Frederico Buchholz Maciel, Hitachi, Ltd.
Editor

February 17, 2005

Resource Management in OGSA

<u>Status of This Memo</u>

This memo provides information to the Grid community on resource management in OGSA (Open Grid Services Architecture). It does not define any standards or technical recommendations. Distribution is unlimited.

<u>Copyright Notice</u>

Abstract

Grids, as any computing environment, require some degree of system management, such as the management of jobs, security, storage and networks. Management in Grids is a potentially complex task given that resources are often heterogeneous, distributed, and cross multiple management domains.

This document contains a discussion of the issues of management that are specific to a Grid and especially to OGSA. We first define the terms and describe the requirements of management as they relate to a Grid, and we then discuss the individual interfaces, services, activities, etc. that are involved in Grid management, including both management *within* the Grid and the management *of* the Grid infrastructure. We conclude with a comprehensive gap analysis of the state of manageability in OGSA, primarily identifying Grid-specific management functionality that is not provided for by emerging distributed management standards. The gap analysis is intended to serve as a foundation for future work.

Contents

1. Introduction

Any computing environment requires some degree of system management: monitoring and maintaining the health of the systems, keeping software up-to-date, maintaining user accounts, managing storage and networks, scheduling jobs, managing security, and so on. The complexity of the management task increases as the number and types of resources requiring management increases, and is further complicated when those resources are distributed.

The Grid computing model, with its use of resources that tend to be both heterogeneous and distributed across multiple management domains, faces all the traditional IT management issues, and also brings new challenges – not only in the management of its component resources, but also of the Grid itself. For example, in a Grid environment shared resources must remain accessible, key infrastructure services must be available, and virtual organizations must be maintained. It must also be possible to detect, report and deal with faults that may occur in *any* of the member domains. As Grid technology is increasingly adopted across institutions and enterprises, the distinctions between Grid environments and traditional IT environments will blur, and these challenges will become more widespread.

Effective system management is only possible if resources are manageable, and if tools are available to manage them. Today, system administrators can choose from a wide variety of management tools from system vendors, third party suppliers and the open source community. However, these tools tend to operate independently and to use proprietary interfaces and protocols to manage a limited set of resources, making it difficult for an organization to build an efficient, well-integrated management system. This issue is being addressed through the development of *manageability standards* that will enable conforming management tools to manage conforming resources in a uniform manner, and to interoperate with each other. In turn this will enable system administrators to choose their management tools and suppliers in the knowledge that, regardless of their origin, the tools can work cooperatively in an integrated management environment.

The Global Grid Forum's (GGF's) Open Grid Services Architecture (OGSA) Working Group (WG) (see https://forge.gridforum.org/projects/ogsa-wg) is developing a standard architecture for the implementation of next-generation Grids based on a Web services infrastructure. Web services are also the basis for the emerging distributed management standards, and are increasingly being used within enterprises for other purposes. However, while this common base allows the Grid community to take advantage of developments in distributed management for general IT, it is essential that we also consider the unique management requirements of Grids, identify any missing areas ("gaps"), and develop additional Grid-management standards as needed to fill those gaps.

In this document we begin the process of identifying the gaps by offering a detailed discussion of the issues of management that are specific to a Grid, as distinct from Web services and from other computational environments. We first define the terms and describe the requirements of management as they relate to a Grid, and we then discuss the individual interfaces, services, activities, etc. that are involved in Grid management, including both management *within* the Grid and the management *of* the Grid infrastructure. We conclude with a comprehensive gap analysis of the state of manageability in OGSA, primarily identifying Grid-specific management functionality that is not provided for by emerging distributed management standards. The gap analysis is intended to serve as a foundation for future work.

1.1 Related Work

The foundation for this work is the OGSA document [1] and its related glossary [2] that were developed by the OGSA-WG.

The document is also intended to build upon the work being carried out in the OASIS (Organization for the Advancement of Structured Information Standards) Web Services Distributed Management (WSDM) Technical Committee (TC) (see http://www.oasis-open.org and http://www.oasis-open.org/committees/wsdm/ for more information on OASIS and WSDM, respectively). The following text appears in the WSDM Statement of Purpose:

> To define Web services management. This includes using Web services architecture and technology to manage distributed resources. This TC will also develop the model of a Web service as a manageable resource.

The WSDM TC is developing separate documents to address Management *of* Web Services (MOWS) [3] and Management *Using* Web Services (MUWS) [4, 5]. The interfaces defined in those documents are expected to become key standards for manageability across the IT landscape, and will form the basis for management of Grids.

As the documents being developed by these and other groups mature, new versions of this document may need to be developed.

Other related work includes the following:

- Other gap analyses exist, such as the e-Science Gap Analysis [6, 7] and the GGF Data Area gap analysis [8]. These analyses mention management with respect to Grids; however they do not appear to specifically analyze the *manageability* aspects of Grids.
- The Grid Monitoring Architecture (GMA) [9, 10] describes the major components of a Grid monitoring architecture and their essential interactions. The scope of our work overlaps to some extent with that of the GMA, since monitoring is a subset of management. However, these works do not conflict: our work contains many of the GMA elements, though sometimes in a refactored form, or described using different terminology.

2. Definitions

Management (in Grids or otherwise) is the process of monitoring an entity, controlling it, maintaining it in its environment, and responding appropriately to any changes of internal or external conditions.

A **manager** initiates management actions; it might be either a management console operated by a human or a software entity that is able to monitor and control its targets automatically.

Manageability defines information that is useful for managing an entity. Manageability encompasses those aspects of an entity that support management specifically through instrumentation that allows managers to interact with the entity. The manageability may be provided by the entity itself or by a separate means.

Manageability interfaces are sets of standardized interfaces that allow a manager to interact with an entity in order to perform common management actions on it. Typical management actions include starting the entity, stopping it, and gathering performance data.

Manageable entities are entities that provide manageability interfaces and thus, as the name implies, can be managed. Manageable entities can be:

- *physical* (e.g., a node, a network switch or a disk) or *logical* (e.g., a process, a file system, a print job, or a service)
- *discrete* (e.g., a single host) or *composite* (e.g., a cluster)
- *transient* (e.g., a print job) or *persistent* (e.g., a host)

A **resource model** is an abstract representation of manageable entities which defines their schema (conceptual hierarchy and inter-relationships) and characteristics (attributes, management operations, etc.).

The term **manageable resources** (or simply **resources**) means the same as *manageable entities*. The term includes entities such as software licenses, bandwidth and routing tables that do not expose generally-useful manageability interfaces, but may still be managed by some other means.[1]

Resource management is a generic term for several forms of management as they are applied to resources. These forms of management include (but are not limited to) typical distributed resource management (DRM) activities and IT systems management activities, such as:

- reservation, brokering and scheduling
- installation, deployment and provisioning
- metering
- aggregation (service groups, WSDM collections, etc.)
- VO management
- security management
- monitoring (performance, availability, etc.)
- control (start, stop, etc.)
- problem determination and fault management

Resource management includes the various management tasks, but not the mechanisms they use, such as discovery.

