Semantics-Based Grid Resource Management*

Alexandre C.T. Vidal
Francisco José da Silva e Silva
Universidade Federal do Maranhão, Brazil
{vidal, fssilva}@deinf.ufma.br

Sergio Takeo Kofuji
Fabio Kon
University of São Paulo, Brazil
kofuji@lsi.usp.br, kon@ime.usp.br

ABSTRACT

Scheduling parallel and distributed applications efficiently onto grid environments is a difficult task and a great variety of scheduling heuristics have been developed aiming to address this issue. A successful grid resource allocation depends, among other things, on the quality of the available information about software artifacts and grid resources. In this paper, we propose a semantic approach to integrate selection of equivalent resources and selection of equivalent software artifacts in order to improve the schedule of resources suitable for a given set of application execution requirements. We also describe a prototype implementation of our approach based on the Integrade grid middleware and experimental results that indicate its benefits.

Categories and Subject Descriptors

H.3.4 [**Distributed Systems**]: [Systems and Software, Information Storage and Retrieval]

1. INTRODUCTION

Computational grids are dynamic and heterogeneous environments that should deals efficiently with the coordinated sharing of resources for the solution of computational problems. A central issue in grid environments is the appropriate match of grid resources with application requirements. To our best known, most grid resource management systems focus their approaches to match grid resources and application requirements based only on resource descriptions. Usually a search engine gets a resource application request and tries to find a suitable resource or a set of suitable resources over which an application may runs on. In general, resource discovery mechanisms, perform an exact matching of application requirements with attribute-based resources descriptions. More flexible mechanisms, as shown in [11], allow asymmetrical matching of inexact or incomplete application requirements with ontology-based resource descriptions.

A successful grid resource allocation depends, among other things, on the quality of the available information about software artifacts and grid resources. Strategies for application scheduling and scheduling algorithms are based on descriptions of resource attributes and applications requirements and preferences. So, the way how this information is captured, stored, organized, and made available plays a relevant role in resource matching activities. Every one of these tasks is built around the available grid resources and applications metadata. Grid metadata permeates and connects them. From our point of view, the grid metadata design can affect the execution of tasks in different and relevant ways. In order to face this problem, we propose a semantic approach, based on ontologies, intendend for efficient application execution on the grid.

Ontologies can improve the quality of information about grid software and resource. They enable reuse of domain knowledge, the sharing of common understanding about domain concepts, explicit definition of domain assumptions, and interoperability enhancement among other important features, in different grid domains. A worthwhile grid ontology should comprise a taxonomy of grid concepts, properties, and axioms which can be reused in different contexts and by other ontologies. Such ontology, used in conjunction with inference tools, provides a flexible and powerful way of reasoning about grid elements.

In this paper, we present a set of related extensible grid ontologies and describe how this semantic approach may be used on grid environments, highlighting resource matching for scheduling application execution requests. Our prototype implementation uses the InteGrade grid middleware [6]. Integrade follows an opportunist approach, taking advantage of idle computational resources for executing extensive parallel applications. We describe how our semantic approach could be integrated to Integrade in order to allow better management and effective reuse of software and grid resources. We also performed experiments, comparing the results of the conventional resource matching approach with the semantic one. The results sustain our claim that semantics improve the reuse, sharing, and integration of software and computational resources on grid environments.

This paper is organized as follows: Section 2 introduces semantic grid concepts and presents a set of grid ontologies and their relation mechanisms on the knowledge base. Sec-

*This work is part of the Integrade2 project supported by the Brazilian Federal Research Agency, CNPq, grant $N^o55.0094/05-9$, FAPESP, Brazil, process $N^o2004/08928-3$, and from CAPES PQI, Brazil, process $N^oADM0049/03-4$.

MGC '07, November 26, 2007 Newport Beach, CA, USA

tion 3 explores the use of semantics on grid environments considering two related scenarios. Section 4 describes our prototype based on the Integrade middleware. Section 5 describes the experiments performed and compares the semantic approach for resource matching with the Integrade conventional one. Section 6 describes some relevant related works, comparing them with our approach, while Section 7 presents our conclusions and describes future directions of this work.

