Computing and Informatics, Vol. 26, 2007, 1–26, V 2007-Jan-29

UNDERSTANDING SEMANTIC AWARE GRID MIDDLEWARE FOR E-SCIENCE

Pinar Alper, Oscar Corcho, Carole Goble

School of Computer Science, University of Manchester, Manchester, UK
e-mail: {penpecip,ocorcho,carole}@cs.man.ac.uk

Abstract. In this paper we analyze several semantic-aware Grid middleware services used in e-Science applications. We describe them according to a common analysis framework, so as to find their commonalities and their distinguishing features. As a result of this analysis we categorize these services into three groups: information services, data access services and decision support services. We make comparisons and provide additional conclusions that are useful to understand better how these services have been developed and deployed, and how similar services would be developed in the future, mainly in the context of e-Science applications.

Keywords: Semantic Grid, Middleware, e-Science

1 INTRODUCTION

The Science 2020 report [40] stresses the importance of understanding and managing the semantics of data used in scientific applications as one of the key enablers of future e-Science. This involves aspects like understanding metadata, ensuring data quality and accuracy, dealing with data provenance (where and how it was produced), etc. The report also stresses the fact that metadata is not simply for human consumption, but primarily used by tools that perform data integration and exploit web services and workflows that transform the data, compute new derived data, etc. This vision of e-Science is also presented in [41], where the authors foresee a high degree of seamless automation, with flexible collaborations and computations on a global scale. Here the Semantic Grid is defined as an extension of the current Grid where information and services are given well-defined meaning, better enabling computers and people to work in cooperation, and is considered as a suitable approach for the new challenges. In such an environment, the use of explicit knowledge and

metadata underpins sharing and brings interoperation, flexibility and extensibility to Grid systems.

System development in the Semantic Grid has evolved from a set of pioneer applications, with ad-hoc developments, to a phase of systematic investigation, with more logical, consistent and ordered approaches and where know-how transfer to middleware and applications has started. A non-exhaustive list of examples of systems developed in the first exploratory phase is: Geodise [34], ^{my}Grid [35], Comb-e-Chem [24], and CoAKTinG [38]. All of them use semantic annotations of resources to fulfill their requirements (exposing metadata, reasoning with knowledge and metadata, etc.). Besides, they all use semantic technologies, but these technologies are not especially adapted to be used in Grid middleware platforms and applications.

Another non-exhaustive list of systems, most of them under development, in the second phase is: caGrid [15], BIRN [16], the Semantic Reef [37], the OntoGrid's QUARC system [36], and GEON [39]. These systems have been developed following a more systematic approach, are more tightly integrated with existing Grid infrastructure and in some cases they have been developed according to any of the Semantic Grid architectures recently proposed, such as S-OGSA [19], InteliGrid's architecture [42] and NextGrid's architecture [43].

In the second phase several Grid middleware systems have been also partially or completely re-implemented with semantic technologies, or have been wrapped so that they provide semantic-aware interfaces. This paper is specifically focused on these services, which we call **Semantic-Aware Grid Middleware Services (SAGMS)**. In the following section they will be analysed and compared, delving into their role in Grid middleware, their interplay with services that give them semantic support, and the means with which they have been developed.

The advantage of focusing only on middleware services is that we do not have to make assumptions about the usually complex domains of complete Semantic Grid applications, what could hinder the descriptions and results presented in the paper. Middleware systems are easily understood and have clearly-defined specifications and functionalities. This study can be then extrapolated to complex applications.

In summary, the purpose of this paper is to provide insight onto the features, commonalities and differences of different Semantic-Aware Grid middleware services, including the design decisions behind them. We try to use these experiences to guide future system designs and developments.

The paper is structured as follows: Section 2 gives some background about Grid architectures and specifications, which will be useful to understand the system descriptions provided later. Section 3 describes the analysis framework used to describe and compare the Semantic-Aware Grid Middleware Services (SAGMS) selected. This framework is not aimed at evaluating the decisions made, but for understanding such decisions. Section 4 describes each of the systems that are part of our study. Section 5 summarises the results of applying the framework to these services, categorises them according to their functionalities, and provides comparisons between them. Finally, section 6 summarises the work done and the results obtained.

2 BACKGROUND: GRID SYSTEMS AND THE OPEN GRID SERVICE ARCHITECTURE

In the end of 2000, the Grid community formed the Global Grid Forum (now Open Grid Forum -OGF-) to lead their standardisation efforts. The convergence between Web Services and Grid computing led to the proposal of the Open Grid Service Architecture specification (OGSA) [12]. OGSA defines the Grid architecture by outlining the requirements of the Grid (based on the needs of relevant use cases), and identifying the major groups of service capabilities to deliver these functionalities.

OGSA identifes six **service groups**: data (concerned with the management and transfer of data resources), resource management (concerned with the management of physical and logical resources and services, and of Grid infrastructure), execution management (concerned with the instantiation, management and completion of units of work), security (concerned with the enforcement of security-related policies within a virtual organization), self-management (concerned with resource self-configuration, self-healing and self-optimization) and information (which acts as databases of attribute metadata about resources).

Some cross-cutting OGSA **requirements** are: interoperability and support for dynamic and heterogeneous environments, resource sharing across organizations, optimization, Quality of Service (QoS) assurance, job execution, data services, security, administrative cost reduction, scalability, availability, and ease of use.

OGF actively works on key infrastructure services and protocols that make up a foundation for higher level OGSA capabilities. These foundational standards are related to naming, security, state, notification, transactions and orchestrations. Specifications can be found for all of them, among which we can highlight those related to state. Two alternatives exist for supporting stateful resources: WSRF [13] (Web Services Resource Framework), which is composed of four specifications: WS-ResourceProperties, WS-ResourceLifeTime, WS-ServiceGroup, and WS-BaseFaults, and is being standardised by OASIS; and WS-Management, composed of WS-Transfer, WS-Eventing, WS-Enumeration, and WS-ManagementCatalog, and proposed by a group of companies, including Microsoft, Intel, Dell, AMD and Sun Microsystems. The most relevant OGSA-observant middleware platforms are: gLite [2], UNICORE [3], Globus [1] and OMII [4]. They all provide a set of middleware functionalities, in the form of Web or non-Web service interfaces, which can be classified in any of the aforementioned OGSA service groups.

3 AN ANALYSIS FRAMEWORK FOR SEMANTIC-AWARE GRID MIDDLEWARE SERVICES

In this section we describe our framework for the analysis of Grid middleware services implemented or extended with semantic technologies. As aforementioned, this analysis framework does not aim at ranking these implementations, but at providing insight on the design and implementation decisions taken to build them, so that lessons learned can be derived and used in similar future developments.

The analysis is based on a two-dimensional set of features, shown in Table 1. The first dimension focuses on aspects related to Grid Middleware, Knowledge and Metadata. For each feature group, we focus on its most relevant aspects (WHAT) and on how these are implemented (HOW), as follows:

- In the Grid Middleware dimension we analyse the service purpose, including its (added-value) functionality, the standards used and the status of the software, how the service was (re-)developed, the standards used or extended, etc.
- In the Knowledge dimension, we focus on the semantic capabilities used, on their lifecycle (from its creation to its use and disposal), and on their role and coverage in the system. Besides, we analyse how knowledge entities are accessed and used, and which representation languages are used to express them.
- Similarly, in the Metadata dimension we analyse the metadata capabilities used, their lifecycle and their role and coverage, and we analyse how metadata is accessed and used, and the representation languages used to express it.

