Bringing Knowledge to Middleware – Grid Scheduling Ontology

P. Wieder
`ph.wieder@fz-juelich.de`

Central Institute for Applied Mathematics
Research Centre Jülich
52425 Jülch, Germany

W. Ziegler
{`wolfgang.ziegler@scai.fraunhofer.de`

Fraunhofer Institute SCAI
Department for Web-based Applications
53754 Sankt Augustin, Germany

CoreGRID Technical Report
Number TR-0008
May 19, 2005

Institute on Resource Management and Scheduling

CoreGRID - Network of Excellence
URL: http://www.coregrid.net

CoreGRID is a Network of Excellence funded by the European Commission under the Sixth Framework Programme

Project no. FP6-004265

Bringing Knowledge to Middleware – Grid Scheduling Ontology

P. Wieder
`ph.wieder@fz-juelich.de`

Central Institute for Applied Mathematics
Research Centre Jülich
52425 Jülch, Germany

W. Ziegler
{`wolfgang.ziegler@scai.fraunhofer.de`

Fraunhofer Institute SCAI
Department for Web-based Applications
53754 Sankt Augustin, Germany

CoreGRID TR-0008
May 19, 2005

Abstract

The Grid paradigm implies the sharing of a variety of resources across multiple administrative domains. Assuming that such an environment is highly dynamic, it is essential to abstract potential drawbacks away from resource users and resource providers. One crucial aspect in designing and operating Grids to gain the respective abstraction is the provision of a sophisticated scheduling and resource management framework. Experience shows that scheduling a single resource like an HPC system is already a challenge, but the co-ordinated scheduling of multiple resources to automatically process a complex work flow is impossible if the capabilities of resources are not a priori known. We propose to make scheduling-specific parts of such knowledge exploitable by introducing a scheduling domain ontology. This ontology provides a common semantic understanding to be shared between the components involved in the scheduling process. By agreeing upon and integrating such an ontology we increase the automation level and make usage and administration of Grids easier.

1 Introduction

Hiding the complexity of future generation Grids from the end-user is one major task to be solved en route to make the daily use of Grids real. Relieving the system administrator of the burden to manually tweak his Grid system to integrate and deal with external, usually unknown, resources is the other side of the coin when making Grids a technology for a broad community. These requirements are based on an important observation which could be made in Grid testbeds and production-oriented Grids over the last years: Users and system administrators of Grid systems are well advised to have specific knowledge apart from their default usage profile about major aspects of their Grid environment (as there are architecture, resources, policies, etc.). In case of users this regularly implies detailed knowledge about the structure of jobs and work flows, while system administrators struggle e.g. with the integration of resources into diverse resource management systems.

Scheduling and resource management is a crucial aspect to overcome these drawbacks and make Grids usable and easy to administrate, especially if one faces challenges like seamlessness, resource autonomy and platform independence [1]. Reconsidering the above-mentioned examples from a resource management point of view, the user needs specific knowledge in order to find and select the appropriate sources for information retrieval and negotiation to make the decision where and when to use Grid resources for her job. Administrators, however, have to manually provide resource knowledge in a format consumable by users for every resource they integrate into a Grid.

Delegating these processes and responsibilities from humans to machines raises the need to provide this knowledge in a machine-processable form. This paper focuses on concepts and work in progress to create knowledge which is needed for scheduling in Grids and therefore helps to facilitate the issues addressed above: the Grid Scheduling Ontology (GSO). The roots of this work originate in the Global Grid Forum (GGF, [2]) and will most likely converge to a GGF Working Group (WG), probably in close cooperation with a planned GGF Research Group (RG) on resource ontologies. Requirements for this work come from and results of this work will be used in several national and European projects, as there is for example VIOLA [3].

Subsequently we picture the environment the Grid Scheduling Ontology will be deployed to and derive the requirements based on some usage scenarios in this environment. Following that, we introduce the instruments which will be applied to realise a Grid Scheduling Ontology (see Section 3). The ontology itself, its objectives and realisation are subject of Section 4, while the final part of this document summarises problem and solution and provides a brief outlook onto the future of resource management and scheduling systems.

