revision)
- LastBootUpTime=20080201
- CurrentTimeZone=-6 (CentralUS)
- NumberOfUsers=4
- NumberOfProcesses=67
- MaxNumberOfProcesses=0 (no maximum)
- TotalVirtualMemorySize=amt of total RAM + amt of paging space (SizeStoredInPagingFiles)=12000000 KB
- FreeVirtualMemory=amt of free RAM + amt of free paging space (FreePhysicalMemory + FreeSpaceInPagingFiles)=4100000 KB
- FreePhysicalMemory=500000 KB

- TotalVisibleMemorySize (amount of physical memory allocated to this OS)=1000000 KB (1GB)
- SizeStoredInPagingFiles=8000000 KB
- FreeSpaceInPagingFiles=4000000 KB
- MaxProcessMemorySize (max bytes allocated to a process)=400000 KB
- MaxProcessesPerUser=32

Additionally, ComputerSystem A has a benchmark SpecINT2000 set to 640, and services need to be of production quality (as opposed to, for example, development or test level quality). Although these pieces of managed information are not present in current CIM, they could be added. For purposes of this example they are assumed to exist in CIM.

ComputerSystem B has 50 machine nodes / processors. Each machine node/processor has the following managed information:

- ProcessorFamily=179 (Intel(R) Xeon(TM))
- Version=5160
- MaxClockSpeed=2660 MHz
- CurrentClockSpeed=2660 MHz
- DataWidth=8 bit
- AddressWidth=32 bit
- CPUStatus=1 (CPU enabled)
- ExternalBusClockSpeed=200 MHz (front side bus)
- Characteristics=3 (32-bit capable)

ComputerSystem B is configured and running Scientific Linux operating system. Managed information for the current running Linux operating system is:

- OSType=1 (Other)
- OtherTypeDescription=Scientific Linux
- Version=4.0.5 (major.minor.revision)
- LastBootUpTime=20080201
- CurrentTimeZone=-6 (CentralUS)
- NumberOfUsers=2
- NumberOfProcesses=31
- MaxNumberOfProcesses=0 (no maximum)
- TotalVirtualMemorySize=amt of total RAM + amt of paging space (SizeStoredInPagingFiles)=1500000 KB
- FreeVirtualMemory=amt of free RAM + amt of free paging space (FreePhysicalMemory + FreeSpaceInPagingFiles)=1100000 KB
- FreePhysicalMemory=100000 KB
- TotalVisibleMemorySize (amount of physical memory allocated to this OS)=4000000 KB (4 GB)
- SizeStoredInPagingFiles=1000000 KB
- FreeSpaceInPagingFiles=500000 KB
- MaxProcessMemorySize (max bytes allocated to a process)=150000 KB
- MaxProcessesPerUser=8

Additionally, ComputerSystem A has a benchmark SpecINT2000 set to 2061, and services need to be of production quality (as opposed to, for example, development or test level quality). Although these pieces of managed information are not present in current CIM, they could be added. For purposes of this example they are assumed to exist in CIM.

4.2 Advertised Capabilities

The advertised capabilities of Grid resources can be represented with information models such as GLUE. Execution environments Xeon and Pentium 4 advertise their capabilities as stated below. These example capabilities are algorithmically generated from granular detailed managed information. For this example, the detailed managed information is from CIM and the resulting advertised capabilities (names and unit values) are from GLUE exposed via BES. Some capabilities may map almost one-to-one with the managed information, e.g. the units of the value are different (CIM memory elements are kilo-bytes and GLUE memory elements are bytes). Some capabilities are computed from one or more pieces of managed information, e.g. PhysicalMemory for a machine with 2 processors that is advertised as available for 1 or more activities is the computation (TotalVisibleMemorySize for processor 1 + TotalVisibleMemorySize for processor 2).

Execution environments Xeon and Pentium 4 can execute work (activities) because they do not have an advertised work restriction. Capability names were selected for ease of use by the reader.