Since resource management comprises many activities in many management disciplines, using the term to refer to a single activity may be ambiguous, and should be avoided.

A **resource manager** is a manager that implements one or more resource management functions.

3. Management in OGSA

3.1 Requirements

The basis for manageability in an OGSA Grid is the WSDM MUWS specification [4, 5]. This means that for a resource to be manageable, it must provide the minimum set of manageability capabilities specified by MUWS. The current 0.5 version of MUWS specifies requirements for *identity*, *state* and *metrics*. In forthcoming MUWS releases it is anticipated that *notification*, *discovery*, *configuration* and *collections* will be included. All of these topics are critical to management, and must be supported as appropriate within OGSA services.

The following list enumerates the main requirements for management in OGSA. These requirements are especially important in a large-scale, distributed environment with no centralized notion of control, such as a Grid:

[1] In a Grid environment the term *resource* is often applied only to manageable entities that are *pooled* (e.g. hosts, software licenses, IP addresses, etc.) or that provide a given *capacity* (e.g. disks, networks, memory, etc.). For these classes of resource some part of the pool and/or the capacity may be allocated and used. By this definition processes, print jobs, registry services and VOs (Virtual Organizations) are *not* resources. Notice that this is a subset of the definition of resources as manageable entities.

- **Scalability**: Management architecture needs to scale to potentially thousands of resources. Management needs to be done in a hierarchical and/or peer-to-peer (federated/collaborative) fashion to achieve this scalability, so OGSA should allow these forms of management. Hierarchical management can be implemented through manageability interfaces that allow resources to be grouped and managed collectively (e.g. Grid Monitoring Architecture (GMA) [9, 10] aggregators and intermediaries that implement WSDM collection interfaces). Hierarchical management techniques include:
  - Providing a proxy that allows a manager to perform the same action on multiple resources with a single request.
  - Computing metrics that aggregate resource data (e.g., average load, average reservation rate).
  - Filtering and aggregating events.
  - Polling resources for state (reserved, running, failed, idle, saturated, etc.) and providing the results on request, as well as sending events when the state changes (a.k.a. *pull* or *push* notification).
- **Interoperability**: Management architecture must be able to span software, hardware and service boundaries, e.g., across the boundaries between different products, so standardized and broad interoperability is essential to avoid "stovepipes." Two kinds of interoperability are needed:
  - between levels: e.g., between a resource and its manager;
  - at the same level: e.g., a scheduler accessing a broker.

  Interoperability in both cases requires that the interfaces are defined in a standard way. This applies both to Grid-specific standards and to general IT management standards.
- **Security**: There are two security aspects in management:
  - Management *of* security: the security infrastructure must be manageable; this includes the management of authentication, authorization, access control, VOs and access policies.
  - Secure management: using the security mechanisms on management tasks. Management should be able to ensure its own integrity and to follow access control policies of the owners of resources and VOs.
- **Reliability**: A management architecture should not force a single point of failure. To make this possible, managers must be allowed to manage multiple manageable resources, and a manageable resource must be allowed to be managed by multiple managers.
- **Policy**: A management architecture must be able to enforce policy assertions that are put in place to support requirements and capabilities such as authentication scheme, transport protocol selection, QoS metrics, privacy policy, etc.
- **Performance Monitoring**: Performance monitoring facilities should satisfy the following requirements outlined in the Grid Monitoring Architecture:
  - Low latency to keep performance data relevant
  - Handle high data rates
  - Minimal measurement overhead
- **Peer-to-Peer Management Requirements**: Grid systems that comprise large peer-to-peer systems have the following general requirements, which apply also to manageability [11]:
  - Discovery: While discovery mechanisms are used in traditional distributed systems, membership of peer-to-peer systems is typically highly dynamic, and hence they rely even more heavily on discovery mechanisms being both efficient and effective.

- o Security: Some specific requirements are around community-based trust mechanisms, replication, and verification of user identities. User privacy and anonymity are also characteristics of such systems.
- o Location awareness: This is the capability of an application to take advantage of proximity – relative, absolute or contextual. This is important in providing location-based services or system-level optimizations.
- o Group support: Peer-to-peer systems allow for the creation and management of dynamic groups with large transient populations. Management must be able to create and manage dynamic user groups.

3.2 Levels

In an OGSA Grid there are three types of management that involve resources:

- Management of the resources themselves (e.g., rebooting a host, or setting partitions on a switch).
- Management of the Grid resources (e.g., resource reservation, monitoring and control).
- Management of the OGSA infrastructure, which is itself composed of resources (e.g., monitoring a registry service).

Different types of interfaces realize these forms of management. These interfaces can be categorized into three levels, shown in the middle column of Table 1, and also on the right in Figure 1.

Table 1: Relationships between types of management and interfaces in OGSA

Type of management	Level of interface	Interface
Management of the resources themselves	Resource level	WBEM, SNMP, etc.
	Infrastructure level	WSRF, WSDM, etc.
Resource management on the Grid	OGSA functions level	Functional interfaces
Management of OGSA infrastructure		Specific manageability interfaces

A detailed description of each level and its interfaces is given below. Note that the descriptions focus on the manageability interfaces, *not* on the locus of implementation (e.g., on the services that implement them). Also note that a service may implement multiple interfaces (which are possibly unrelated in terms of functionality), and that a service may be separated from the functionality that it represents (e.g., a manageability provider for a resource that is separate from this resource). Therefore a description based on services would be imprecise, and a description based on interfaces is chosen instead.

In Figure 1, the OGSA capabilities cover all levels, extending to capabilities in the resources that are needed to implement these OGSA capabilities. The interfaces are shown as small circles.