2. DEFINING THE GRID ONTOLOGIES

De Roure et al. [5] define semantic grid as "an extension of the current grid in which information and services are given a well-defined meaning, better enabling computers and people to work in cooperation". Semantic grid, as semantic web technologies, enable integration of applications, data, resource, and users in specific domains on the grid and eases automation of middleware tasks, such as selecting grid resources for application execution based on their requirements [10].

Our grid ontologies are described with OWL [4], a W3C recommended ontology language. OWL embodies the so called *Classical Paradigm*: a modelling paradigm based on characteristics appropriate to the open environment of the semantic web [9]. In this paper, we explore cases of inferencing and querying a grid ontology based on OWL-DL, a sublanguage of OWL.

Ontologies in OWL-DL can be processed by a *reasoner*, through inference, to meet different goals. Conclusions not explicitly presented in the knowledge base can be inferred from rules and axioms in this base, finding what is necessarily true and consistent with the system axioms. Subsumption inference task produces a class hierarchy based on the classes description. It enables one to construct an asserted hierarchy and let a reasoner engine infer and maintain any additional inheritance relation. Such task is related to the domain definition knowledge and, in general, it is useful to perform most inferences in advance, since inferences will apply to a domain as a whole. The reasoner can also perform consistency checking, and computation of inferred type.

We used the Protégé-OWL tool [7] to enter the proposed set of grid ontologies and to work on refining it. This tool enables the creation of an extensible OWL ontology, including classes, properties, restrictions on classes, and individuals (instances). Additionally, reasoning to check consistency and to infer class hierarchy was performed through a connected Pellet [12] reasoning tool.

We defined our system architecture based on the grid system ontology described in [15], which includes a large and unordered variety of different, although related, concepts, properties, and axioms. That ontology was an exploratory approach to expose the taxonomic and reasoning possibilities on the grid domain. In order to improve the knowledge base maintenance and scalability, we separated the original ontology in a set of related ontologies of three types: upper-level ontologies, ancillary ontologies, and *concrete* ontologies.

An *upper-level ontology* or *top-level ontology*, as defined in [3], is an "ontology that describes knowledge at higher levels of generality". In general, upper-level ontologies are used to describe basic concepts that can be extended in more specific ontologies. Ancillary ontologies contains domain specific concepts. We consider as ancillary an ontology that support another one, but performing a less important role. Concrete grid ontologies extends the upper-level ontologies to describe specific grid concepts, reusing the top ontologies. These different ontologies can be connected through an *import* mechanism.

The *import* mechanism allows an ontology to import other ontology. This mechanism enables to refer and extend all concepts and properties of the imported ontology, enhancing knowledge reuse and sharing. Concepts and properties from the imported ontology receive a prefix referring to the related name space. Other known mechanisms to relate ontologies, not explored in this work, are the so called *mapping* and *e-connection* mechanisms. These approaches are relevant to improve the knowledge base scalability.

In our knowledge base prototype, we have one upper-level ontology, which is called the *Grid Base Ontology*, one ancillary ontology, the *Platform Ontology*, and a concrete grid ontology, the *Grid Resource Management Ontology*.

Figure 1 shows the asserted class hierarchy of a specific *Platform Ontology* branch, the operating system branch. The *Platform Ontology* contains a short class hierarchy and related properties about the computational platform domain. It includes, among others, concepts such as operating systems, architectures and processors. This hierarchy can be extended with new concrete concepts, allowing to obtain new knowledge from incomplete asserted information and based on axioms defined in the ontology. Figure 2 presents the asserted class hierarchy of another specific *Platform Ontology* branch, the processor branch.