	WHAT	**HOW**
Grid Middleware	Purpose	System Development
Knowledge	Knowledge Entities	Knowledge Technologies
Metadata	Metadata Entities	Metadata Technologies

Table 1. Dimensions for the analysis of Semantic-Aware Grid Middleware Services.

For each dimension, the framework proposes the set of questions or discussion topics that are identified in the following sections.

3.1 GRID MIDDLEWARE FEATURES OF SAGMS

- *Purpose of the system.* In this category we identify the functionalities of the system from a Grid perspective. The items in this category are:

  1. **OGSA Service Group,** namely Data, Security, Information, Execution, Resource Management, and Optimization.
  2. **Functionality.** The exact functionality that the system performs in the context of the aforementioned service group.
  3. **Added-Value Functionality.** Does the service provide new functionality that did not exist before or is it providing better or flexible ways to deliver a previously existing functionality?
  4. **Implementation Standards.** The standards that the system implements (e.g. OGF models and protocols).
  5. **Software Status.** The current status of the software (production, beta, alpha, etc.).

- *System Development.* Here we identify how the service has been developed.

1. **Approach to Obtain Semantic Awareness.** Has the system been developed from scratch or by re-working or extending an an existing Grid component to make it semantically-aware?.
2. **Re-Working method.** In the case of re-working there are several approaches: **a.** Non-intrusive extension (e.g. client side wrapping). **b.** Minimally intrusive extension (e.g. interface extensions that might require a re-compilation without changing the code of the original Grid component). **c.** Intrusive extension (e.g. extending protocols or changing code).
3. **Extensions to Standards.** Has any standard been extended during the refactoring? These extensions are normally done using the extensibility points that most Grid middleware standards provide (e.g. xsd:any fields in most OGF specs), which in the case of SAGMS could be used to represent or carry semantic add-ons.
4. **Service Orientation.** Is a service-based approach adopted to deliver the service functionality and access metadata and knowledge?

3.2 KNOWLEDGE MODEL FEATURES OF SAGMS

- *Knowledge Entities.* We capture the knowledge requirements of the service.

1. **Required Knowledge Models**. Systems can operate over a pre-specified knowledge model (i.e. a specific ontology), be agnostic to the knowledge models (the system is generic and can use any), or not use any form of knowledge model.
2. **Knowledge Model Lifecycle**. If it exists, the change rate of the knowledge model and the type of changes (e.g. mostly additions, rarely updates etc.).
3. **Knowledge Model Coverage**. The coverage, focusing on whether they are models of a domain or they are used to represent dynamic aspects such as conditions and control of the system (e.g. business rules).
4. **Knowledge Model Generation.** Whether the service supports the knowledge provisioning process or it is agnostic to it (that is, it assumes that they have been created, no matter by whom). In the first case, we focus on the nature of the generation process: automatic, semi-automatic, manual, etc.
5. **Dependency levels to knowledge.** A service can have different dependency levels to knowledge, such as mission critical or optional.
6. **Security.** Whether any security requirements can be applied for accessing and using knowledge.

- *Knowledge Technologies.* In this category we identify the knowledge technologies used in the service design and deployment.

1. **Knowledge Representation Languages**. Which languages and associated inference techniques have been used within the service implementation. Examples are RDFS [31], OWL-DL [30], SWRL [27] etc.

2. **Knowledge Management Tools and Infrastructures.** The knowledge management tools used by the service. Examples are reasoners and rule engines (e.g., FaCT [21], Jess [22]), RDF libraries (e.g., Sesame [46]), Jena [8]), OWL2Java source code generators (e.g., Jastor [23]), etc.
3. **Knowledge Model Access.** The means to access and use knowledge models, which can be local or remote, and use Grid-compliant tools or not. For example, an ontology available in a URL can be accessed and used remotely with specialized ontology services or downloaded and used locally.

3.3 METADATA FEATURES OF SAGMS

The features in this dimension are very similar to those in the knowledge dimension:

- *Metadata Entities.* We capture the metadata requirements of the service.

  1. **Metadata based on a Knowledge Model.** Metadata can be based on a knowledge model or not.
  2. **Metadata Lifecycle.** The types of changes that metadata suffers (frequent or non-frequent, additions, updates or removals, etc.).
  3. **Metadata Coverage**. How much system information metadata represents (current state of the resources used in the application, data for condition checking, for application business logic, etc.).
  4. **Metadata Generation.** Whether metadata is generated by the system or externally, and what is the generation process: automatic, semi-automatic, manual, etc.
  5. **Dependency levels to metadata.** A service can have different dependency levels to metadata, such as mission critical or optional.
  6. **Security.** Whether any security requirements can be applied for accessing and using metadata.

- *Metadata Technologies.* We identify the metadata technologies used in the service design and deployment.

  1. **Metadata Representation Languages.** The languages used within the service implementation for the representation of metadata. Examples are RDF [32], XML [45], etc.
  2. **Metadata Management Tools and Infrastructures**. The metadata management tools used by the service. Typical examples are RDF libraries (e.g., Sesame, Jena).
  3. **Metadata Access.** The means to access and use metadata (local or remote), and whether Grid-compliant tools are used or not.

4 DESCRIPTION OF ANALYSED SEMANTIC-AWARE GRID MIDDLEWARE SERVICES

The list of SAGMS selected for our analysis contains some systems developed externally to our group, such as S-SRB [44], GRIMOIRES [5], CaBIG data access services [15] and S-MDS [29], and systems developed by our group, such as Semantic-OGSA-DAI [33], Grid Meta-Scheduling Service [7] and an ontology-based Role-Based Authorisation Component [18].

Other Grid middleware services use explicit knowledge and metadata to provide their functionalities, or their interfaces have been enhanced to use explicit knowledge and metadata. Our selection is based on the usage of some of these systems (GRIMOIRES and CaBIG data access services are used in production Grids) and in the representativeness of the others (we cover the range of OGSA service groups where knowledge and metadata has been applied).

All descriptions follow the same structure. First, we provide an overview of the system, describing what the system consists of, where the system is being applied, etc. Then we describe the role of metadata in the system, and how it is dealt with. Then we talk about the knowledge models that these systems use for describing metadata. Finally, we provide an architectural description of the system once that it has been enhanced with semantics.

4.1 SEMANTIC SEARCH ENGINE FOR THE STORAGE RESOURCE BROKER (S-SRB)

The Storage Resource Broker (SRB) is a widely-used system for handling shared data collections distributed across multiple organizations and heterogeneous storage systems. As part of the SRB toolkit, the Metadata Catalog system (MCAT) enables data discovery by providing support for storing, indexing and querying metadata about the data items stored in SRB.

MCAT is based on a relational model and supports textual and attribute-value pair based metadata, which is normally enough for most applications. However, in some cases there is a need for improved search mechanisms that take into account more information about the application domain or allow more complex queries that relate different pieces of information. This is the purpose of S-SRB [44].

In S-SRB metadata is represented using the RDF language [32]. S-SRB does not impose a specific metadata knowledge model, so that any domain knowledge model can be used, depending on the application that will use it. These domain knowledge models have to be expressed in the OWL ontology language [30].

S-SRB has been built in as a non-intrusive extension of the existing SRB tooling, as shown in Figure 1. The system is composed of a semantic server backend and a web-based client interface. The backend provides access to knowledge models and metadata, and to their associated reasoning functions, using the Jena API [8] and the Pellet reasoner [25]. It can be populated via programmatic ways or using the

client interface (an extension of mySRB), which allows uploading domain knowledge models and transforming MCAT conventional metadata into semantic metadata.