ExecutionEnvironmentID=pentium.acme.com

- Processor
  - PhysicalCPUs=50
  - LogicalCPUs=50
  - CPUMultiplicity=singlecpu-singlecore
  - CPUModel= Intel® Pentium 4®
  - CPUClockSpeed=2000 (units are Mhz)
  - OSFamily=linux
  - OSName=ScientificLinux
  - OSVersion=4.0.5
  - MainMemorySize=1000000000 (units are bytes)
  - VirtualMemorySize=1000000000 (units are bytes)

ExecutionEnvironmentID=xeon.acme.com

- Processor
  - PhysicalCPUs=100
  - LogicalCPUs=200
  - CPUMultiplicity=multicpu-multicore
  - CPUModel=Intel® Xeon™
  - CPUVersion=5160
  - CPUClockSpeed=2660 (units are Mhz)
  - OSFamily=linux
  - OSName=ScientificLinux
  - OSVersion=4.0.5
  - MainMemorySize=4000000000 (units are bytes)
  - VirtualMemorySize=4000000000 (units are bytes)

4.3 Activity Requirements

Activities 1 and 2 list their activity requirements needed to execute. These example activity requirements (names and unit values) are from JSDL, where possible; some have been extended or annotated for illustrative purposes. These activity requirements are mapped against advertised capabilities to determine which resources this activity will consume and hence where the job will run.

Activity1 requirements

- CPUArchitecture=x86
- OperatingSystemType=LINUX
- ServiceQuality=production

Activity2 requirements

- CPUArchitecture=x86 (xeon)
- TotalCPUCount>59
- ServiceQuality=production

It should be noted that the set of terms defined in JSDL 1.0 is expected to be revised in favor of query expressions defined against a resource representation. This means that instead of a descriptive approach where users' requirements are expressed as predicated on individual terms, the requirements are expected to be expressed as query expressions using languages such as

XPath/XQuery written considering the resource descriptions as defined, for instance, in the GLUE information model XML rendering.

4.4 Matching activity requirements with advertised capabilities

A system component, such as a job manager, will match an activity's requirements against advertised capabilities to determine the possible place(s) that activity may execute. Note that some names and/or values may need to be matched through mapping.

Activity 1 can execute in either execution environment xeon or pentium.

Activity 2 can execute in execution environment xeon.

5. OGSA Model Architecture

To move from the concept described above to a concrete architecture (and implementation), the following items form the architecture or pattern.

1. Relationship to detailed systems/network management detailed models (Figure 3. OGSA Data Model Concept)
2. A concrete advertisement / requirement model: approach based on Condor class-ads (Figure 5. OGSA Data Model Architecture and Figure 6. Advertisement-Requirement Matching)
3. Representation of advertised capabilities: a simple XML document format (Figure 7. Advertisement-Requirement Matching)
4. Representation of requirements: XPath 2.0 or XQuery 1.0 (examples 1-4)
5. Basic set of resource capabilities and properties from which to extend

5.1 The concrete model architecture and relationship

Figure 5 shows the concrete OGSA model architecture and its relationship to systems management models.