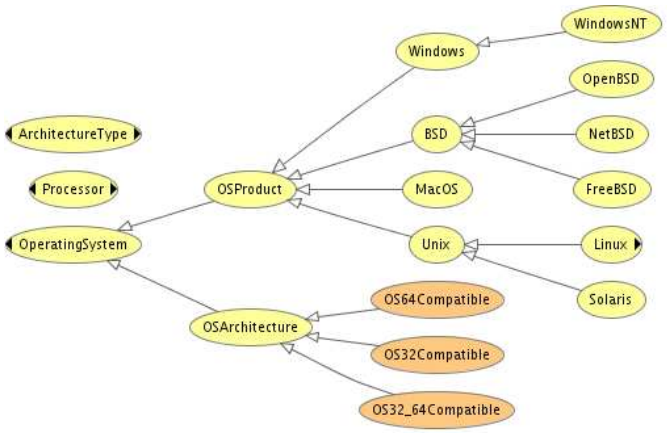

Figure 1: Platform Ontology - OS branch

The *Grid Base Ontology* initially acts as a fundamental taxonomy encompassing the main concepts related to grid systems. It contains concepts, properties and axioms which can be considered common to grid domain application and users. Figure 3, generated by the Protégé-OWL tool, shows the asserted class hierarchy of the *Grid Base Ontology*. The class hierarchy is based on two root concepts: *Grid Software Concepts* and *Grid Resource Concepts*. The former encompasses other related concepts, such as *Domain*, *Problem*, *Approach*, *Algorithm*, *Application*, *Software Artifacts*, and so on. These concepts are sufficiently high level to be extended by *Grid Application Developers* to describe more specifically their domains, problems, and related software solutions. *Grid Resource Concepts* encompasses concepts related to grid computational resources, such as *Computer*,

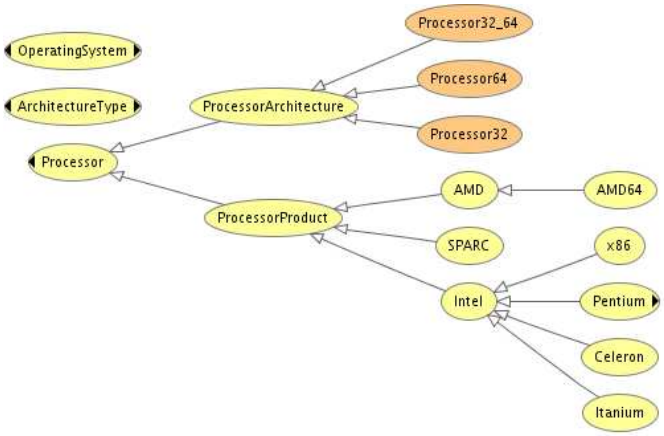

Figure 2: Platform Ontology - Processors

Cluster, *DiskSpace*, *MemorySpace*, and *Platform*. These concepts, in turn, can be extended by the *Grid Manager* to describe a concrete computational grid infrastructure. The upper-level ontology imports an ancillary ontology, the *Platform Ontology*, and reuse its concepts and properties.