The client interface also allows querying metadata. Users build data search requests with the predicates and restrictions defined in the corresponding knowledge models. Those requests are executed by the semantic server backend, which uses instance reasoning (a.k.a. A-Box reasoning) provided by the Pellet reasoner. Then the results are returned to the client.

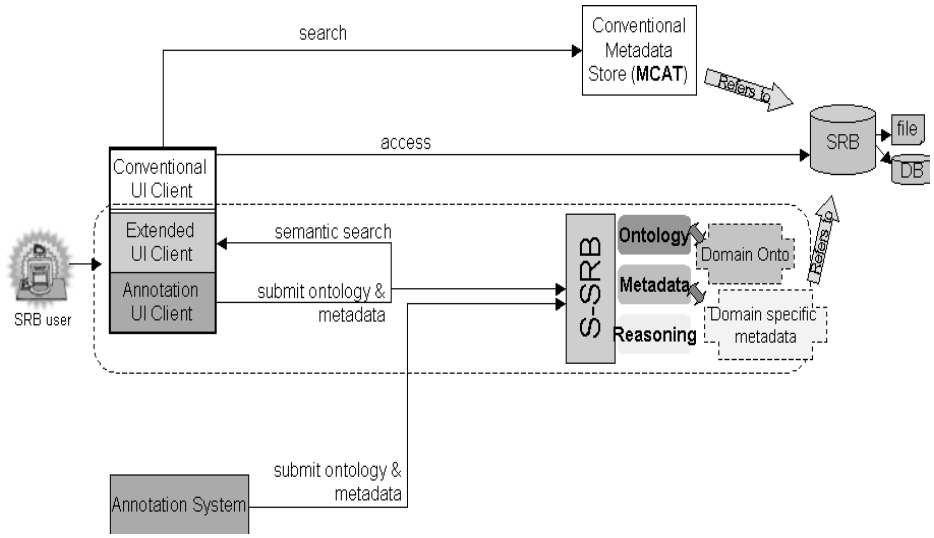

Fig. 1. S-SRB architecture.

4.2 GRIMOIRES Service Registry

GRIMOIRES [5] stands for Grid RegIstry with Metadata Oriented Interface: Robustness, Efficiency, Security. It is a metadata enabled UDDIv2-compliant [28] web service registry, distributed as part of the OMII Grid middleware suite. GRIMORES extends the UDDI information model and interface to support the WSDL model for describing service interfaces.

In GRIMOIRES metadata can be attached to entities in the UDDI and WSDL models, in the form of key-value pairs or sets of RDF triples. If no metadata is attached, then it acts as a plain UDDI service registry. The metadata lifecycle can be managed with WSRF-compliant interfaces, and security is taken into account in terms of user authentication and authorization at the operation level (e.g. User A cannot submit a description, User B cannot delete, etc.).

GRIMOIRES does not impose a specific knowledge model to structure metadata. It only imposes the basic domain-independent UDDI and WSDL models over service descriptions, which can be extended with domain-specific parts of service information. The mechanism to populate and query the GRIMOIRES registry is through controlled API calls, as shown in Figure 2, although support for freeform RDQL queries [47] is also planned for the future.

The GRIMOIRES backend is implemented entirely in RDF, in contrast to other hybrid models where UDDI registries are supported by relational backends and enhancements are attached as architectural add-ons [10, 9]. The choice for a native implementation is to avoid the additional communication costs and architectural complexity of hybrid systems. GRIMOIRES uses Sesame and Jena to manage RDF.

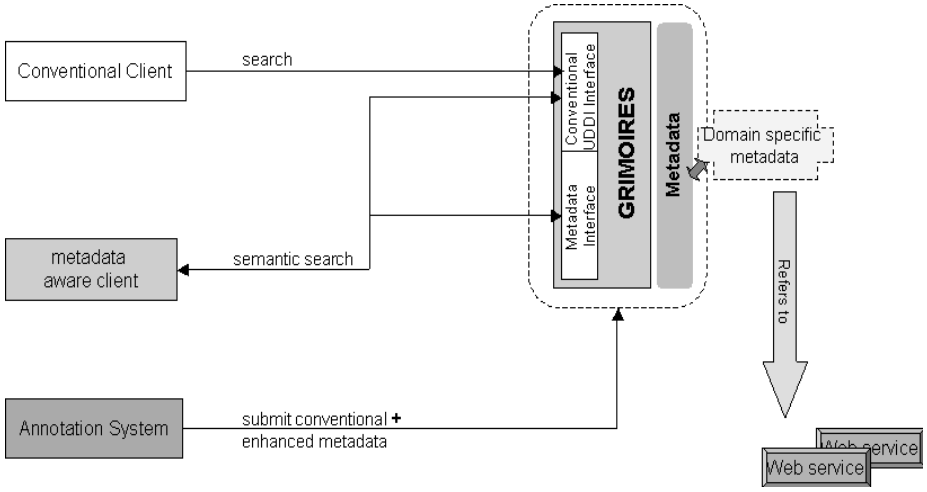

Fig. 2. GRIMOIRES architecture.

4.3 SEMANTIC MONITORING AND DISCOVERY SERVICE (S-MDS)

The Globus Toolkit Monitoring and Discovery System (MDS) is an information service that consists of two WSRF services for discovering and monitoring resources. The Index Service provides query/subscription interfaces to resource data, and the Trigger Service can be configured to take action when specific conditions are met in the data collected. As in the previous systems, metadata enables improved search mechanisms that include domain knowledge and use complex queries. S-MDS [29] extends MDS to create, aggregate and maintain WS-Resource semantic metadata.

The semantic metadata of a WS-Resource is created from its XML-based WS-Resource properties and transformed into RDF, by a semantic metadata provider (aka annotation service), which is associated to each WS-Resource. The metadata provider also monitors the WS-Resource, by polling the information from the property values or by implementing a notification-producer interface, so that the metadata is always up to date.

Two types of semantic models are used to generate and store semantic metadata in S-MDS: OWL-S and domain-specific ontologies. The first allows describing services associated to resources, while the second is application-dependent and allows classifying each resource according to its functions.

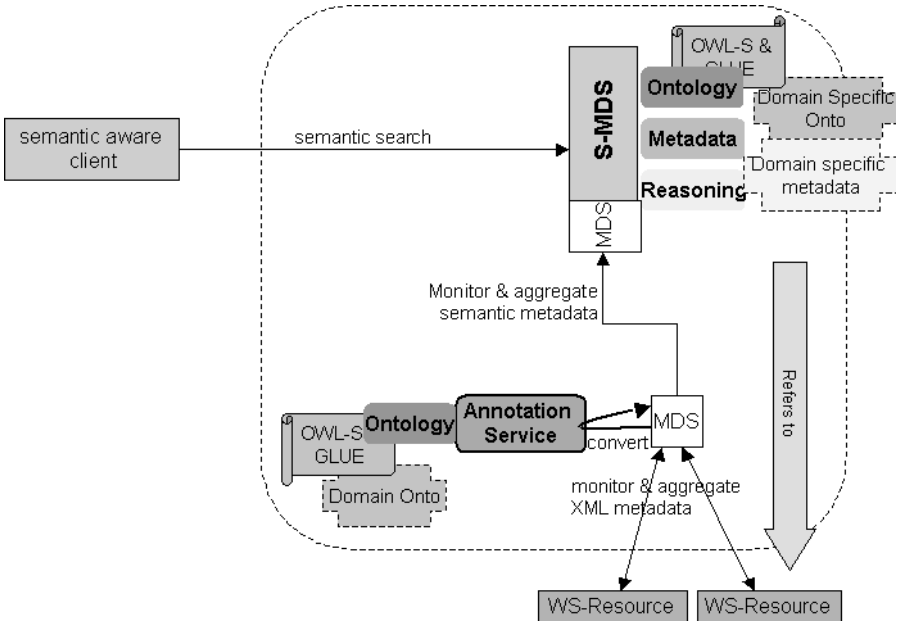

Fig. 3. S-MDS architecture.