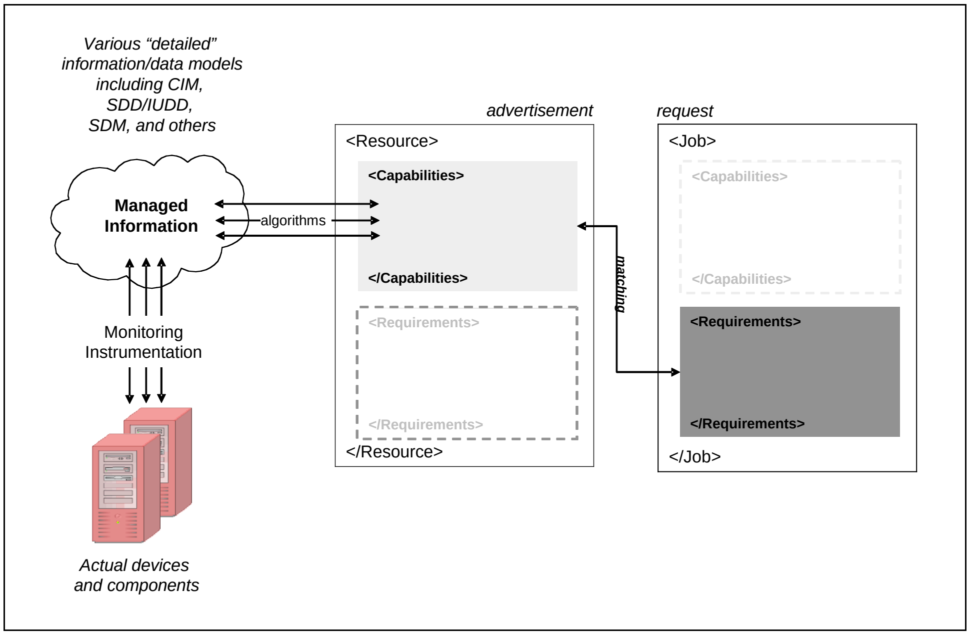

Figure 5. OGSA Data Model Architecture

Resources advertise capabilities – physical, logical, and generated. These capabilities are abstracted from detailed managed information via algorithms. Some capabilities are static, others are dynamic. Those that are dynamic are refreshed as their values change or are created/deleted per their lifespan. Jobs (or activities) specify their activity requirements and are matched against advertised capabilities advertise to determine placement, execution, etc.

Taking this one step further, resources also have a need to express activity requirements (e.g. policy) and jobs (or activities) also have a need to express advertised capabilities (e.g. identity). The concrete model allows for this symmetry as shown in Figure 6. It is recognized that there are specifications in place for things like policies and identification – this architecture is not meant to replace any of those specifications but rather to note that policy and identification are an integral part of the concrete model and further investigation needs to occur to understand how best to convey that information with respect to resources and jobs (or activities).

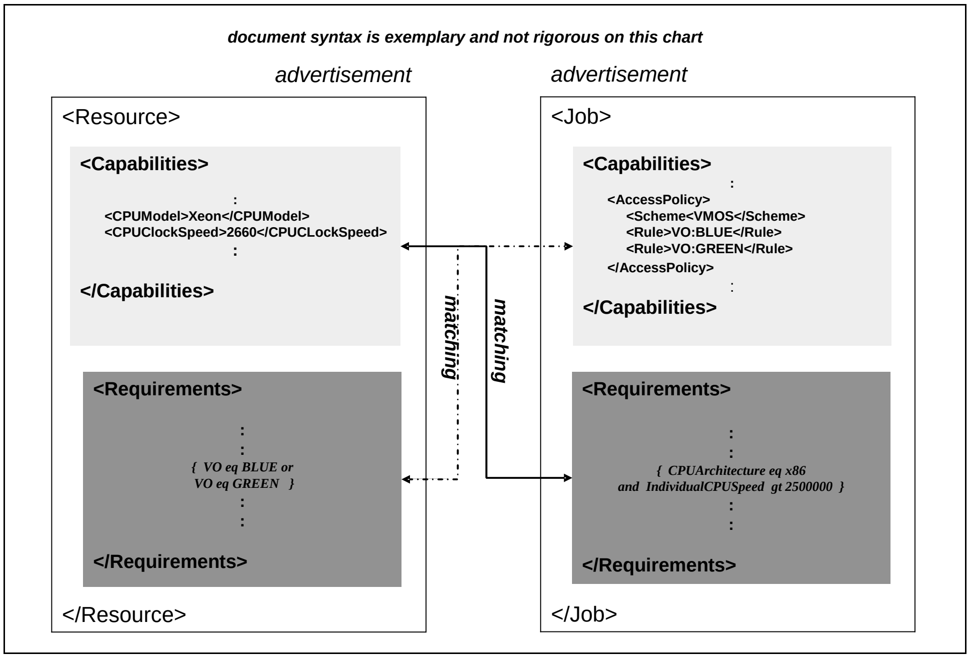

Figure 6. Advertisement-Requirement Matching

5.2 Representation of advertised capabilities

Advertised capabilities are represented in one or more XML documents. This provides the user/developer/tooling a declarative format. Capabilities are expressed in XML, that is, each capability has a name and value(s). The advertised capabilities can be stored either natively as an XML document or in a relational table (for searching and matching). Most database systems today support XML document to relational table conversion. The concrete rendering uses XSD schema and XML typing. In XML typing, reuse of element names requirements use of namespaces. The <*any*> construct is used for extensibility in XSD. Extensions can be derived from existing models and the OGSA basic resources. And existing XML models can be included in the XML document. Because typing an XML document can be tedious and error-prone, this representation and its characteristics for advertised capabilities was chosen to capitalize on existing tooling for generation and validation to quickly seed adoption.