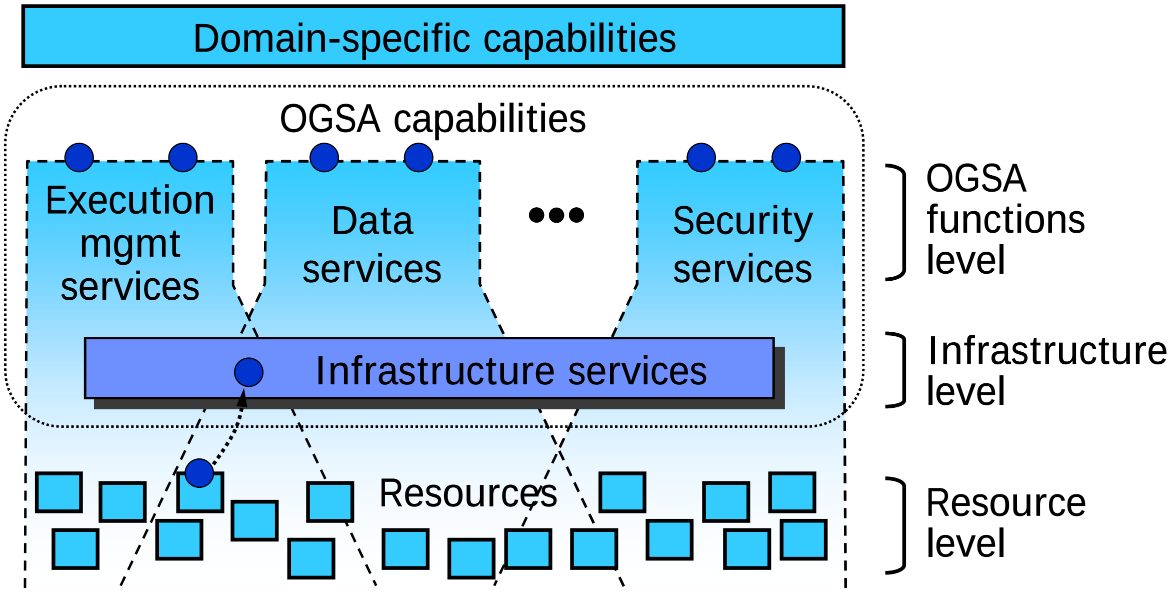

Figure 1: Levels of management in OGSA

At the resource level, the resources are managed directly through their native manageability interfaces. For discrete resources, these are usually SNMP (Simple Network Management Protocol), CIM/WBEM[2], JMX (Java Management eXtensions), or proprietary interfaces. Management at this level involves *monitoring* (i.e. obtaining the state of the resource, using techniques such as event notification), *setup and control* (i.e. setting the state of the resource), and *discovery*.

The infrastructure level provides the base management behavior of resources, forming the basis for both manageability and management in an OGSA environment. Standardization of this base management behavior is required in order to integrate the vast number and types of resources—and the more limited set of resource managers—that are introduced by multiple suppliers. The infrastructure level provides:

- The base manageability model, which represents resources as services and allows resources in OGSA to be manipulated through the standard Web services means for discovery, access, etc. This model allows the resources to become manageable at least to a minimum degree, by enabling discovery, termination, introspection, monitoring, etc.

  By using a single basic manageability model and possibly a single resource model, the identities, relationships (dependencies and component definitions), operational status, statistics, etc. become consistent throughout all layers of management, which improves interoperability. For instance, if a Grid node is reserved, an application is deployed on it, and the usage of this application is metered, then the identities used by reservation, deployment and resource usage services must be common and refer to the same entities if interoperability is to be possible.

  It is important to note that the base manageability model is *not* itself a resource model—the resource model of the resources themselves is accessed *through* the base manageability model. This is shown in Figure 1 by the arrow linking the interface at the resource level to the interface corresponding to this resource at the infrastructure level, showing the former as the latter.

---

[2] WBEM (Web-Based Enterprise Management) is a set of technologies developed to unify the management of enterprise computing environments. It is composed of CIM (Common Information Model), which defines the resource model semantics, and a set of encodings and protocols to access the resource model.

- Basic functionality that is common to the OGSA capabilities, e.g.:
  - Interfaces for capabilities that are common to many resources (e.g., start, stop, etc.).
  - Representation of the state graph of a resource, including the states and transitions, and operations to change the state.
  - Ways to describe and discover relationships among resources, including the types of the relationships (“contains,” “uses,” etc.).
  - Notifications.
- A *generic manageability interface* that is common to all services implementing OGSA capabilities. This manageability interface has functionality such as introspection, monitoring, and creation and destruction of service instances.

At the OGSA functions level there are two types of management interfaces, denoted by the two circles on the top of each of the capabilities shown in Figure 1:

- Functional interface: Some common OGSA capabilities (such as job management) are a form of resource management. Services that provide these capabilities expose them through functional interfaces.
- Manageability interface: Each capability has a specific manageability interface through which the capability is managed (e.g., monitoring of registries, monitoring of a job manager, etc.). This interface could extend the generic manageability interface, adding any manageability interfaces that are specific to the management of this capability.

A simple example of these interfaces for a job manager service is given in Figure 2.

The functional and manageability interfaces are often not clearly separated (especially in the case of resource managers). While overlap is inevitable, some distinction is desirable, since these interfaces are invoked by different users with different roles and access permissions. For instance, in Figure 2, the functional interface is used by the manager (or user) of the application being run (the “Grid administrator” in the Commercial Data Center use case [3]), and the manageability interface is used by the system manager (the “IT business activity manager” in [3]).

One way to logically separate the functional and manageability interfaces would be to create management “capabilities” such as those for Metrics, Operational Status, Relationships, Identity etc. as outlined in the WSDM MUWS specifications [4, 5]. This classification allows manageability interfaces to be functional interfaces as well, and it also enables access control policies to be set up at the interface/category level, based on roles and privileges.

Manageability is often an afterthought, so often the functional interface is present but not the manageability interface.

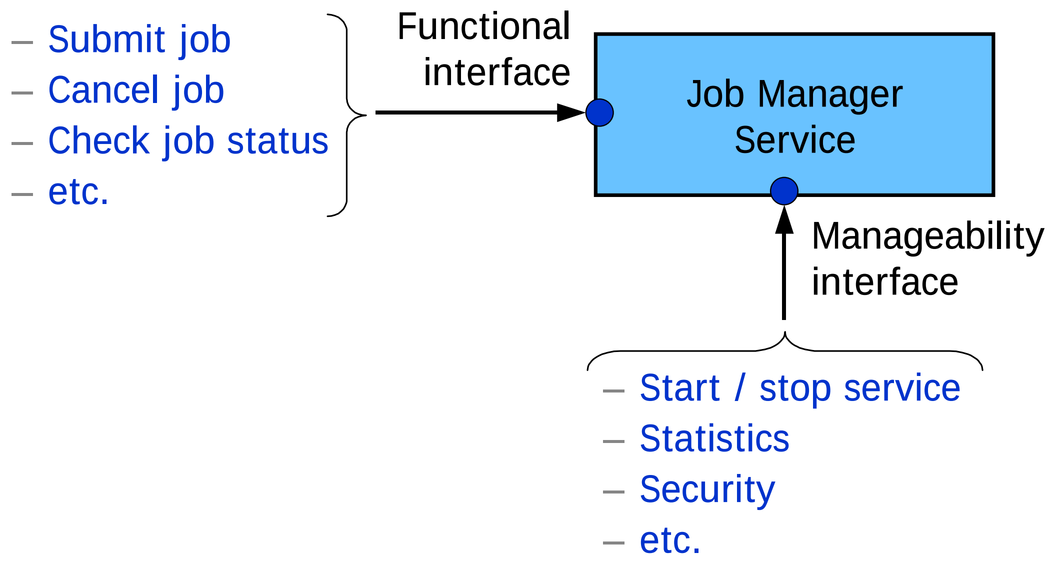

Figure 2: An Example of the Functional and Manageability Interfaces

The interfaces in both the infrastructure and the OGSA functions levels are provided by services, but they have a different nature. In the infrastructure level the services are a wrapper around the manageability of a resource (mainly the resource model semantics), which provide a means to access this manageability. In the OGSA functions level the services provide functionality at a level higher than the features of the resources, or provide interfaces that do not access a resource model.