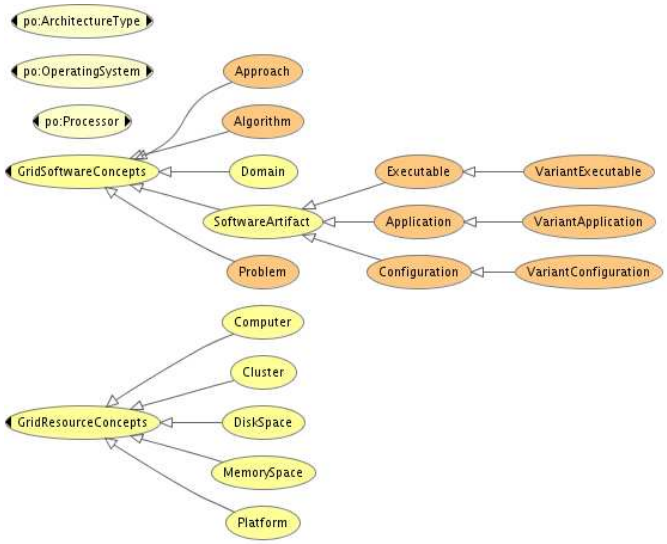

Figure 3: Grid Base Ontology

An example of a more specific grid ontology is the *Grid Resource Management Ontology*. In order to improve visibility, we separated parts of this ontology in two distinct figures. A set of more general software related concepts is presented in Figure 4, while some concepts related to software artifacts are shown in Figure 5. This ontology imports the *Grid Base Ontology* showed in Figure 3. The set of described concepts includes a specific domain, *Data Mining Domain*, and some of its specific related concepts, also derived from the *Grid Base Ontology*, such as *Data Mining Problems*, *Classification Approaches*, and so on. Other different domains and their related concepts can be created as extensions of the *Grid Base Ontology*, such as *Mathematic* and *Biology*. The complete set of ontologies is available for download at `http://www.deinf.ufma.br/~vidal/GBOS/`.

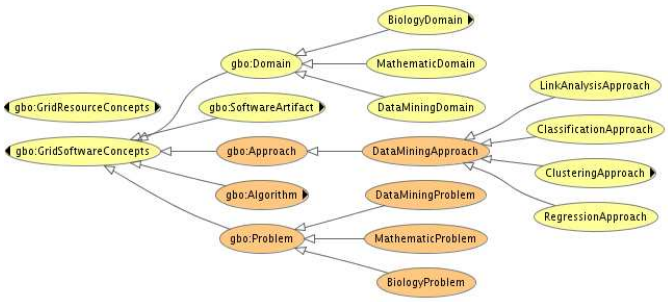

Figure 4: GRM Ontology - software concepts

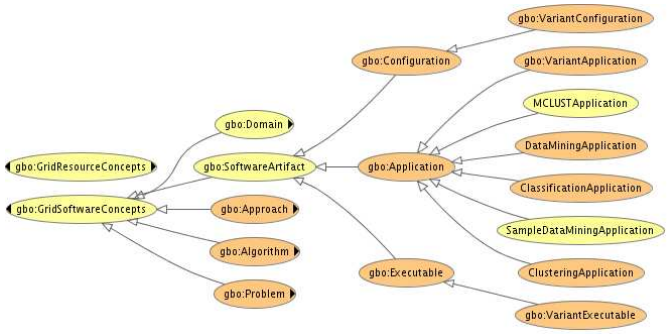

Figure 5: GRM Ontology - software artifacts

3. EXPLORING THE ONTOLOGIES

This section describes two scenarios of exploring the ontologies in order to improve resource management on grid environments. Initially, we discuss how grid scheduling can take advantage of a semantic resource pre-selection. Afterwards, we describe how the proposed ontologies can leverage the grid usage as a distributed and integrated application repository.

For both the scenarios, we adopted an *inference in advance* policy, for the whole set of involved ontologies. A subsumption inference updates the knowledge base in a suitable frequency. The inferred knowledge base contains new inheritance relationships, extending the available metadata with new semantic information.

3.1 Semantic Grid Scheduling

According to [17], a schedule is a mapping of tasks to time and computational resources and the *scheduling problem* consists in computing a schedule that optimizes some metric. The grid resources and applications metadata play a crucial role in such scheduling task, since they describe resource features and application requirements in the grid context.

Grids are heterogeneous environments. In the same way, grid metadata themselves are also heterogeneous. Such heterogeneity makes it difficult to explore grid resources efficiently. For example, different virtual organizations may employ distinct terminologies to describe a same grid resource. As a result, the integration of related grid resources is not an easy task. A semantic approach can mitigate this problem enabling the identification of related resources based on ontologies concepts and axioms.

In general, whenever a grid user requests an application

execution, the grid middleware schedules a set of resources for performing the computation. A semantic approach can also achieve better resource matching than an approach based on exact matching of metadata attributes, leading to more alternative nodes for scheduling the application. This is due to the fact that, through an inference policy, which extends the set of selected resources, the grid middleware can discover interchangeable or equivalent resources, considering the user restrictions for a given application submission (such as the required minimum memory and processing power).