Figure 7 is an example of rendering a few advertised capabilities in this representation. It is based on the GLUE 2.0 XSD realization [GLUEReal]. Advertised capabilities can be physical, logical, or generated.

```
<?xml version="1.0" encoding="UTF-8"?>

<Domains xmlns="http://schemas.ogf.org/glue/2008/05/spec_2.0_d42_r01"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://schemas.ogf.org/glue/2008/05/spec_2.0_d42_r01
../xsd/GLUE2.xsd">
  <AdminDomain xmlns="" BaseType="Domain">
    <ID>urn:admindomain:acme</ID>
    <Services>
      <ComputingService BaseType="Service">
        <ID>urn:acme:cs</ID>
        <Name>ACME BES Endpoint</Name>
        <Type>org.cs</Type>
        <QualityLevel>production</QualityLevel>
        <ComputingEndpoint BaseType="Endpoint">
          <ID>urn:acme:cs:bes</ID>
          <URL>https://bes.acme.com:7443/...</URL>
          <Capability>executionmanagement.jobexecution</Capability>
          <Technology>webservice</Technology>
          <Interface>urn:ogsa:bes:1.0</Interface>
          <WSDL>http://someurl/ogsa-bes.wsdl</WSDL>
          <QualityLevel>production</QualityLevel>
          <HealthState>ok</HealthState>
          <ServingState>production</ServingState>
          <AccessPolicy BaseType="Policy">
            <LocalID>urn:rule:1</LocalID>
            <Scheme>basic</Scheme>
            <Rule>fqan:/blue</Rule>
            <Rule>fqan:/green</Rule>
          </AccessPolicy>
          <Associations>
            <ComputingShareLocalID>green</ComputingShareLocalID>
            <ComputingShareLocalID>blue</ComputingShareLocalID>
          </Associations>
        </ComputingEndpoint>
        <ComputingShares>
          <ComputingShare BaseType="Share">
            <LocalID>green</LocalID>
            <MaxWallTime>432000</MaxWallTime>
            <MinWallTime>1000</MinWallTime>
            <MaxCPUTime>432000</MaxCPUTime>
            <MaxTotalJobs>50</MaxTotalJobs>
            <MaxRunningJobs>500</MaxRunningJobs>
            <MaxWaitingJobs>200</MaxWaitingJobs>
            <ServingState>production</ServingState>
            <TotalJobs>40</TotalJobs>
            <RunningJobs>30</RunningJobs>
            <WaitingJobs>1</WaitingJobs>
            <FreeSlots>1</FreeSlots>
            <MappingPolicy BaseType="Policy">
              <LocalID>urn:rule:2</LocalID>
              <Scheme>VOMS</Scheme>
              <Rule>fqan:/green</Rule>
            </MappingPolicy>
            <Associations>
              <ComputingEndpointID>urn:acme:cs:bes</ComputingEndpointID>
              <ExecutionEnvironmentLocalID>urn:acme:xeon
              </ExecutionEnvironmentLocalID>
            </Associations>
          </ComputingShare>
          <ComputingShare BaseType="Share">
            <LocalID>blue</LocalID>
            <MaxWallTime>432000</MaxWallTime>
            <MinWallTime>1000</MinWallTime>
            <MaxCPUTime>432000</MaxCPUTime>
            <MaxTotalJobs>50</MaxTotalJobs>
            <MaxRunningJobs>500</MaxRunningJobs>
            <MaxWaitingJobs>200</MaxWaitingJobs>
            <ServingState>production</ServingState>