Discovery provides a concrete example of the differences between the resource, infrastructure and OGSA functions levels. Discovery at the resource level might involve scanning a network to discover the devices attached to it. Discovery at the infrastructure level can involve introspecting the metadata (e.g., resource properties) of a service to find its capabilities. Discovery at the OGSA functions level might involve accessing one or more registries that contain references to the available resources.

The division in levels helps interoperability between levels by defining clear interfaces between them. While it is possible to build services (implementing OGSA capabilities) that bypass these levels (e.g., using a proprietary adapter in a resource that feeds data directly to the service), that is not desirable from the point of view of interoperability, because, for example, it limits the kinds of resources with which the service and the adapter will be compatible.

4. Resource Models

Resource models describe resources by defining their properties, operations, and events, and their relationships with each other. Resources are managed (monitored, allocated, etc.) by following the description given by the model, and therefore resource models are essential to all facets of resource management. Resource models are used for both the functional and manageability interfaces.

Two kinds of resource models exist:

- IT system management, used mostly for system monitoring and control
- Resource descriptions, used mostly for brokering, metering and provisioning

Examples of resource models are:

- SNMP MIBs (Management Information Bases), which cover mainly network management but are also used in other areas such as host management.

- JMX's JSR77, a resource model for the manageability aspects of the J2EE (Java 2 Enterprise Edition) platform [13].
- CIM, which includes models (*schemas*) for the following areas[3]:
  - Core: high-level abstractions (logical and physical elements, collections)
  - Physical: things that can be seen and touched (e.g., physical package, rack and location)
  - System: computer systems, operating systems, file systems, processes, jobs, diagnostic services, etc.
  - Device: logical functions of hardware (e.g., battery, printer, fan, network port and storage extent)
  - Network: services, endpoints/interfaces, topology, etc.
  - Policy: if/then rules and their groupings and applicability
  - User and Security: identity and privilege management, white/yellow page data, RBAC (Role-Based Access Control), etc.
  - Applications and Metrics: deployment and runtime management of software and software services
  - Database: properties and services performed by a database (addresses database components, backing storage, status and statistics)
  - Event: notifications and subscriptions
  - Interoperability: management of the Web-Based Enterprise Management (WBEM) infrastructure
  - Support: help desk knowledge exchange and incident handling
  - Security Protection and Management: notifications for and management of intrusion detection, firewall, anti-virus and other security mechanisms
  - Block and file storage
  - Application Server: updates JSR77's CIM mapping, for managing the J2EE environment
  - New work in the areas of Behavior and State (modeling state and transitions) and utility computing (management of utility computing services and related data for provisioning, accounting and metering, reservation handling, etc.)
- WSDM MOWS Web service model.
- Resource descriptions for reservation/brokering/scheduling:
  - UNICORE Resource Schema
  - Globus RSL (Resource Specification Language) and the GLUE (Grid Laboratory Uniform Environment) schema (see http://www.hicb.org/glue/glue.htm)
  - JSDL (Job Submission Description Language, being defined by GGF's JSDL-WG)
- Resource descriptions for accounting/metering:
  - Usage Record (defined by GGF's UR-WG)
- Resource descriptions for installation/deployment/provisioning:

---

[3] The work on JSIM (Job Submission Information Model, defined by GGF's CGS-WG) was added to the schemas of multiple areas.

- o Configuration Description Language (CDL, being defined by GGF's CDDML-WG)
- o Data Center Markup Language (DCML, see http://www.dcml.org)

Note that although some resource descriptions are not intended to be models by themselves, they contain an implicit model which defines, for instance, which entities exist, and what their attributes are.

As previously mentioned, as Grid technology is increasingly adopted across institutions and enterprises, the distinctions between Grid environments and traditional IT environments will blur. This applies also to resource models: the models used in traditional IT environments will increasingly be used in Grid environments and vice-versa. A good example of this is CIM, which is currently under consideration as the resource model to be used in OGSA. Due to its breadth and extensibility, CIM can be used in the management tasks of multiple OGSA capabilities such as security, execution management, self-management, etc.

In resource models, it is important to make a distinction between *semantics* and *renderings*. The semantics contain the concepts of the model (its entities, their properties and relationships). A rendering is a representation of the semantics in a given language, and/or a specification of how to transmit and access the model on the wire. A rendering of a model allows its semantics to be conveyed. For instance, the CIM model contains the semantics of resources, while its XML representation and HTTP mapping are a rendering of CIM. The semantics may have multiple renderings: for instance, other CIM renderings have been proposed.

The semantics of a resource model contain its meaning, and thus they are more important in achieving interoperability than its renderings: translating between two renderings of a single model is not a difficult problem, but translating between the semantics of two different models is likely to be complex. For instance, in different models a fan may be a physical or a logical entity; it may be classified under chassis, cooling devices, enclosure services or physical packaging; or it may have similar properties, such as a status, which have different value sets. Automatic translation between semantics can't be done unless these semantics are matched. An example of this matching is the mapping between Globus and UNICORE resources being done as part of the GRIP project [14] (see also http://www.grid-interoperability.org). Also, CIM has mechanisms to map its semantics to those of other resource models [15].

Ideally, the use of a single resource model is desirable, since it makes interoperability easier to achieve when compared to mediation between models. However, this is a utopia: developments in general IT and in Grid have so far not led to a total unification of the resource models, so it must be expected that multiple resource models will be in simultaneous use in a given Grid. Thus, coordination between models to make them compatible (as done with the GLUE schema), and mechanisms to match the semantics of different models, will have to be used. This is especially important for OGSA, in which the functionality of a Grid is formed by the composition of multiple capabilities—each of them possibly using multiple semantics and/or renderings—which have to interoperate.

It is desirable that new resource models are created by re-using existing models, which not only allows higher interoperability but also requires less work. For instance, this new resource model could be created as a subset or superset of another resource model. Or, multiple resource descriptions could be created as renderings of a single resource model (with each resource description language representing this model, or a subset of it, using its own syntax, e.g., its own XML schema).