3.2 Semantic Application Repository

Computer grids have been used to solve problems in varied areas of scientific, enterprise, and industrial activities, such as: computational biology, image processing for medical diagnosis, weather forecast, high energy physics, marketing simulations, and oil prospection. Grid computing empowers the conception of a new generation of applications that allow combining computations, experiments, observations, and data got in real time. The phenomena modeled by these applications require diverse software components whose compositions and interactions are extremely dynamic. Under particular conditions, application executables in a domain can be interchangeable and software components can be reused in different domains. We propose to leverage the grid usage as a distributed and integrated repository of software artifacts through the use of metadata.

To be able to select the best application component for each situation, it is important to have the means to classify these software artifacts and be capable of explicitly reasoning about them. The search mechanism for a software artifact should also be flexible, as far as possible, allowing to obtain equivalent alternatives for a proposed query.

Using our proposed ontologies, each virtual organization could feed the knowledge base during the application registering process, either extending the hierarchy of software concepts or creating individuals ones to represent the software component. Later on, the knowledge base could be enhanced through a subsumption inference. Some kind of bulk policies may be adopted, reducing the reasoner cost. Our semantic data structure privileges the class hierarchy and the subsumption inference, since both abstractions are central in ontology based approaches.

4. INTEGRADE BASED PROTOTYPE

The Integrade project [1] [6] is a multi-university effort to build a novel grid computing middleware infrastructure to leverage the idle computing power of personal workstations for the execution of computationally-intensive parallel applications.

The basic architectural unit of an Integrade grid is the cluster, a collection of machines usually connected by a local network. Clusters can be organized in a hierarchy, allowing to encompass a large number of machines. They can also be organized in a P2P fashion. Each cluster contains a *Cluster Manager* node that executes Integrade components responsible for managing the cluster computing resources and for inter-cluster communication. Other cluster nodes are called *Workstations*, which export part of its resources to Grid users. They can be shared or dedicated machines.

[1]Homepage: `www.integrade.org.br`

Integrade currently allows the execution of three application classes: (a) regular applications, where the executable code is assigned to a single grid node; (b) parametric or BoT applications, where several copies of the executable code are assigned to different grid nodes and each of them processes a subset of the input data independently and without exchanging data; (c) parallel applications following the BSP or MPI model whose processes occasionally exchange data between themselves.

We developed on Integrade a prototype of a *Semantic Grid Integration Architecture*, shown in Figure 6. Its main component is the SGIC Semantic Grid Integration Component, which is a facade module composed of two other functional specific components:

1. Semantic Metadata Manager (SMM): responsible for managing the semantic metadata. It allows to load and save the semantic metadata, to perform class hierarchy inferences, to check ontologies consistency, and to perform computation of inferred types. The current implementation of this component uses the Protégé-OWL API and the Pellet inference tool;

2. Semantic Grid Resource Manager (SGRM): responsible for managing grid resources, performing functions such as resource and application selection for resolving a given user query or execution submission. The current component implementation uses a query interface based on the Protégé-OWL API and the SPARQL query language.

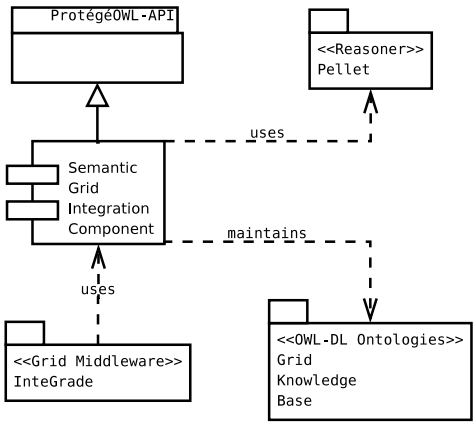

Figure 6: Semantic Grid Integration Architecture

While the described prototype was based on the Integrade grid middleware, our Semantic Grid Integration Architecture is sufficiently general in order to be ported to other grid software infrastructures with minimum effort.