S-MDS is implemented using the Globus Toolkit 4 aggregator framework. Besides obtaining semantic metadata from WS-Resource properties, semantic metadata providers are also in charge of registering that metadata in the semantic metadata repository, implemented in Jena, which aggregates it and acts as a centralised metadata repository of a set of WS-Resources, as shown in Figure 3. S-MDS clients use RDQL queries to access the semantic metadata. In [29] we can find a description of a prototype implementation that uses the GLUE ontology for describing computing resources and that is deployed in a cluster with a local scheduler. This description shows how all the previous components (manager, semantic metadata provider and

semantic metadata repository can be deployed in the Globus Container.

4.4 SEMANTIC DATA INTEGRATION IN THE CANCER BIOMEDICAL INFORMATICS GRID (caBIG)

The caBIG project of the U.S. National Cancer Institute (NCI) is a large scale initiative to deliver IT infrastructure to share distributed data and computing resources in cancer research. caBIG tackles the semantic data integration problem by the caGrid infrastructure [15] and the caCore toolkit [17]. Here we focus on the OGSA-DAI compliant caGrid Data Services [1], which provide an object-oriented view of data kept in native formats (mostly relational). Two metadata types are managed: for describing Data Access service interfaces and for mappings between a service object model and the native data backend that it refers to.

- The first group of metadata is managed by the cancer Data Standards Repository (caDSR). caDSR holds annotated OO model fragments (termed Common Data Elements -CDE) of the data providers, and is maintained by expert curators, who monitor and supervise model providers according to the caCore's metadata provisioning processes. Joining the caGrid as a provider requires devising an OO model of the data in UML, annotating this model with ontology terms and storing these annotations as CDEs in the metadata registry. Expert monitoring of the whole process prevents metadata duplication and increases model re-use. Besides, caCore provides support for metadata lifecycle management and change notification. Finally, the caDSR metadata services in caGrid are used by client applications, not by the data access services themselves.
- The second group of metadata is used by access services during their execution. CaGRid uses an ad-hoc representation for this metadata.

The object model of a caGrid OGSA-DAI service is annotated with ontologies, provisioned by an ontology service (the Enterprise Vocabulary Service). Ontologies are developed using description logic, but deployed only as controlled vocabularies. As with metadata, ontologies are used by client applications, and not by the caGrid Data Access services, to query and aggregate data from different data access services.

The system has been implemented by extending OGSA-DAI with a new query activity that accepts semantic search requests in the caGrid Query Language (CQL), as shown in Figure 4. CQL is an object oriented query language that allows expressing queries with objects, related objects and attributes with desired values. While the access interface to data is OO based, the data is kept in native formats (mostly relational data or flat files). For the case of relational data caGrid provides the tools

[1] OGSA-DAI [14] has been developed by the UK National eScience Centre and the Universities of Edinburgh and Manchester. It allows exposing data resources on to grids, including a collection of components for querying, transforming and delivering data, and a simple toolkit for developing client applications.

for making object relational mappings and respective auto-generation of data access software code.

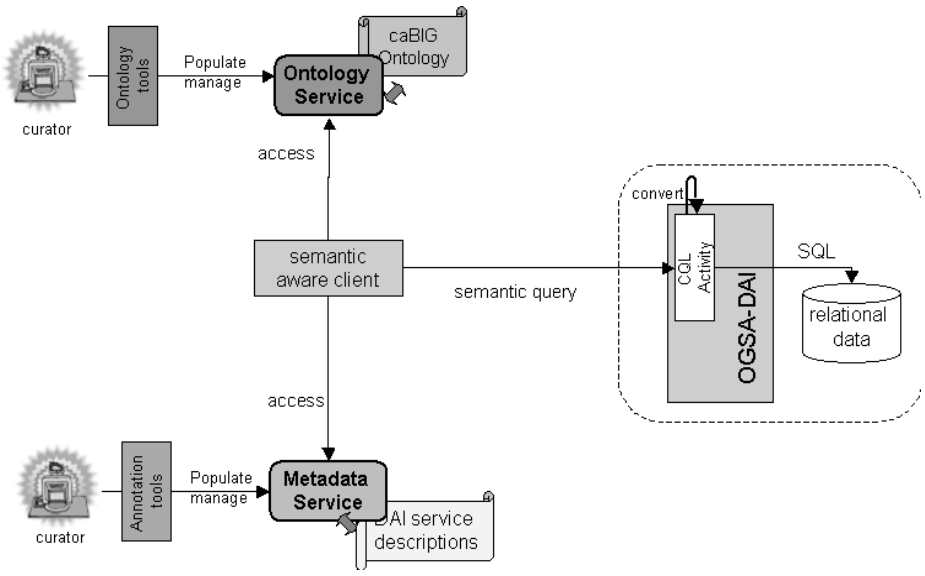

Fig. 4. caGRID Data Access Services Architecture.

4.5 SEMANTIC OGSA-DAI

Semantic-OGSA-DAI [33] is an extension of OGSA-DAI for ontology-driven access to relational data. It aims at supporting distributed query processing, semantic integration of distributed databases, and dynamic discovery of data sources depending on their contents and capabilities.

In S-OGSA-DAI, metadata describes the relationships between relational data sources and RDFS ontologies, using the D2R mapping language [26]. Metadata is stored within S-OGSA-DAI and can be retrieved if needed. S-OGSA-DAI does not impose a specific knowledge model to describe data sources, and it assumes that the data sources and the ontologies have been developed separately.

The extension of OGSA-DAI follows the extensibility option recommended for OGSA-DAI. It defines a new activity that is able to process RDQL queries, as shown in Figure 5. This approach is minimally intrusive, as the core of the OGSA-DAI code remains unchanged and it does not affect the service interface. Instead, the new functionality is developed as an add-on to the existing OGSA-DAI functionality. For the implementation of this new activity, RDQL queries are translated to SQL

with the D2RQ [26] query translation engine.

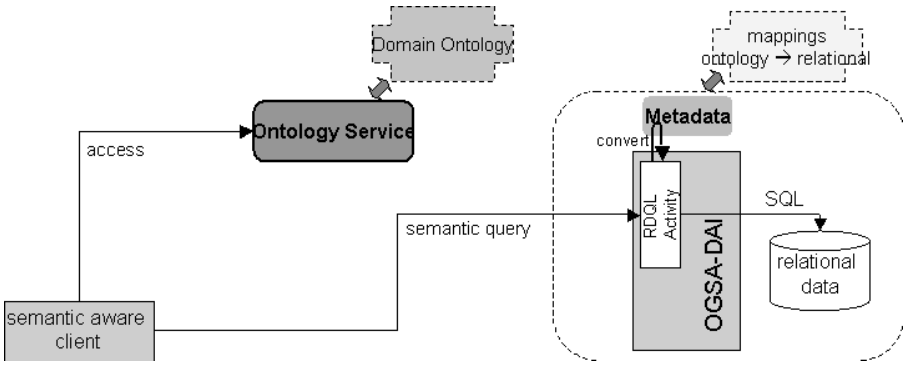

Fig. 5. Semantic OGSA-DAI Architecture.