            <TotalJobs>40</TotalJobs>
            <RunningJobs>30</RunningJobs>
            <WaitingJobs>1</WaitingJobs>
            <FreeSlots>1</FreeSlots>
            <MappingPolicy BaseType="Policy">
              <LocalID>urn:rule:2</LocalID>
              <Scheme>basic</Scheme>
              <Rule>fqan:/blue</Rule>
            </MappingPolicy>
            <Associations>
              <ComputingEndpointID>urn:acme:cs:bes</ComputingEndpointID>
              <ExecutionEnvironmentLocalID>urn:acme:Pentium
              </ExecutionEnvironmentLocalID>
            </Associations>
          </ComputingShare>
        </ComputingShares>
        <ComputingManager BaseType="Manager">
          <ID>urn:acme:cluster</ID>
          <Type>OpenPBS</Type>
          <Version>3.0</Version>
          <Homogeneity>false</Homogeneity>
          <ExecutionEnvironment BaseType="Resource">
            <ID>urn:acme:pentium</ID>
            <Platform>i386</Platform>
            <TotalInstances>50</TotalInstances>
            <PhysicalCPUs>50</PhysicalCPUs>
            <LogicalCPUs>50</LogicalCPUs>
            <CPUVendor>Intel</CPUVendor>
            <CPUModel>Intel(r) Pentium(r) 4</CPUModel>
            <CPUClockSpeed>2000</CPUClockSpeed>
            <MainMemorySize>1000000000</MainMemorySize>
            <VirtualMemorySize>1000000000</VirtualMemorySize>
            <OSFamily>linux</OSFamily>
            <OSName>scientificlinux</OSName>
            <OSVersion>4.0.5</OSVersion>
            <Benchmark>
              <LocalID>benchmark:1</LocalID>
              <Type>specint2000</Type>
              <Value>785</Value>
            </Benchmark>
            <Associations>
              <ComputingShareLocalID>blue</ComputingShareLocalID>
            </Associations>
          </ExecutionEnvironment>
          <ExecutionEnvironment BaseType="Resource">
            <ID>urn:acme:xeon</ID>
            <Platform>itanium</Platform>
            <TotalInstances>50</TotalInstances>
            <PhysicalCPUs>100</PhysicalCPUs>
            <LogicalCPUs>200</LogicalCPUs>
            <CPUVendor>Intel</CPUVendor>
            <CPUModel>Intel(r) Xeon(tm)</CPUModel>
            <CPUVersion>5160</CPUVersion>
            <CPUClockSpeed>3000</CPUClockSpeed>
            <MainMemorySize>4000000000</MainMemorySize>
            <VirtualMemorySize>4000000000</VirtualMemorySize>
            <OSFamily>linux</OSFamily>
            <OSName>scientificlinux</OSName>
            <OSVersion>4.0.5</OSVersion>
            <ConnectivityIn>false</ConnectivityIn>
            <ConnectivityOut>true</ConnectivityOut>
            <NetworkInfo>infiniband</NetworkInfo>
            <Benchmark>
              <LocalID>urn:acme:xeon:benchmark</LocalID>
              <Type>specint2000</Type>
              <Value>3016</Value>
            </Benchmark>
            <Associations>
              <ComputingShareLocalID>green</ComputingShareLocalID>
            </Associations>
          </ExecutionEnvironment>
        </ComputingManager>
```

```
      </ComputingService>
    </Services>
  </AdminDomain>
</Domains>
```