5. MODEL EVALUATION

In order to evaluate our approach and demonstrate its usefulness, we performed experiments in a scenario of resource scheduling for application submissions, comparing the result of the conventional InteGrade resource selection with the one obtained with our semantic enhancements.

The grid environment hardware considered on the experiments is described on Table 1. Platforms P1, P2, P3, P6, and P7 execute the **i686 GNU/Linux** operating system,

platforms P4 and P5 the **amd_64 GNU/Linux**, and platform P8 the **PowerPC_Darwin** operating system. From the application execution perspective, platforms P1, P2, P3, P6, and P7 are equivalents and they correspond to 19 grid nodes. Platforms P4 and P5 are also equivalent and the grid environment has one node of each type, comprising two grid nodes. Only one grid node follows platform P8, which has none equivalent platform.

Types of Hardware Platforms	
Type	Platforms
P1	**Intel(R) Pentium(R) 4 CPU 3.00GHz**
P2	**AMD Athlon(tm) XP 2800+**
P3	**Intel(R) Xeon(TM) CPU 3.00GHz**
P4	**AMD Athlon(tm) 64 Processor 3200+**
P5	**AMD Athlon(tm) 64 Processor 3000+**
P6	**Intel(R) Pentium(R) 4 CPU 2.80GHz**
P7	**Intel(R) Pentium(R) D CPU 2.80GHz**
P8	**Power PC G4**

Table 1: Hardware Platform types

For performing the experiments, we used several software artifacts consisting of a set of algorithms for *Image Texture Analysis* (ITA), which are related to the *Pattern Recognition* domain. The algorithms describe conventional texture analysis methods, e.g. the Gray-Level Run-Length Method (GLRLM) and the Spatial Gray Level Dependency Method (SGLDM), used to extract characteristics from a given image [8]. The algorithms can be used in different horizontal domains, such as aerial photo image analysis for geographic researches and tomography image analysis for cancer diagnosis. The methods used by the different algorithms lead to different CPU and memory requirements but, under a specific context, these algorithms could replace each others. Table 2 shows the used set of algorithms and applications and the suitable platforms for executing each one of them, considering the available executable binaries.

Suitable Platforms and Nodes		
Application	**Platforms**	**#nodes**
SGLDM_App	**P1**	**8**
GLRLM_App	**P5 e P8**	**2**
GLDM_App	**P4**	**1**
PSM_App	**P4**	**1**

Table 2: ITA available software artifacts

On InteGrade, during the submission process the user can either select a specific application executable binary or an application, in a more general sense, to be executed by the grid. In the first case, the middleware will try to identify the nodes on the local cluster whose platforms are described exactly as the platform description of the correspondent binary. The second case is more flexible, in the sense that the middleware will try to first discover all available executables of the selected application. Such approach increases the set of suitable platforms to execute the application and, potentially, will increase the number of available machines for application execution.

The experiment consists of submitting a user request for each ITA application (SGLDM, GLRM, GLDM, and PSM) for the conventional Integrade middleware and for our extended version of it (with semantics), comparing the resource matching results obtained with both approaches. For the conventional Integrade, we used the more general case of submitting an application, since it increases the set of suitable platforms to execute the application, as explained on the previous paragraph. As shown on Table 2, there are binaries for executing the SGLD application on platform P1, the GLRM application on platforms P5 and P8, and both GLDM and PSM applications on platform P4.