4.6 GRID META-SCHEDULING SERVICE

Meta-schedulers [6] are services that interface with multiple local schedulers (which offer advance reservation of resources based on job execution start and stop times, as well as at least partial access to local schedules) or other meta-schedulers to negotiate with them advance reservation of resources based on user requirements, such as time or QoS constraints. The goal of this negotiation is to determine feasible time slots in which all required resources are available for the requested start time to execute a distributed workflow consisting of multiple jobs.

[7] proposes a semantic model to describe scheduler capabilities, where each local scheduler or meta-scheduler contributes annotations for its capabilities to a common metadata repository. Metadata is based on an OWL DL ontology, called the Grid Scheduling Ontology (GSO), where each class represents a specific configuration of scheduler capabilities. Schedulers are represented as individuals of one or more classes of this ontology, hence reducing the scheduler selection problem to querying the ontology model about all the known individuals of a class. The metadata is relatively stable, given the slow rate of change of a scheduler's capabilities.

This approach improves the accuracy of the scheduler pre-selection, and additional schedulers can be added to the pool at low cost. The new components, i.e. those that represent an evolution with respect to [6], are shown in Figure 6. This includes the use of ontology and reasoning services, a metadata repository and an annotation service (the LS Semantic Adapter).

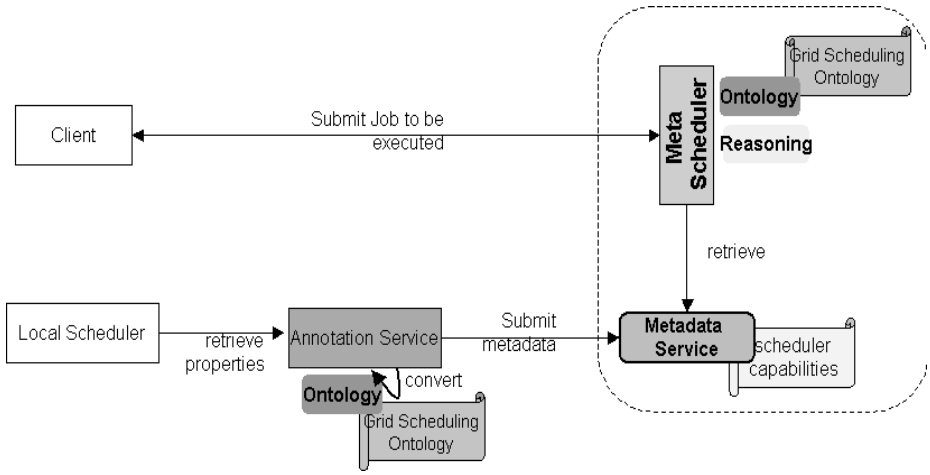

Fig. 6. Meta-scheduler architecture.

4.7 ONTOLOGY AND ROLE-BASED VO AUTHORIZATION

VO-AuthZ [18] is an ontology and role-based access-control service, which accepts and produces XACML-compliant authorization requests and responses for a resource access operation. Ontogrid-AuthZ calculates at run-time a subject's eligibility to access a resource, using a set of declaratively defined access control rules.

Metadata in VO-AuthZ describes domain attributes of the resource requestors and is represented in RDF. The system does not impose a particular provisioning process for metadata regarding subjects: this can be done manually or obtained from other information sources. Metadata is stored and managed by the S-OGSA Semantic Binding service [19].

Subject roles are defined with certain restrictions on subjects' properties and are deduced for a particular subject at run-time taking into account its properties at that time and using instance reasoning (aka ABox reasoning). Role definitions are based on the KAoS suite of ontologies [11], which contain descriptions of actors, groups, actions, resources, policy types, etc. These ontologies are specialized to describe domain specific access control policies and are implemented in OWL.

The system operation starts with the receipt of an authorization request, as shown in Figure 7. The request contains the subject's Distinguished Name and further properties (i.e. RDF based metadata), which had been obtained by the caller service by contacting known metadata servers (i.e. Semantic Binding services). This information together with the VO ontology is passed onto a description logic classifier (Pellet [25]), which performs instance level (A-box) reasoning to deduce the roles of the subject. These roles are then used to decide whether access is allowed

or not, using a lookup table.

VO-AuthZ has been re-factored from an existing XACML-based service operating over access control lists for subjects. The old decision logic has been replaced with the aforementioned one, while keeping the service interface intact. In the new XACML request, information about the subject is submitted as attributes, including additional RDF metadata about the subject, using XACML extensibility options. In the absence of additional metadata the system would not be able to deduce subject eligibility and would return an INDETERMINATE response.

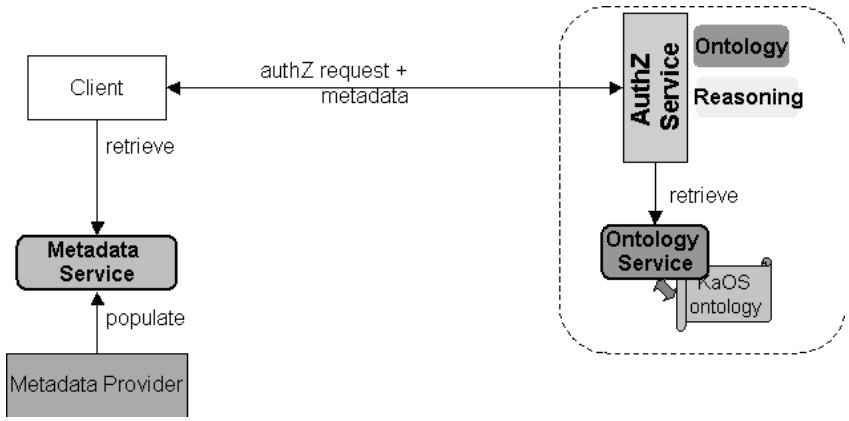

Fig. 7. Ontology and role-based VO Authorisation architecture.

5 GRID MIDDLEWARE SYSTEM ANALYSIS

Tables 2, 3 and 4 summarise the results of applying the analysis framework to the selected SAGMS. The information contained in the tables aims at giving some light about their most characteristic features, the role of semantics in them and the decisions taken in their development.

In our analysis, we identify three service groups: information services, data access services and decision-support services. These groups are tightly related to the type of OGSA service groups that the services belong to (information, data, and the rest of groups that require more decision-support functions). However, the borderlines between these groups are not rigid.

We make this classification to focus better on the features that are similar and different in each group, so as to obtain better guidelines and lessons learned for future developments. The distinguishing features of each service group are as follows:

	SYSTEM	Semantic SRB	GRIMOIRES	S-MDS	caBIG Data Access Services	Semantic OGSA-DAI	Grid Meta Scheduling	VO Authorization
Purpose	OGSA service Group	Information	Information	Information	Data	Data	Execution	Security
	Functionality	Search engine	Registry	Index	Data integration	Data integration	Scheduling	Role-based Access control
	Added value functionality	New capability	New capability	New capability	New capability	New capability	More accuracy	More flexibility
	Implementation Standards	-	UDDI	-	OGSA-DAI	OGSA-DAI	-	XACML
Development	Software Status	Proof-Of-Concept	Production	Proof-Of-Concept	Production	Alpha, Proof-of-Concept	Alpha, Proof-of-Concept	Alpha, Proof-of-Concept
	Approach to obtain Semantic Awareness	re-working	Development from scratch	re-working	re-working	re-working	re-working	re-working
	Re-working Method	Augment	re-write	Augment	Augment	Augment	Augment	Intrusive extension
	Extensions to Standards	None	UDDI	None	None	None	None	None
	Service Orientation	No	Yes	Yes	Yes	Yes	Yes	Yes