Figure 7. Advertised Capabilities XML Rendering

5.3 Representation of activity requirements

A job's or activity's requirements are expressed using XPath 2.0 or XQuery 1.0. The XQuery language is a fairly complete algebra with many comparison and arithmetic operators that are needed to express requirements both as single values and ranges of values. User defined function and prologues can be exploited to make the use of XPath/XQuery specified requirements simpler to write. Implementation of a resource repository to hold a job's or activity's requirements may be either a native XML document store or a database. These queries work against XML documents – those documents that express advertised capabilities. Because writing an XML document of XPath/XQuery can be tedious and error-prone, this representation and its characteristics for specifying requirements was chosen to capitalize on existing tooling for generation and validation to quickly seed adoption.

XPath 2.0 is a reasonable, usable, and simple subset of XQuery 1.0 and can be used to express many typical activity requirements.

Below are examples of activity requirements specified using XPath 2.0 and XQuery 1.0 and the expected result obtained when processed against the advertised capabilities.

Examples 1-3 (Figure 8, Figure 9, and Figure 10) use XPath 2.0. The XPath expression itself is in bold type and the rest of each example is documentation.

```
;
; Select the values of all local IDs of all computing shares
;
//ComputingShare/LocalID

;
; Result:
;
; <LocalID>blue</LocalID>
; <LocalID>green</LocalID>
```

Figure 8. Activity Requirements XML/XQuery Rendering Example 1

```
;
; Select the IDs of all execution environments that:
; - are part of a computing service on production level, and
; - can provide 40 or more CPUs, and
; - whose CPUs are of type Xeon
;
//ComputingService[QualityLevel eq "production"]
     //ExecutionEnvironment[PhysicalCPUs gt 39]
                           [lower-case(CPUModel) eq "intel(r) xeon(r)"]/ID

;
; Result:
;
; <ID>urn:acme:xeon</ID>
```

Figure 9. Activity Requirements XML/XQuery Rendering Example 2

```
;
; Parameterized version of example 2.
;
; XPath 2.0 allows to use variables in expressions, allowing to share and
; distribute XPath expressions that can be executed in any environment,
; and possibly shared by multiple installations.
;
; Every XPath 2.0 compliant engine MUST be able to evaluate variables.
; However, initialising variables with values is dependent on the XPath
; engine!
;
//ComputingService[QualityLevel eq $qlevel]
     //ExecutionEnvironment[PhysicalCPUs gt 39]
                           [lower-case(CPUModel) eq $cpuModel]/ID

;
; Result:
;
; <ID>urn:acme:xeon</ID>
```

Figure 10. Activity Requirements XML/XQuery Rendering Example 3

Example 4 (Figure 11) described below uses XQuery.

A user belonging to the BLUE VO wants to know all the ComputingServices which contain

- at least a ComputingEndpoint matching requirements on its attributes (e.g. Type, QualityLevel, HealthState) and having a match in AccessPolicy with fqan:blue
- at least a ComputingShare matching requirements on its attributes (e.g. MaxCPUTime, MaxWallTime) and having a match in MappingPolicy with fqan:blue
- at least an ExecutionEnvironment matching requirements on its attributes (e.g. OSName and SPECInt2000)
- matching ComputingEnpoint is associated to matching ComputingShare (via //ComputingEndpoint/Associations)
- matching ComputingShare is associated to matching ExecutionEnvironment (via //ComputingShare/Associations)

To capture the complex set of requirements given above in a single expression, a more powerful query language than XPath 2.0 is required. The proper solution is XQuery for which a number of open source implementations are available (e.g. Saxon http://saxon.sourceforge.net/). There are many possibilities to write an XQuery expression for the same set of requirements, each of them having a different impact on the computational cost.

The main XQuery clause is the FLWOR (For Let Where Order by Return). The first "for" clause iterates over all the "ComputingService" elements and binds each element to the *$service* variable. It should be noted that for the sake of simplicity, the source of information is supposed to be an XML file containing the representation of all Computing Services available in a Grid system. In a real production scenario, this could be a collection of documents about distributed Grid services periodically updated.

For each value of the *$service* variable, the following "for" clause generates a triple of variables (*$endpoint, $share,* and *$exec*). Each triple contains a ComputingEndpoint element which satisfies the requirements on the Type, QualityLevel, and AccessPolicy, a ComputingShare element which satisfies the requirements on the MaxWallTime, MaxCPUTime, MappingPolicy and that is accessible by the matching ComputingEndpoint, an ExecutionEnvironment which satisfies the requirements on OSName, SPECInt2000 and that is allocated to the matching computing share. The Return clause generates the solution.

```
xquery version "1.0";
declare namespace glue = "http://schemas.ogf.org/glue/2008/05/spec_2.0_d42_r01";

<Results xmlns:glue="http://schemas.ogf.org/glue/2008/05/spec_2.0_d42_r01"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> {

for $service in doc("BES-Example.xml")
      /glue:Domains/AdminDomain/Services/ComputingService
for $endpoint in $service/ComputingEndpoint[Interface eq "urn:ogsa:bes:1.0"]
     [QualityLevel eq "production"]
     [some $rule  in AccessPolicy/Rule satisfies ($rule/text() eq "fqan:/blue")],
    $share in $service/ComputingShares/ComputingShare[MaxWallTime > 400000]
     [MaxCPUTime > 40000]
     [some $rule  in MappingPolicy/Rule satisfies ($rule/text() eq "fqan:/blue")]
     [some $association in Associations/ComputingEndpointID satisfies
               ($association/text() eq $endpoint/ID)],
    $exec in $service/ComputingManager/ExecutionEnvironment
     [OSName eq "scientificlinux"]
     [Benchmark/Type[.="specint2000"]/../Value>200]
     [some $association in Associations/ComputingShareLocalID satisfies
              ($association/text() eq $share/LocalID)]
 return
      <ComputingService>
         {$service/ID}
         <ComputingEndpoint>{$endpoint/ID}</ComputingEndpoint>
         <ComputingShare>{$share/LocalID}</ComputingShare>
         <ExecutionEnvironment>{$exec/ID}</ExecutionEnvironment>
      </ComputingService>
} </Results>
; Result
;
<?xml version="1.0" encoding="UTF-8"?>
<Results xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:glue="http://schemas.ogf.org/glue/2008/05/spec_2.0_d42_r01">
   <ComputingService>
      <ID>urn:acme:cs</ID>
      <ComputingEndpoint>
         <ID>urn:acme:cs:bes</ID>
      </ComputingEndpoint>
      <ComputingShare>
         <LocalID>blue</LocalID>
      </ComputingShare>
      <ExecutionEnvironment>
         <ID>urn:acme:pentium</ID>
      </ExecutionEnvironment>
   </ComputingService>
</Result>
```