There are two areas in which there is need for coordination between resource models:

- Between the *resource descriptions* (to ease interoperability between OGSA services—reservation, metering, provisioning, etc.).
- Between the *standard management* models and the resource descriptions (to ease interoperability between resources and their resource managers).

Another desirable direction for work on resource models is model neutrality on the mechanisms for resource management. This allows the unification of the mechanisms to use multiple resource models despite there being no unification on the models themselves. WSRF and WSDM are examples of these mechanisms.

5. Analysis of the OGSA Capabilities

The gap analysis has the objective of finding missing functionality on each level of manageability interface for each of the OGSA capabilities. Thus the gap analysis can be viewed conceptually as filling a table in which the rows are the management levels and the columns are the OGSA capabilities, as shown in Figure 3. Lack of (or insufficient) contents in a cell indicates a gap. However, the analysis of the functional interfaces is one of the tasks of the OGSA-WG, and therefore this gap analysis will only cover manageability (the base manageability, generic manageability, and specific manageability interfaces). When applicable, models are analyzed for each capability.

Capabilities / Levels	Execution mgmt services	Data services	●●●	Security services
Specific manageability I/F	(Section 5.3.1)	(Section 5.3.2)	●●●	(Section 5.3.7)
Generic manageability I/F	(Section 5.2)			
Base manageability	(Section 5.1)			
Models	(Section 5.3.1)	(Section 5.3.2)	●●●	(Section 5.3.7)

Figure 3: The Gap Analysis (conceptual view)

The gap analysis lists elements of a Grid that are candidates for management, and hence need to provide manageability interfaces. The list is intended to be used to identify the types of management actions that need to be possible, and the set of common manageability interfaces that are required. Some interfaces are expected to be defined already, while others will need to be specified. The list is derived in part from the OGSA document [1].

The OGSA capabilities are still being defined, as are many of their underlying specifications, and in some areas work has not progressed sufficiently to allow the analysis of their manageability to be completed. There are also cases where specifications have been completed, but it is not clear that they will be adopted. In such cases the gap analysis will point out items for which future analysis is required.

5.1 Base Manageability (Infrastructure Services)

5.1.1 Analysis

In an OGSA-based Grid, all manageable resources either *are* Web services or are *represented by* Web services. The assumed basis for representing resources is the interfaces and behaviors defined by the WS-Resource Framework (WSRF) family of specifications (see http://www.oasis-open.org/committees/wsrf). Furthermore, OGSA assumes the availability of WS-Notification (WSN) interfaces and behaviors for event notification (http://www.oasis-open.org/committees/wsn). Between them, WSRF and WSN specify a basic set of interfaces that are useful in management. For example, a suitably-privileged manager might make use of the following features, provided that they are implemented by the resources:

- WS-Resource Lifetime specifies operations that a manager can use to query the termination time of a resource, and to change it, possibly causing immediate termination.
- WS-Resource Properties provides a means for a resource to publish a list of its properties (the resource properties document), and for a manager to retrieve the document and to query and modify the values of the properties.
- WS-Resource Properties also defines a facility by which managers can request and receive notification when the value of a resource property changes. This facility is generally provided through the use of WSN notification messages.

The WSRF and WSN specifications, plus WSDM MUWS, will provide the core functionality for the base manageability interfaces. The following paragraphs enumerate the key features of each of the specifications, as they apply to resource management.

- **WSRF** specifications currently include WS-Resource Properties, WS-Base Faults, WS-Resource Lifetime, and WS-Service Groups. Together, these specifications provide the following key management features:
    - Resource representation: WSRF specifies the *implied resource pattern* to associate a Web service with a stateful resource.
    - Resource property values: Arbitrary resource properties can be discovered, queried and modified by suitably-privileged managers.
    - Monitoring capabilities: A manager can subscribe for asynchronous notification of any change to the value of a resource property. WSRF relies on WSN for notification support.
    - Resource lifetime management: A resource can be scheduled for automatic destruction at a specified time. The scheduled destruction time can be changed, and immediate destruction can be requested.
    - Service aggregation: Collections of services can be represented by *service groups*. Service groups may be useful for grouping services that belong to a specific management domain or some other organization, or that require specific management operations. A manager can request and receive notifications of changes to the membership of a service group. Note, though, that service groups do not explicitly provide for bulk operations on all members of the group.
    - Fault management: WS-Base Faults supports fault determination and management by providing a common way to specify Web services fault messages.
- **WSN** supports resource monitoring through topic-based event subscription, either directly from the notification source or from a broker. Notifications may indicate, for example, that a resource has been created or destroyed, or that the value of a resource property has changed. WSN specifications include WS-Base Notification, WS-Brokered Notification and WS-Topics.
- **WSDM MUWS** provides a foundation for management *using* Web services. The following functionalities are among those currently being investigated:
    - Identity: A unique identifier of the resource.
    - State: A base model for the state graph of the resource, with standard states, transitions, and operations to change the state. This state graph is extensible.
    - Metrics: Collected values from the resource (such as statistics), plus associated metadata (which allow the determination of type of metric, such as counter or gauge, the collection period, etc.), plus a reset operation.

- Configuration: representation and modification of the properties that influence the behavior of the resource.
- Notifications and events: A common rendering for events, containing base information that is common among multiple existing event formats.
- Relationships between resources: Related resources, and the kind of relationship (“contains,” “uses,” etc.).
- Collections: A grouping of resources that allows a manager to send an operation to all or part of the group, instead of operating each resource separately.
- Discovery: Ways to discover resources and their manageability.
- Correlatable names: Additional information on resources to allow two managers to determine if one of their resources is the same.

- **WSDM MOWS** builds on the MUWS specification for the management *of* Web services. The following MOWS functionalities are among those being investigated in addition to those in MUWS:
  - Identification: The Web service endpoint being managed.
  - Request processing state: The state of the processing of each request submitted to the Web service.

5.1.2 Gaps and Open Issues

The following gaps have been identified:

- Manageability functionality and possibly resource models need to be defined for the services in the infrastructure level:
  - It may be important to identify general factory services as such, so that they can be managed in the same way as other key infrastructure services. Manageability interfaces will be needed to query which services the factory can create, and also for monitoring state and performance.
  - WS-Agreement is part of the infrastructure services, and will be used in activities such as reservations and data access. Each of these is likely to have specialized interfaces, and may require specialized management. Their correct operation and performance will be critical to a Grid, and must be monitored.
- The WSDM TC is creating a Web service model in MOWS, and defining its mapping to MUWS. However, the TC’s charter does not include defining the mappings from MUWS to other models (e.g., CIM and SNMP, and Grid-related models). These mappings need to be defined (all further mentions of WSDM assume also resource models and their respective mappings). The WS-CIM working group in the DMTF (Distributed Management Task Force) is currently working on a mapping from CIM to Web services.
- There is research on mapping Grid-related models among themselves, and ways to map IT standards (e.g., CIM and SNMP) among themselves, but to the authors’ knowledge there is currently no work on mapping the Grid-related models with the general IT standard ones.