Table 2. Grid Middleware Features of Selected Semantic Aware Grid Systems

KNOWLEDGE		Semantic SRB	GRIMOIRES	S-MDS	caBIG Data Access Services	Semantic OGSA-DAI	Grid Meta Scheduling	VO Authorization
Entities	Required Knowledge Models	Agnostic	Agnostic	Agnostic (OWL-S by default)	Agnostic	Agnostic	Yes, Scheduling ontology	Yes, Role ontology
	Knowledge Model Lifecycle	Non-frequent Changes	Non-frequent changes	Non-frequent Changes	Frequent Curated Changes	Frequent Changes	Non-frequent changes	Frequent Changes
	Knowledge Model Coverage	Domain Model	Domain Model	Domain Model	Domain Model	Domain Model	Control, Scheduler pre-selection	Control, Access Control Rules
	Knowledge Model Generation	Agnostic	Agnostic	Agnostic	Agnostic	Agnostic	Agnostic	Agnostic
	Dependency level to Knowledge	Optional	Optional	Optional	Mission Critical	Mission Critical	Optional	Mission Critical
	Security	No security	No security	No security	Secured ontology	No security	No security	No security
Technologies	Knowledge Representation Languages	OWL	RDFS/OWL	OWL	UML, OWL	RDF(S)	OWL	OWL
	Knowledge Management Tools and Infrastructures	Jena, Pellet	Jena, Sesame	Jena	Apelon Ontology Server	Jena, Sesame, D2RQ	WS-DAOINT, Pellet	OWL-API, Pellet
	Knowledge Model Access	local	N/A	local	remote	local	local	remote

Table 3. Knowledge Features of Selected Semantic-Aware Grid Middleware Systems

	METADATA	**Semantic SRB**	**GRIMOIRES**	**S-MDS**	**caBIG Data Access Services**	**Semantic OGSA-DAI**	**Grid Meta Scheduling**	**VO Authorization**
Entities	Metadata based on a Knowledge Model	yes	no	yes	yes	yes	yes	yes
	Metadata Lifecycle	Frequent Changes	Frequent changes	Frequent Changes	Non-frequent changes	Non-frequent changes	Non-frequent changes	Frequent Changes
	Metadata Coverage	Domain Model	Domain Model	Domain Model	Domain Model and Transformation	Domain Model and Transformation	Domain Model	Domain Model
	Metadata Generation	Manual	Agnostic	Automatic	Semi-automatic / Curated	Manual	Manual for capabilities, automatic for state	Agnostic
	Dependency level to Metadata	Optional	Optional	Optional	Mission Critical	Mission Critical	Optional	Mission Critical
	Access Security	No security	Secured	No security	Secured	No security	Secured	No security
Technologies	Metadata Representation Languages	RDF	RDF and Attribute Value pairs	RDF	ISO/IEC 11179 Standard	RDF	RDF	RDF
	Metadata Management Tools and Infrastructures	Jena	Jena, Sesame	Jena	Oracle RDBMS	Jena, Sesame, D2RQ	WS-DAOINT, Jena	OntoGrid Semantic Binding Service
	Metadata Access	local	local	local	remote	local	local	remote

Table 4. Metadata Features of Selected Semantic-Aware Grid Middleware Services

- Semantic-aware Information Services use semantics to describe the metadata of resources available in a system. Metadata is attached to those resources, stored in repositories and harvested by the system clients mainly for resource discovery.
- Semantic-aware Data Access Services use metadata for the description of the data sources to be accessed. Metadata consists in mappings between the data source schema and a set of knowledge models, which have been normally developed separately. Mappings are executed when a system needs to access the data and transform it into the knowledge model structure (e.g., by creating instances of classes defined in them).
- Semantic-Aware Decision Support services use semantics to implement part or all of their business logic. The services provide added-value functionality (more accuracy, flexibility, etc.) than the ones provided without semantics. Normally these services use the reasoning mechanisms associated to the knowledge models used to represent metadata and the business logic.

In the following subsections we give more details about each group of services, describing commonalities and differences between the systems described and providing guidelines and lessons learned for similar future developments.

5.1 Semantic-Aware Grid Information Services

Three services belong to this profile: S-SRB, GRIMOIRES and S-MDS. They can be classified as OGSA information services, aimed at **improving resource discovery.**

Metadata is the major entity in this service group: services use ontology-based metadata to describe existing resources, consolidate it in a common repository and query it to discover resources. Hence **metadata is treated as a first class citizen**. These services also provide metadata management facilities, such as lifetime management, change notification, etc., as described for S-MDS.

These systems do not impose any specific annotation process (that is, the process of providing metadata for the resources that they refer to). They all assume that **metadata provision has been done either manually or by specialised systems** (e.g., those able to transform the metadata already exposed by resources, normally in the form of WS-Resource properties, into ontology-based metadata). In the case of S-SRB, users can also provide this metadata manually with the extended mySRB user interface. Only in S-MDS the metadata provisioning service is more coupled to the internal architecture.

Each service is focused on the annotation of different types of entities (files and databases in S-SRB, services in GRIMOIRES and any type of resources in S-MDS). To increase interoperation they recommend using a **minimal generic metadata schema**: UDDI and WSDL in GRIMOIRES, Dublin Core in S-SRB, and OWL-S in S-MDS. However, **the three systems are generic with respect to the domain knowledge model used**, in the sense that they could be easily applied to other types of Grid entities by just changing the ontologies used for their descriptions.

With respect to the access to ontology-based metadata, GRIMOIRES and S-SRB allow using non-semantic-aware clients (which will not benefit from the additional services provided by them) as well as semantic-aware clients. S-MDS forces clients to be semantic-aware in order to interact with the system. Semantic-aware clients query these services in different ways:

- With API calls or system-specific objects (e.g. a set of attribute-value pairs), which are matched against existing metadata. In this case the service is responsible for calculating matches programmatically.
- With a query language (e.g. RDQL, SPARQL) suited to the underlying representation formalism (RDF). In this case the query is built by the client and forwarded to (and executed by) the underlying metadata storage system.

Currently these systems only support **basic querying and some instance reasoning** (as in S-SRB). Hence **the awareness of specific domain knowledge and metadata lies within the client applications**, which interpret the results obtained from the services. We think that, in the future, semantic-aware Grid services in this category will provide more sophisticated matchmaking techniques (e.g. lexical similarity), support for inexact/partial matches [10], etc.

5.2 Semantic Aware Grid Data Access Services

S-OGSA-DAI and the caGRID Data Services belong to this profile. They implement services that can be classified inside the OGSA Data Access Service group.

Their common feature is that they all **allow accessing distributed heterogeneous data sources via the façade of a commonly agreed domain model**. Here metadata describes the schema-level mappings that can be established between data sources and domain models. These mappings are executed when the data has to be retrieved according to the domain model that the clients are able to deal with.

The services are **generic with respect to the domain knowledge models used**. These domain models can overlap completely with the data that is included in the data sources, can be narrower or wider.

Another common feature of all these services is that they are implemented as **extensions of OGSA-DAI**, which is one of the standards de facto for access to data sources in Grid systems. As a result, the extensions done in both S-OGSA-DAI and the caGrid Data Services are similar: they consist in the creation of a set of activities for dealing with the format used to specify mappings and for executing those mappings on the specified data sources. However, the interface that they provide to clients is different in each case: S-OGSA-DAI offers an RDQL-aware interface (clients have to specify their queries to the system in this language, which will be translated to SQL) and the caGrid Data Services offer a proprietary CQL-aware interface, an object-oriented query language that is suited to the specific type of applications for which this extension is devised.