Figure 11. Activity Requirements XML/XQuery Rendering Example 4

5.4 Basic set of resource capabilities and properties

The basic set of resource capabilities and properties for the OGSA information model is GLUE [GLUE] which is planned to be exposed through the BES interface. These GLUE resources can be adopted also by other Grid services (e.g. SRM for the Storage entities). A JSDL document [JSDL] provides an activity's requirements.

6. Security Considerations

Use of managed information in models is subject to an activity's authentication and authorization to use those resources. That authentication and authorization is outside the scope of this document. It is the responsibility of the security subsystem and protocols.

XQuery is a fairly complete algebra and can be used to express complex activity requirements. To evaluate such requirements substantial processing may be required. Therefore consuming systems should make necessary provision to bound the processing required for evaluating queries to avoid potential denial of service attacks.

7. Appendix

Appendix 7.A: Requirements specification use cases to drive any intermediate steps and XQuery 1.0 or XPath 2.0 suggested profile elements. Suggested use case items to address:

Appendix 7.A.1 JSDL version 1

JSDL 1.0 [GFD.56] defines a set of resource elements that can be used to express the requirements of the submitted job. These elements are grouped under the *jsdl:Resources* element.

Enabling the use of existing, non-XQuery enabled, JSDL documents in the Information Model Architecture is a useful intermediate step as it may facilitate adoption by existing implementations.

At a minimum a definition of a mapping of the JSDL resource requirements elements to the GLUE 2.0 information model is needed. This mapping could be restricted to a subset of JSDL resource elements, perhaps the subset identified by the HPC Basic Profile 1.0.

Additionally, an explanation or definition of how JSDL resource requirements might be translated to an equivalent XQuery representation using the GLUE 2.0 XML rendering would be useful to early adopters.

Appendix 7.A.2 JSDL XQuery extension for basic requirements

An extension to JSDL version 1 that allows the expression of XQuery requirements in JSDL documents is under development. This extension replaces the *jsdl:Resources* element and its sub-elements and is an intermediate step towards the two-way matching model described in this document.

The *jsdl:Resources* element only supports the expression of "*and*" requirements between its sub-elements. Some sub-elements, however, support the expression of "*or*" requirements within their sub-elements—e.g., *jsdl:CandidateHosts*. In addition value ranges can be defined for element values allowing the expression of constraints such as *less-than*, *equals*, *greater-than*.

Therefore a set of XQuery to satisfy "*and*" and "*or*" requirements[2] between elements and the expression of range values using the GLUE 2.0 XML rendering should be profiled.

Experience gained with the JSDL XQuery extension is expected to feed into a JSDL 2.0 with full support for the two-way matching model described in the Information Modeling Architecture.

Appendix 7.A.3 Expression of advanced requirements

Once alternatives can be expressed, it is possible that multiple available configurations may satisfy the requirements. Expressing a preference for a particular configuration then becomes desirable. For example, a heterogeneous cycle-scavenging pool may have multiple different operating systems available within it. A job might require that it runs on Linux, and specify that running on RedHat Enterprise is preferred due to certification reasons. It is therefore useful to be able to specify a preference for one configuration over another so that automated submission to the overall pool of execution engines will function correctly.