The following are open issues:

- It needs to be investigated whether the state model of WSDM is suitable for the OGSA capabilities—e.g., job control and provisioning.
- The original CMM plans included “canonical services factored out from across multiple resources or domain specific resource managers, such as an operational port type (start/stop/pause/resume/quiesce).” This specific interface (start/stop) can be realized by the canonical state operations of WSDM. The need for other sets of canonical interfaces should be investigated (they are not among the current planned functionality for WSDM).

- WSDM is model-independent, and therefore there is the need to choose a set of resource models to be used to allow minimum levels of interoperability. Given that agreement on a single resource model cannot be expected, probably a set of profiles will have to be defined. This will be a complex task, given the wide variety of resources (e.g. licenses) that will need to be addressed.
- One single resource might need multiple state graphs for multiple resource management activities (deployment state, running state, etc.). WSDM allows this functionality, but it needs to be verified if it is necessary.

5.2 Generic Manageability Interface

5.2.1 Analysis

Any service in OGSA will provide interfaces for at least minimal management—e.g. termination, introspection and monitoring. The OASIS WSDM TC will define standard manageability interfaces for Web services (MOWS) that should be applicable to services in OGSA.

5.2.2 Gaps and Open Issues

The following gaps have been identified:

- Security functions are pervasive, and some activities on the management of security should be common to all services. Examples of such activities are the management of access permissions to the service for different roles (end-user, managers, etc.) and of the protocol bindings to be used.
- Any service may suffer security incidents such as a denial-of-service attack, and ideally the manageability should provide a way to indicate such incidents as they occur. Such manageability interfaces may need to be defined for all services, including perhaps the manageability of resources under WSDM.

The following are open issues:

- It needs to be determined whether there are additional general interfaces beyond MOWS that are specific to an OGSA Grid.

5.3 Specific Manageability Interfaces

This section analyzes the specific manageability interfaces, plus the models that are specific to a given capability where applicable. The following sections analyze the capabilities at the OGSA functions level, i.e., all OGSA capabilities except for the Infrastructure services. For each capability we describe why it is important to manage its services, give an analysis of its manageability, and list any gaps and open issues.

5.3.1 Execution Management Services

5.3.1.1 Analysis

OGSA's execution management services (EMS) capability handles selection, reservation, and configuration of resources, and controls the execution of programs over them. This is a central functionality in Grids, and effective management is essential. EMS was analyzed from the point of view of manageability and our conclusions are as follows:

- The **job** and the **service container** are resources. They have a resource model that defines their capabilities and properties, which can be accessed through WSDM-compliant interfaces. The job document corresponds to manageability information of the job (i.e., attributes).

  In the case of a **service container** (e.g. a node), the resource itself has manageability interfaces for basic tasks such as monitoring resource consumption and halting execution. However, the resources are unlikely to directly realize interfaces for *all* the functionality that is needed (for example, they are unlikely to provide interfaces for deployment), so their manageability providers must realize the missing interfaces for them. Note that the manageability of a service container, whether provided directly or indirectly, is likely to be “close to” the actual resource—for example, in most cases it will be implemented and/or run on the resource itself.

Although a **job** is also a resource, it is significantly different from a service container. EMS specifies that a job is created *before* resources have been committed and the actual execution begins, so at the time of its creation it is not known which service container will execute it. Therefore, it is not possible to require that the manageability of a job be realized only through a provider that is close to the job’s execution resources, and the dynamic nature of job execution can make it difficult for the provider to keep track of the job’s location through its life. One approach to handling this issue might be for the job manager to contain the manageability providers, and to refresh the reference to a job as it is assigned to a container and on any subsequent migration etc.

- The **job manager, execution planning service, candidate set generator, information services, deployment and configuration services,** and **reservation services** are all services at the OGSA functions level.

  The resources (job and service container) have interfaces with functionality that *enables* the functionalities of services at the OGSA functions level, but they do not implement these functionalities themselves. For instance:

  - The resources have functionality to enable migration, but migration is actually implemented at the functions level.
  - The job manager provides interfaces to reschedule a job, or to perform batch operations on a set of jobs, but these are not capabilities of the jobs themselves.

5.3.1.2 Gaps and Open Issues

No gaps have been found.

The following are open issues:

- The resource models for the job and service container need to be defined. Existing resource models (CIM, GLUE, UNICORE Resource Schema, etc.) should be analyzed and re-used.
- An investigation is needed to determine whether the job manager should expose the same interface of the jobs to its clients (e.g., by extending the interface for the jobs), which would provide a “one-stop shop” to control jobs.
- The manageability provider for the job may not be “close to” the actual job. Further analysis is needed to determine whether an interface that is guaranteed to be close to the job (implemented by a manageability provider in the actual container) should also be defined. This would provide interoperability between the job managers and the containers.
- It is not clear whether the **persistent state handle service** is a resource or a service at the OGSA functions level.
- On manageability interfaces:
  - A manageability interface for job management exists (JSIM). Further analysis is needed to determine if the functionality of this interface is sufficient for the needs of the execution management services.

    For instance, a job manager needs to monitor the status of individual resource queues, and to be able to control them—e.g. to move jobs between queues to balance loads, to override priorities and to accommodate planned downtime. Also the job manager, as defined in OGSA, may manage workflows and arrays of non-interacting jobs, which could require specific manageability functionality, e.g. to identify and represent a set of job instances.

- o Other manageability interfaces, e.g. for controlling and monitoring the execution planning service, the candidate set generator, etc., will be needed.

5.3.2 Data Services

5.3.2.1 Analysis

The Data Services capability contains services that provide facilities for data access, representation, transformation, and management. In many Grids such services may be numerous and diverse; they will be fundamental to most, if not all, Grids. They will be critical infrastructure services, and their availability and performance must be monitored and managed.

At the OGSA functions level, the data services concern primarily the management of data—i.e., its provisioning and allocation, caching and replication, virtualization, etc. These are ultimately implemented through functional and manageability interfaces in the various devices involved, which are resources. It is worth mentioning that the data itself can be modeled as a resource, although the current data service proposal does not focus on this aspect of data.

5.3.2.2 Gaps & Open Issues

Currently an architecture for data services is being created by a design team in the OGSA-WG, and interfaces (both functional and manageability) are being defined by other GGF WGs.

At the current stage no gaps or open issues are apparent, but further progress on the architecture for the data services should allow a more detailed analysis of the manageability to be done. Especially, manageability should not be forgotten: as with other functionalities, the interfaces of the data services should allow the management of the infrastructure, e.g. monitoring the efficiency of caching and replication.