Table 3 shows the results obtained. The conventional Integrade discovered 8, 2, 1, and 1 nodes available for executing applications SGLDM, GLRM, GLDM, and PSM, respectively, considering the grid hardware described on Table 1. The second column of Table 3 (1st Case) shows the results obtained with our semantic approach using the equivalence relationships between platforms asserted with the *Platform Ontology*. The third column of Table 3 (2nd Case) shows the results obtained when our semantic approach also considers the use of equivalent applications related to a set of equivalent algorithms, defined on the *Grid Management Ontology*. The better results are due to the fact that algorithms described through the assertion *gbo:describes some TextureAnalysisApproach*, are inferred to be also subclass of the class *TextureAnalysisApproach*. This happens because the class *TextureAnalysisApproach* is subclass of the class *Algorithm* and is also *defined* by the assertion *gbo:describes only TextureAnalysisApproach*. The acronym *gbo* refers to the name space of the imported ontology *Grid Base Ontology*.

Comparison of Scheduling Approaches			
Application	**#Selected Nodes by approach**		
	InteGrade	**Semantic**	
		1st Case	**2nd Case**
SGLDM_App	**8**	**19**	**22**
GLRLM_App	**2**	**3**	**22**
GLDM_App	**1**	**2**	**22**
PSM_App	**1**	**2**	**22**

Table 3: Comparison of scheduling approach results

As the experimental results demonstrated, the use of semantics can increase the amount of possibilities for execution a given user submission, since each application can be related to different equivalent executables. In the same way, each executable can be related to different equivalent platforms, leading to better resource matching results. This is just one advantage of exploring the use of semantics on grid environments, as described on Section 3.

6. RELATED WORK

The Core Grid Ontology (CGO), described in [16], provides a common knowledge base about grid systems. CGO is an extensible grid system ontology that expresses fundamental grid-specific concepts and relationships. However, our focus is on improving grid tasks, such as discovering applications and suitable resources on the grid, while exploring reasoning capabilities and query languages.

Cannataro [2] describes the Data Mining for Grid Programming project which uses an ontology (DAMON) for the data mining domain. DAMON exploits only one specific grid application domain, namely, data mining, while we propose a more general approach to the management of grid content and resources.

Matching of grid resources and application requirements is largely explored by semantic grid applications. Brook et

al. [1] propose a semantic approach to build a broker of resources described by different grid middleware, aiming to provide seamless access to resources on the grid. Tangmunarunkit et al. [14] introduce a similar approach. In a similar way, [11] describes an asymmetric mapping between applications and resources based on incomplete application requirement description. Somasundaram et al. [13] propose an ontology template component integrated with a grid service broker. All of them propose an ontology-based matching of task requirements and grid resource policies. Our work, in comparison, aims to integrate the search and matching of applications and grid resource description within a comprehensive perspective.

In our approach we borrow the general idea of *decoupled scheduling* from [17]. However, while they try to reduce the amount of resources selected aiming to optimize some scheduling algorithm, our work aims to identify the maximum amount of suitable resources for satisfying an user request in a grid cluster.

7. CONCLUSION AND FUTURE WORK

This work explores the use of ontologies to describe core concepts related to grid applications and resources. Such ontologies are related between them through an *import* mechanism and can be extended either by adding new classes and individuals to a specific ontology or by connecting new ontologies derived from the top level ontologies.

We described some important scenarios for exploring the knowledge base that can be constructed through the use of the proposed ontologies and the inference mechanisms that can improve the performance of grid scheduling and leverage the grid usage as a distributed and integrated application repository, leading to better discovery and composition of software artifacts. We propose a *Semantic Grid Integration Architecture* based on the Protégé-OWL API and the Pellet reasoner to be used for grid knowledge base maintenance and to explore the described scenarios. A prototype was developed using the Integrade grid middleware as its base.

We also evaluated our approach, comparing the resource matching results for application submissions obtained with and without the use of semantics, sustaining the conclusion that the use of semantics can lead to substantial gains in resource management on grid environments.

The future directions of this work includes the investigation of the semantic relations between ontologies which have intersection (mapping) and between disjointed ontologies (e-connections), which comprehends a relevant aspect in order to improve scalability of the knowledge base.