When a preferences system gets beyond a very low level of complexity (for example, where there are two sets of theoretically-independent preferences given for parts of the configuration) it becomes simpler to use a weighting or scoring system. This allows the user to describe how much more they prefer having, say, more memory over a specific operating system or version of

[2] XPath 2.0 (which XQuery 1.0 embeds) may be sufficient if it is possible to define the document-space that it searches over succinctly.

a library. Care needs to be taken here because there can be non-functional reasons for choosing a particular service to execute a job—for example, the delay to start running the job, the likelihood that the job will complete successfully, or the cost of running the job.

Evaluating advanced requirements such as preferences or weighting may not be widely supported. It is therefore useful to allow expressing such advanced requirements as optional—if they can be evaluated they can provide additional benefit to the job execution, but if not, the job can still be accepted for execution.

It should be noted that the expression of advanced requirements such as preferences and weighting may be within the scope of the OGSA Resource Selection Services WG.

8. Contributors

Ellen Stokes
IBM
stokese@us.ibm.com

Sergio Andreozzi
INFN-CNAF
sergio.andreozzi@cnaf.infn.it

Michel Drescher
Fujitsu Laboratories of Europe
Michel.Drescher@uk.fujitsu.com

Andreas Savva
Fujitsu Laboratories
andreas.savva@jp.fujitsu.com

Contributions to this document have been made by Donal Fellows, Hiro Kishimoto, Steve McGough, David Snelling, and Jay Unger.

We are also grateful to members of the OGF OGSA, Reference Model, GLUE, BES, and JSDL workgroups for discussion on the topics covered in this document.

9. Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

10. Disclaimer

This document and the information contained herein is provided on an “As Is” basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

11. Full Copyright Notice

12. References

[CIM2.17final] CIM Schema, Version 2.17 Final, Distributed Management Task Force, August 2007, http://www.dmtf.org/standards/cim/cim_schema_v217

[CIM2.17exp] CIM Schema, Version 2.17 Experimental, Distributed Management Task Force, August 2007, http://www.dmtf/org/standards/cim/cim_schema_v217

[CIMspec] “CIM Infrastructure Specification, Version 2.3“, Distributed Management Task Force, October 4, 2005, http://www.dmtf.org/standards/documents/CIM/DSP0004.pdf

[CIMOPS] “Specification for CIM Operations over HTTP, Version 1.2”, DSP0200, Distributed Management Task Force, January 9, 2007, http://www.dmtf.org/standards/published_documents/DSP200.html

[BES] I. Foster, A. Grimshaw, P. Lane, W. Lee, M. Morgan, S. Pickles, D. Pulsipher, C. Smith, M. Theimer, “OGSA Basic Execution Services, Version 1.0”, Open Grid Forum, GFD.108, July 2006, http://www.ogf.org/gf/docs/?final

[GLUE] S. Andreozzi, S. Burke, F. Ehm, L. Field, G. Galang, B. Konya, M. Litmaath, P. Millar, JP Navarro, “GLUE Specification v2.0”, Open Grid Forum, public comment version 20 May 2008, https://forge.gridforum.org/sf/go/doc14639

[GLUEReal] S. Andreozzi, S. Burke, F. Ehm, L. Field, G. Galang, B. Konya, M. Litmaath, P. Millar, JP Navarro, “GLUE v. 2.0 – Reference Realizations to Concrete Data Models”, Open Grid Forum, public comment version 20 May 2008, https://forge.gridforum.org/sf/go/doc14639

[JSDL] A. Anjomshoaa, F. Brisard, M. Drescher, D. Fellows, A. Ly, S. McGough, D. Pulsipher, A. Savva, “Job Submission Description Language (JSDL) Specification, Version 1.0”, Open Grid Forum, GFD.056, November 2005, http://www.ogf.org/gf/docs/?final

[OGSAglossary] J. Treadwell, “Open Grid Services Architecture Glossary of Terms, Version 1.6”, Open Grid Forum, GFD.120, December 2007, http://www.ogf.org/gf/docs/?final

[OGSAarch] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich, “The Open Grid Services

Architecture, Version 1.5”, Open Grid Forum, GFD.080, July 2006, http://www.ogf.org/gf/docs/?final

[MODELguide] F. Maciel, “Guidelines for Information Modeling for OGSA ® Entities”, Open Grid Forum, GFD.118, April 2008, http://www.ogf.org/gf/docs/?final