5.3.3 Context Services

5.3.3.1 Analysis

The context services currently comprise VO management and policy management. The current service descriptions do not allow a gap analysis to be performed, but an analysis of their manageability follows.

VOs are complex resources, and will provide significant management challenges. A manager will need to be able to discover and manage VO registries, create and destroy VOs, and manage the set of resources and users assigned to an individual VO. Given that VOs are a fundamental part of the concept of Grids, their management is essential.

The interface to manipulate VOs provides manageability functionality, such as creation and destruction of VOs, associating entities such as users, groups, and services with a VO, manipulation of user roles within the VO, and attachment of agreements and policies to the VO. Some sort of model could be needed for these interfaces. Existing models such as CIM schemas related to security and user management could be re-used.

A Policy subsystem, when fully defined, is likely to be composed of multiple related services, including a repository. The subsystem will be a critical infrastructure component of most Grids, and the ability to monitor it and to control certain elements will be essential.

Two kinds of manageability interfaces will be needed: one to manage (add, remove, change, etc.) the policies on the resources, and one to the policy repository to perform the management of policies themselves.

5.3.3.2 Gaps and Open Issues

As noted above, no gaps analysis is possible at this stage.

5.3.4 Information Services

5.3.4.1 Analysis

The OGSA information services focus on the monitoring part of resource management, including related activities such as the transmission and storage of this monitoring information. The information services contain discovery and logging services, two important components in resource management. The management of events is also expected to be classified here.

Discovery services are likely to be deployed in every Grid. As mentioned in Section 3.2, discovery occurs at many levels, and the functionality in the OGSA functions level consists mainly of registries. A service, including a resource represented as a service, must be registered in one or more registries so that it can be discovered, and so that its interfaces and capabilities can be queried. A primary registry service is likely to be the starting point for discovering and mapping, and hence managing, all resources in the Grid. It is important that registry services are available, and that they operate correctly, so managers will need to be able to monitor their operation and performance, and to create and destroy instances and copies as needed.

Logging services are essential infrastructure services, and they must be managed accordingly. It will be necessary not only to monitor their performance, but also to deal with storage space thresholds, low-space or insufficient-space conditions, periodic purging, access control, and many other facets. Different management domains within a given Grid may have different policies for retention etc. It's likely that this will be one of the more complex management operations. The logging services will be defined by a new working group currently in formation in the GGF.

One of the central points of the information services is the consumer and producer interfaces, which provide a unified way to publish and retrieve monitoring information in a Grid. WSDM meets the base requirements of the consumer and producer interfaces, such as model-neutrality and extensibility. However the consumer and producer interfaces assume richer functionality, such as a push model to send data from the producer to the consumer; persistent storage of monitoring information, including queries to retrieve it and the setting of retention periods; and the aggregation of information and computation of metrics (statistical functions, such as a mean across resources, time, etc). It should be possible to implement at least part of this functionality by extending the WSDM interfaces, which makes WSDM a candidate for basis of the consumer and producer interfaces.

The information services implicitly assume a messaging and queuing service as the basis for information delivery, and it is likely that these services will become critical infrastructure services. Management requirements will include monitoring performance and managing the number of available instances and copies to handle the message volume and, if applicable, storage space.

5.3.4.2 Gaps and Open Issues

The following gaps have been found:

- Manageability interfaces for registries and for the messaging and queuing services, and possibly simple models to represent their manageability, will be needed.
- If specialized notification and event services are defined they will need to be managed as critical infrastructure services.

The following are open issues:

- It is still not clear how the producer and consumer interfaces relate to the WSDM interfaces, including their respective roles in monitoring in OGSA. This will need to be resolved as the specification of the producer and consumer interfaces is defined.
- A manageability interface is needed for logging for management tasks such as setting the retention period, erasing logs, etc. The current proposal for the logging interfaces contains some of these functions, but it needs to be determined if they are enough. The same comment applies also to the producer and consumer interfaces.

- It needs to be investigated whether events unique to Grid environments exist.

5.3.5 Resource Management Services

5.3.5.1 Analysis

The resource management services, despite the name, provide only part of the functionality for resource management in OGSA. A possible classification criterion for these services is that they should be applicable to most types of resources. This implies that reservation, provisioning and metering fall in this category (e.g., bandwidth, which is a resource, can be reserved; data, also a resource, can be deployed; the usage of licenses, also a resource, can be metered). This is in contrast with the execution management services and data services, which are primarily concerned with execution and data. The resource management services are at the OGSA functions level.

The CDDLM working group will address how to describe configuration of services, deploy them in a Grid, and manage their deployment lifecycle (instantiate, initiate, start, stop, restart, etc.). Managers will need the ability to configure, deploy, redeploy (relocate, perhaps with a different configuration) and terminate applications and other types of services within Grids, using the interfaces defined by CDDLM. Installation and provisioning may be separate issues.

The metering service is effectively an infrastructure service. It must be permanently available if resource usage is to be recorded and charged for, and hence the manager must be able to monitor and control its operation as for any other critical service. Accounting, billing and payment services are not part of OGSA, but are built over the capabilities of the metering service.

5.3.5.2 Gaps and Open Issues

The following gaps have been found:

- Provisioning needs to cover all kinds of resources, from hosts and services up to licenses, bandwidth and data. The work of the CDDLM-WG is currently focusing on services, and should be extended to cover other kinds of resources.

The following are open issues:

- The relationship between metering, the information services and WSDM has to be analyzed. This includes resource models, which can be different for metering and the information services.

5.3.6 Self-management services

5.3.6.1 Analysis

The self-management services configure, heal and optimize IT systems, following policies and/or meeting service level agreements (SLAs). The definition of the services and mechanisms to provide this functionality is still in preliminary stages and does not yet allow a gap analysis to be done.

It is expected that the self-management services will not be a centralized and monolithic service, but a set of services in multiple levels. For instance, to meet a given SLA on a database under increasing load, a manager might optimize its storage; if this is not enough, a manager at a higher level may add one or more nodes to this database; if not enough yet, a manager at a still higher level could distribute accesses among multiple sites. Due to this hierarchical nature, managers will need interfaces to report status and to receive commands and SLA parameters.

5.3.6.2 Gaps and Open Issues

As noted above, no gap analysis has been performed at this stage.

5.3.7 Security Services

5.3.7.1 Analysis

The security services that compose OGSA (and their interfaces) are currently being defined by the OGSA-WG.

Services such as authentication and authorization will need to be managed, and may need specialized manageability interfaces.