Similarly to the previous service category, **a portion of semantic awareness also lies mostly within the client applications of these services**. These

applications retrieve and aggregate data views with respect to a common domain model, and they can access ontology services and use ontology alignment and mapping techniques to integrate data across multiple data grids, but this support is not provided by any of the revised systems.

A difference between both systems lies in how they use the domain model. S-OGSA-DAI uses the domain model as a schema that is populated by the data transformed from the original data sources. In caGrid, data is transformed into an object-oriented model and then it is strongly typed with elements from the domain model.

In the future we foresee that similar semantic-aware data access services will be developed, with a focus on new query languages and query result frameworks and with the inclusion of heterogeneous data integration processes inside services, as opposed to the current situation where clients perform their own integration. We also foresee that the extensibility options of OGSA-DAI will still be used, due to its robustness. Other foreseen extensions are related to the use of data access control mechanisms, so that the sharing and access to data is better controlled.

5.3 Semantic-Aware Decision Support Grid Services

The two last services belong to this profile: the ontology and role-based VO Authorisation system and the ontology-based Meta-Scheduler. The first one can be classified as an OGSA Security Service and the second one as an OGSA Execution Management Service, as shown in Table 2.

The common feature in this group of services is that they **outsource a portion (or all) of their business logic to an ontology and its associated reasoning mechanisms.** These services use application-domain ontologies (e.g., an application-specific extension of the KaOS ontology set in the VO AuthZ system and a scheduling ontology in the meta-scheduler) to encode their business rules (e.g. rules of "whom is authorized" in KaOS). At run time, they take decisions based on the combination of the metadata obtained from different resources and the use of the automated reasoning mechanisms of the formalism that the business logic is represented with.

This contrasts with the conventional approach to implement control logic, which mainly consists in hard-coding the system's business rules into software programs. The justification for using such type of approach in Grid systems is that business rules can be, in many cases, complex, dynamic, and could be provided by multiple parties. In such circumstances the conventional approach has the disadvantages of 1) rendering the business logic un-sharable, and 2) increasing code maintenance and debugging costs.

The knowledge models used in this category are represented with **richer knowledge representation formalisms** (e.g. restrictions expressed with axioms, rule language constructs, etc.), due to the fact that they aim at capturing business logic.

With respect to metadata, **annotation processes are normally application-dependent**. Besides, metadata can be deployed in different ways. It can reside

alongside knowledge, that is, the ontology driving the service business logic also contains metadata describing the current situation of the system, as in the Meta Scheduler. Or it can be hosted by metadata services and pushed-to or pulled-by the service for its decision-making process, as in the VO AuthZ system.

Finally, **the interfaces of these systems to their conventional clients are not changed**. Systems only provide an alternative way to perform an existing function inside the business logic of the system, with better quality of service properties (more accuracy, flexibility, etc.).

We foresee that in the future other similar systems will be developed. Due to the complex nature of the applications that can be developed using this approach, it is not easy to characterise how they will be implemented. However, we foresee that the main aspect of outsourcing the business logic of the system to external ontology and reasoning services will be maintained.

6 SUMMARY

SAGMS are only recently making their way from research labs to real-life Grid deployments. Hence this is a good time to start analysing the decisions taken in their development and to compile lessons learned, so that this can be taken into account in the development of the next generation of systems. At the time of writing this review we have not found another work that analyzes the role that knowledge and metadata plays in Grid middleware services.

In this paper we aimed at reviewing the main features of the early Grid middleware systems that use semantics in e-Science applications. We focused on: 1) how they have been developed and how they are used, 2) what their architectural features are and 3) what is the role of knowledge and metadata in their architecture.

To improve the understanding of these services, we proposed an analysis framework organised according to three dimensions: service development, metadata and knowledge. For each dimension we focused on a general description (WHAT) and on how they are implemented (HOW). For each of them we identified several specific features and we formulated questions to be answered or topics to be discussed.

We applied the framework to analyse in detail each of the seven Grid middleware services selected. We classified these services into three categories: information services, data access services and decision-support services. This classification allowed us to focus on the characteristics of each system in the context of the problems that they aim at solving, and to provide more insight about which are the typical techniques and design proposals that are applied for each group. For example, we have seen that those services devoted to data access use the extensibility options of OGSA-DAI, that those devoted to information provisioning are focused on the use of metadata to describe active resources and the consolidation of metadata in common registries, and that those that provide decision-support reason with the represented knowledge and metadata to take decisions.

7 ACKNOWLEDGEMENTS

This work is supported by the EU FP6 OntoGrid project (STREP 511513) funded under the Grid-based Systems for solving complex problems, and by the Marie Curie fellowship RSSGRID (FP6-2002-Mobility-5-006668).

REFERENCES

[1] Globus Toolkit: `http://www.globus.org`

[2] EGEE gLite: `http://glite.web.cern.ch/glite/`

[3] UNICORE: `http://www.unicore.eu/`

[4] Open Middleware Infrastructure Institute: `http://www.omii.ac.uk/`

[5] S. Miles, J. Papay, T. Payne, K. Decker, L. Moreau. Towards a Protocol for the Attachment of Semantic Descriptions to Grid Services. In The Second European across Grids Conference, Nicosia, Cyprus, pages 10, January 2004.

[6] O. Waldrich,et. al. A Meta-scheduling Service for Co-allocating Arbitrary Types of Resources. In Proc. of the Second Grid Resource Management Workshop (GRMWS-05) in conjunction with the Sixth International Conference on Parallel Processing and Applied Mathematics

[7] P. Missier, P. Wieder, and W. Ziegler. Knowledge and Data Management in Grids, volume 3 of CoreGRID , chapter Semantic Support For Meta-Scheduling in Grids. Springer, 2006. In press.

[8] J. J. Carroll, et. al. Jena: Implementing the Semantic Web Recommendations. Technical report, HP Labs, December 2003.

[9] T. Kawamura, et.al. Public Deployment of Semantic Service Matchmaker with UDDI Business Registry. In S. A. McIlraith, D.Plexousakis, and F. van Harmelen, editors, Proc. of the 2004 International Semantic Web Conference (ISWC 2004), number 3298 in Lecture Notes in Computer Science, pages 471-485. Springer, 2004.

[10] M. Paolucci et. al. Semantic Matching of Web Services Capabilities. In I.Horrocks and J. A. Hendler, editors, First International Semantic Web Conference, volume 2342 of Lecture Notes in Computer Science. Springer, JUNE 2002.

[11] J. Bradshaw et.al., Representation and reasoning for DAML-based policy and domain services in KAoS and nomads, in Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems. Melbourne, Australia: ACM Press, 2003.

[12] I. Foster, H. Kishimoto, A. Savva. The Open Grid Services Architecture Version 1.0 Specification. Global Grid Forum OGSA Working Group, 2005. `https://forge.gridforum.org/projects/ogsa-wg/document/draft-ggf-ogsa-spec/en/23`

[13] K. Czajkowski et. al , "Web Services Resource Framework (WSRF)." Globus Alliance and IBM, 2005.

[14] M. Antonioletti, M.P. Atkinson, R. Baxter, A. Borley, N.P. Chue Hong, B. Collins, N. Hardman, A. Hume, A. Knox, M. Jackson, A. Krause, S. Laws, J. Magowan, N.W. Paton, D. Pearson, T. Sugden, P. Watson, and

M. Westhead. The Design and Implementation of Grid Database Services in OGSA-DAI. Concurrency and Computation: Practice and Experience, Volume 17, Issue 2-4, Pages 357-376, February 2005.