8. REFERENCES

[1] Brooke, J., Fellows, D., Garwood, K., and Goble, C. Semantic matching of grid resource descriptions. In *AxGrids 2004* (January 2004), vol. 3165, pp. 240–249.

[2] Cannataro, M., and Comito, C. A data mining ontology for grid programming. In *Workshop on Semantics in Peer-to-Peer and Grid Computing* (Budapest, Hungary, 2003), pp. 113–134.

[3] Chandrasekaran, B., Josephson, J., and Benjamins, V. What are ontologies, and why do we need them? *Intelligent Systems and Their Applications, IEEE 14* (jan 1999), 20–26.

[4] Consortium, W. W. W. W. Web ontology language (owl). http://www.w3.org/2004/OWL, april 2007.

[5] de Roure, D., Jennings, N. R., and Shadbolt, N. R. The semantic grid: Present, past, and future. In *Proceedings of IEEE* (march 2005), vol. 93, pp. 669–681.

[6] Goldchleger, A., Kon, F., Goldman, A., Finger, M., and Bezerra, G. C. InteGrade: Object-Oriented Grid Middleware Leveraging Idle Computing Power of Desktop Machines. *Concurrency and Computation: Practice and Experience 16* (March 2004), 449–59.

[7] Knublauch, H., Fergerson, R. W., Noy, N. F., and Musen, M. A. The protégé owl plugin: An open development environment for semantic web applications. In *ISWC 2004* (Hiroshima, Japan, 2004).

[8] Mir, A., Hanmandlu, M., and Tandon, S. Texture analysis of ct images. *Engineering in Medicine and Biology Magazine, IEEE 14* (1995), 781– 786.

[9] Patel-Schneider, P. F., and Horrocks, I. Position paper: a comparison of two modelling paradigms in the semantic web. In *WWW '06* (Edinburgh, Scotland, May 2006), ACM Press, New York, NY.

[10] Roure, D. D. A brief history of the semantic grid. In *Semantic Grid: The Convergence of Technologies* (2005), C. Goble, C. Kesselman, and Y. Sure, Eds., no. 05271 in Dagstuhl Seminar Proceedings, IBFI, Schloss Dagstuhl, Germany.

[11] Siddiqui, M., Fahringer, T., Hofer, J., and Toma, I. Grid resource ontologies and asymmetric resource-correlation. In *2nd International Conference on Grid Service Engineering and Management (NODe/GSEM)* (Erfurt, Germany, September 19-22 2005), G. S. of Informatics, Ed., vol. 69, Lecture Notes in Informatics, pp. 205–219.

[12] Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and Katz, Y. Pellet: A practical owl-dl reasoner. Tech. Rep. 2005-68, Maryland, USA, 2005.

[13] Somasundaram, T. S., R.A.Balachandar, Kandasamy, V., Buyya, R., Raman, R., N.Mohanram, and S.Varun. Semantic-based grid resource discovery and its integration with the grid service broker. In *ADCOM 2006: Proceedings of 14th International Conference on Advanced Computing & Communications* (2006), pp. 84 – 89.

[14] Tangmunarunkit, H., Decker, S., and Kesselman, C. Ontology-based resource matching in the grid - the grid meets the semantic web. In *LNCS* (Sep 2003), vol. 2870, pp. 706–721.

[15] Vidal, A. C., J. R. Braga, J., Kon, F., and Kofuji, S. T. Defining and exploring a grid system ontology. In *Proceedings of the 4th international workshop on Middleware for grid computing* (New York, NY, USA, 2006), ACM Press, p. 16.

[16] Xing, W., Dikaiakos, M. D., and Sakellariou, R. A core grid ontology for the semantic grid. In *CCGrid 2006* (Singapore, May 2006), IEEE, pp. 178–184.

[17] Zhang, Y., Mandal, A., Casanova, H., Chien, A. A., Kee, Y.-S., Kennedy, K., and Koelbel, C. Scalable grid application scheduling via decoupled resource selection and scheduling. In *Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2006)* (2006), vol. 1, pp. 568 – 575.