5.3.7.2 Gaps and Open Issues

There is currently no discussion on models, which is thus an open issue. However, the schema (and the knowledge on manageability behind it) in existing models such as CIM (e.g., the User and Security schema, and the Security Protection and Management schema) could be useful for (and used in) the manageability for security. Specifically, CIM has a basic model and ontology for describing identities/principals, credentials, roles, privileges, and authentication and authorization rules.

6. Conclusion

This document contains a discussion of the issues of management that are specific to a Grid, and especially to OGSA. We first defined the terms and describe the requirements of management as they relate to a Grid, and we then discussed the individual interfaces, services, activities, etc. that are involved in Grid management, including both management *within* the Grid and the management *of* the Grid infrastructure. We concluded with a comprehensive gap analysis of the state of manageability in OGSA, which is intended to serve as a foundation for future work.

6.1 Summary of Gaps

Two main patterns surface from the gap analysis, as follows.

First, currently there is not enough manageability functionality in OGSA. This functionality needs to be defined, since it is essential to provide systems that are more flexible, self-managing, or have lower management burden—attributes in which Grid technologies are expected to bring improvements.

Second, OGSA is defining new entities, such as the jobs and containers of EMS or the VOs, and these entities will need resource models. These models should be defined based on the guidelines of Section 4—e.g., through re-use of existing models.

6.2 Future Work

Many of the OGSA capabilities, their inter-relationships, and the standards on which OGSA is based are currently in evolution. This work takes a snapshot of their current state as the basis for the gap analysis. This work needs to continue to follow refinements and evolution in OGSA and related standards. It is hoped that this work in its current state serves as guidance for this refinement and evolution.

The analysis related to the gaps found, e.g., the definition of interfaces and models and analyses of whether existing functionality is sufficient, is better done by the groups responsible for the respective areas, or together with these groups, since they have the requisite knowledge. Also, this work should be done as an integral part of the definition of each capability, so that manageability in OGSA will not be an afterthought. It is expected that some of the expertise needed to define the models will need to come from other groups, e.g. from the CGS-WG or from the DMTF.

7. Security Considerations

As mentioned in Section 3.1, security is among the main requirements on management. Security is one of the many management functionalities covered in this document.

Author Information

Editor:
Frederico Buchholz Maciel
Hitachi America, Ltd., Research & Development Division
750 Central Expressway, MS: 3224
Santa Clara, CA 95050
Phone: +1 (408) 309-4304
E-mail: Fred.Maciel@hds.com

Contributors: Jem Treadwell, Latha Srinivasan, Andrea Westerinen, Ellen Stokes, Heather Kreger and David Snelling.

Thanks to Hiro Kishimoto, Bryan Murray, Jay Unger, Ravi Subramaniam, Lawrence Flon, Jeffrey Frey and Bill Horn, and also to the members of the OGSA-WG, for discussions that helped improve the contents in this document.

Glossary

The definitions in Section 2 provide a brief glossary of OGSA management terms. Refer to the OGSA Glossary of Terms [2] for further definitions of related terms. Acronyms are defined in the text.

Intellectual Property Statement

Full Copyright Notice

References

1. I. Foster et al., “The Open Grid Services Architecture, Version 1.0,” Global Grid Forum, Lemont, Illinois, U.S.A., GFD-I.30, 2005, http://www.ggf.org/documents/final.htm
2. J. Treadwell (Ed.). “Open Grid Services Architecture Glossary of Terms,” Global Grid Forum, Lemont, Illinois, U.S.A., GFD-I.44, 2005, http://www.ggf.org/documents/final.htm
3. Igor Sedukhin (Ed.). "Web Services Distributed Management: Management of Web Services (WSDM-MOWS) 1.0, Committee Draft," OASIS, December 2004, http://www.oasis-open.org/apps/org/workgroup/wsdm/download.php/10567/cd-wsdm-mows-1.0.pdf
4. William Vambenepe (Ed.). “Web Services Distributed Management: Management Using Web Services (MUWS 1.0) Part 1, Committee Draft,” OASIS, December 2004, http://www.oasis-open.org/apps/org/workgroup/wsdm/download.php/10558/cd-wsdm-muws-part1-1.0.pdf
5. William Vambenepe (Ed.). “Web Services Distributed Management: Management Using Web Services (MUWS 1.0) Part 2, Committee Draft,” OASIS, December 2004, http://www.oasis-open.org/apps/org/workgroup/wsdm/download.php/10557/cd-wsdm-muws-part2-1.0.pdf
6. Geoffrey Fox, David Walker, "e-Science Gap Analysis," National e-Science Centre, UK e-Science Technical Report UKeS-2003-01, June 2003, http://www.nesc.ac.uk/technical_papers/UKeS-2003-01/GapAnalysis30June03.pdf
7. Geoffrey Fox, David Walker, "Appendix : UK Grid Services and Activities, e-Science Gap Analysis," National e-Science Centre, UK e-Science Technical Report UKeS-2003-01 (appendix), June 2003, http://www.nesc.ac.uk/technical_papers/UKeS-2003-01/Appendix30June03.pdf
8. Susan Malaika, Andre Merzky, Reagan Moore, "GGF Data Area Structure and Function Analysis," Global Grid Forum, February 2004, http://www.gridforum.org/Meetings/ggf10/GGF10%20Documents/ggfdataarea.structureandfunctionanalysis.Feb2004.pdf
9. B. Tierney et al., "A Grid Monitoring Architecture," Global Grid Forum, Lemont, Illinois, U.S.A., GFD-I.7, January 2002, http://www.ggf.org/documents/final.htm.
10. R. Aydt et al., "A Simple Case Study of a Grid Performance System," Global Grid Forum, Lemont, Illinois, U.S.A., GFD-I.8, May 2002, http://www.ggf.org/documents/final.htm.
11. Karan Bhatia, Per Brand, Sergio Mendiola, “Peer-To-Peer Requirements on the Open Grid Services Architecture Framework,” forthcoming in the GGF document series, 2005, http://www.ggf.org/documents/final.htm.
12. I. Foster, D. Gannon, H. Kishimoto, J. J. Von Reich (Ed.). "Open Grid Services Architecture Use Cases,” Global Grid Forum, Lemont, Illinois, U.S.A., GFD-I.029, 2004, http://www.ggf.org/documents/final.htm
13. Hans Hrasna, “Java 2 Platform, Enterprise Edition Management Specification,” Java Community Process, JSR-77, June 2002, http://jcp.org/jsr/detail/77.jsp

14. J. Brooke, D. Fellows, K. Garwood, C. Goble, “Semantic Matching of Grid Resource Descriptions,” UoM. 2nd European Across-Grids Conference (AxGrids 2004), January 2004, Cyprus, http://www.grid-interoperability.org/semres.pdf
15. “Common Information Model (CIM) Specification, Version 2.2,” Distributed Management Task Force, 1999, http://www.dmtf.org/standards/documents/CIM/DSP0004.pdf