[15] J. Saltz, et. al. caGrid: Design and Implementation of the Core Architecture of the Cancer Biomedical Informatics Grid, Bioinformatics Journal. Vol. 22 no. 15 2006, pages 1910-1916

[16] V. Astakhov, A. Gupta, J. Grethe, E. Ross, D. Little, A. Yilmaz, M. Martone, X. Qian, S. Santini, M. Ellisman (in press) Semantically Based Data Integration Environment for Biomedical Research. Proceedings of the 19th IEEE International Symposium on Computer-Based Medical Systems.

[17] P.A. Covitz, F. Hartel, C. Schaefer, S. Coronado, G. Fragoso, H. Sahni, S. Gustafson and K.H. Buetow. (2003) caCORE: a common infrastructure for cancer informatics. Bioinformatics Journal, 19, 2404-2412.

[18] P. Alper, et. al. An Authorization Scenario for S-OGSA. Demo presentation at the European Semantic Web conference 2006.

[19] O. Corcho, P. Alper, I. Kotsiopoulos, P. Missier, S. Bechhofer, C. Goble. An overview of S-OGSA: a Reference Semantic Grid Architecture. Journal of Web Semantics 4(2):102-115. June 2006

[20] C. Goble, O. Corcho, P. Alper, D. De Roure. e-Science and the Semantic Web: A Symbiotic Relationship. In proc. Discovery Science Conference 2006.

[21] I. Horrocks, P.F. Patel-Schneider. FaCT and DLP, in proceedings of Automated Reasoning with Analytic Tableaux and Related Methods: International Conference Tableaux-98, in Lecture Notes in Artificial Intelligence pages 27-30,Springer-Verlag, May 1998

[22] E Friedman-Hill. Jess Rule Engine for the Java Platform. Distributed Computing Systems,2003

[23] Jastor Java code generator. http://jastor.sourceforge.net/

[24] J.G. Frey. The CombeChem Experience. (2005). At, Knowledge Managment Workshop, e-Science All Hands Meeting 2005, Notttingham, UK, 19 Sept 2005. UK, EPSRC. http://eprints.soton.ac.uk/17452/

[25] BC Grau, B Parsia, E Sirin. Pellet, An OWL DL Reasoner. Proceedings of the Third International Semantic Web 2004

[26] Christian Bizer, Andy Seaborne. D2RQ - Treating Non-RDF Databases as Virtual RDF Graphs. Poster. 3rd International Semantic Web Conference (ISWC2004).

[27] Ian Horrocks et.al. SWRL: A Semantic Web Rule Language Combining OWL and RuleML W3C Member Submission 21 May 2004. http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/

[28] T. Bellwood. UDDI Version Specification, OASIS, July 2002.http://uddi.org/pubs/uddi_v3.htm

[29] Mirza Pahlevi S, Kojima I. S-MDS: A Semantic Information Service for Advanced Resource Discovery and Monitoring in WS-Resource Framework. GGF16 Semantic Grid Workshop, Athens, February 2006.

[30] M. Dean, G. Schreiber. OWL Web Ontology Language Reference. W3C Recommendation, February 2004.

[31] D. Brickley, R. Guha, and B. McBride. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, February 2004.

[32] G. Klyne, J.J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, February 2004.

[33] I. Kotsiopoulos. Semantic OGSA-DAI. http://www.ontogrid.eu/s-ogsa-dai

[34] M.H. Eres et. al. Implementation and utilisation of a Grid-enabled problem solving environment in Matlab. Future Generation Computer Systems (In press).

[35] C.A. Goble et. al. Knowledge Integration: In silico Experiments in Bioinformatics in The Grid: Blueprint for a New Computing Infrastructure Second Edition eds. Ian Foster and Carl Kesselman, 2003, Morgan Kaufman, November 2003.

[36] M. Snchez-Gestido et.al., Complex Data-intensive Systems and Semantic Grid: Applications in Satellite Missions in proc. e-Science 2006 conference.

[37] T.S. Myers, I.M. Atkinson, W.J. Lavery. The Semantic Reef: Managing complex knowledge to predict coral bleaching on the Great Barrier Reef. To appear in: Proceedings of the AusGrid 2007 conference.

[38] M. Bachler, S. Buckingham Shum, Y. Chen-Burger, J. Dalton, D. De Roure, M. Eisenstadt, J. Komzak, D. Michaelides, K. Page, S. Potter, N. Shadbolt, A. Tate. Collaboration in the Semantic Grid: a Basis for e-Learning. (2004). In Proceedings Grid Learning Services workshop (7th International Conference on Intelligent Tutoring Systems), pp. 1-12, Maceio, Brazil.

[39] Aldouri, R., G.R. Keller, A. Gates, J. Rasillo, L. Salayandia, V. Kreinovich, J. Seeley, P. Taylor, S. Holloway. GEON: Geophysical data add the 3rd dimension in geospatial studies. Proceedings of the ESRI International User Conference 2004, Paper 1898; San Diego, California, August 9-13, 2004

[40] Science 2020. Microsoft Research. http://research.microsoft.com/towards2020science/

[41] D. De Roure, N. Jennings, and N. Shadbolt. Research Agenda for the Semantic Grid: A Future e-Science Infrastructure. Technical report UKeS-2002-02, UK e-Science Technical Report Series, National e-Science Centre, Edinburgh, UK. December 2001.http://www.semanticgrid.org/html/semgrid.html

[42] Z. Turk, V. Stankovski, A. Gehre, P. Katranuschkov, and K. Kurowski. InteliGrid deliverable D13.1 - Semantic grid architecture. 2005.

[43] D. Snelling, M. Fisher, A. Basermann. NextGrid Architecture White Paper. December 2005. http://www.nextgrid.org/white_paper.htm

[44] S.J. Jeffrey, J. Hunter. A Semantic Search Engine for the Storage Resource Broker. GGF16 Semantic Grid Workshop. Athens, Greece. February 2006.

[45] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, F. Yergeau (eds). Extensible Markup Language (XML) 1.0 (Fourth Edition). W3C Recommendation. August 2006. http://www.w3.org/TR/XML/

[46] J. Broekstra, A. Kampman, F. van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. International Semantic Web Conference 2002, Sardinia, Italy

[47] RDQL: http://jena.sourceforge.net/RDQL/

Pinar ALPER (MPhil, MSc) is a Research Associate working within the Information Management Group (IMG) at the School of Computer Science of the University of Manchester. She currently participates in the EU FP6 IST project OntoGrid (FP6-511513), and has participated in UK e-Science pilot project ^{my}Grid. Her research work focuses on application of semantics in the areas of Web Services and the service oriented Grid.

Dr. Oscar CORCHO is working as a Marie Curie fellow at the Information Management Group of the University of Manchester. His research activities include the Semantic Grid, the Semantic Web and Ontological Engineering. Currently he participates at the EU FP6 IST project OntoGrid (FP6-511513). He has published the books "Ontological Engineering" and "A layered declarative approach to ontology translation with knowledge preservation", and over 30 journal and conference and workshop papers.

Carole GOBLE is a Professor of Computer Science at the University of Manchester and co-leads the IMG group since 1997. Her research interests are on the accessibility of information, particularly the use of terminological and ontological services for the representation and classification of metadata in a range of application domains. Her recent work has been focused on two major areas: the Semantic Web and e-Science/Grids, and has been instrumental in an effort to link the two areas by the application of Semantic Web technologies to the Grid and e-Science, a fusion dubbed the Semantic Grid. She is co-chair of the Open Grid Forum Semantic Grid Research Group.