2 Environment and Requirements

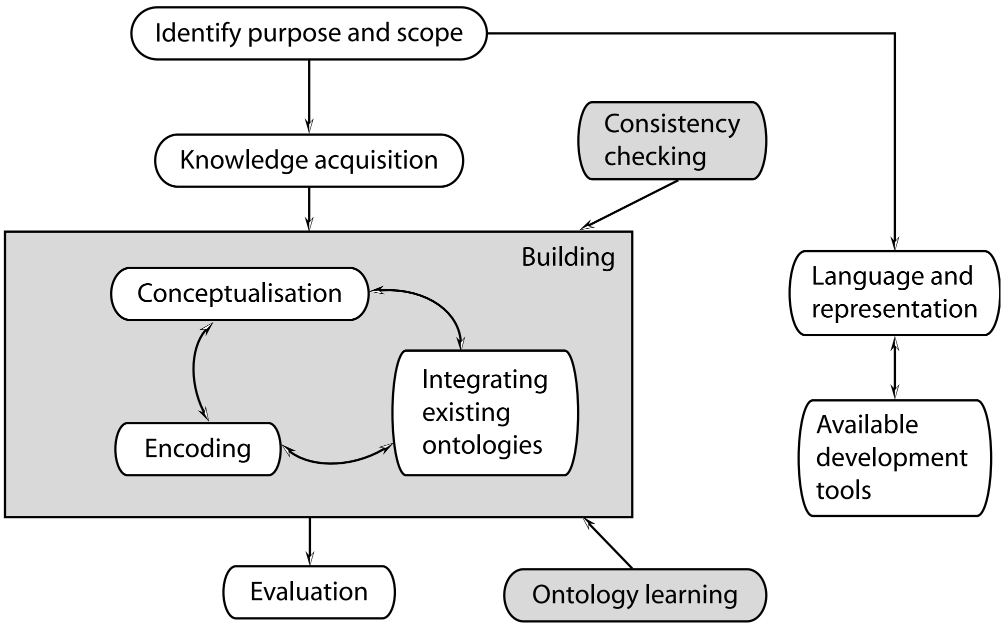
The environment we are dealing with is scheduling and resource management of Grids and our primary focus is the semantic description of entities related to scheduling. Thus we concentrate on an ontology for the Grid scheduling domain.

Grids of today more and more integrate heterogeneous hardware and operating systems located in different administrative domains. Taking into account heterogeneity, site autonomy, and site policies, scheduling of a single resource in this environment is already a challenging task. Scheduling of several independent resources to execute a complex work flow with temporal dependencies is hardly possible with today's Grid middleware if the resources' capabilities and constraints are not known in advance. To perform such a scheduling task knowledge about and understanding of the environment is required on several levels:

- Single resources are described in a site-dependent way.
- Local scheduling systems provide different formats and interfaces to describe and manage resources.
- A meta-scheduler uses specific methods to map a user request to the requests addressed to the different local scheduling systems [4].

As implicitly mentioned above human involvement in the scheduling process can be basically classified into user and provider roles. The respective classification on machine level divides according to [5] the resource space into Resource Requesters (RR) and Resource Providers (RP). Please note that an entity can be a RR and a RP at the same time as the example of a meta-scheduler shows, which acts as a resource requester to underlying schedulers but may take the provider role with respect to superordinated schedulers. With regard to these classifications the main requirement is to automise the mapping from Resource Requester space to Resource Provider space (and vice-versa) and at the same time reduce the manual intervention of users and providers in the scheduling and resource management process.

At this point resource ontologies come into play as potentially useful instruments that provide a sophisticated approach to categorise and draw relationships between the various ways resources and services are described and requested today, finally leading to a shared and common understanding of resources and services. A scheduling domain ontology seems to be a promising way to introduce knowledge exploitable by machines into the scheduling process in Grids. If such an ontology is shared among the resource brokers, resource management systems, and schedulers the discovery of resources or services and the mapping of user requests to resources may become less arbitrarily. At the same time negotiation between sites with different resources available under different scheduling policies will become more distinct and may be carried out automatically by a meta-scheduler.



Identify purpose and scope
Knowledge acquisition
Consistency checking
Building
Conceptualisation
Integrating existing ontologies
Encoding
Language and representation
Available development tools
Evaluation
Ontology learning

Figure 1: Ontology building life-cycle [6]

3 Process and Instruments

The process of building an ontology is pictured in Fig. 1. After defining purpose and scope of the ontology a major prerequisite to be satisfied before actually building the ontology is the knowledge acquisition step. Apart from that the language and representation of the ontology must be fixed on a technical level and subsequently the appropriate tools have to be selected. The building of the ontology itself is an iterative process where the three steps conceptualisation, encoding, and integration of existing ontologies are repeatedly performed (conceptualisation in the case means defining classes that represent domain concepts, determining their attributes, and assessing the relations between the classes). This is the ontology learning phase where each iteration is concluded with a consistency check of the emerging ontology. Once the ontology appears to comprise the domain knowledge in a consistent way the building process is followed by an evaluation of the ontology. In the succeeding paragraphs we exemplarily describe the knowledge acquisition step and the language and tools we have selected as instruments to realise the Grid Scheduling Ontology.

3.1 Knowledge acquisition

The *Grid Scheduling Dictionary – Terms and Keywords* [7] is an informational document produced by the Global Grid Forum and indexed as Grid Forum Document 11 (GFD.11). The purpose of the dictionary is, as stated in the Scheduling Dictionary Working Group's (SD-WG) charter, to "Create a dictionary to define common terms used by various schedulers, both local and grid-level" [8]. The dictionary contains inter alia a linked list of terms and their definitions. The term "Scheduling" for example is defined as "The process of ordering tasks on compute resources and ordering communication between tasks. Also, known as the allocation of computation and communication 'over time' ".

The purpose and scope of the Grid Scheduling Dictionary make it an ideal input to the knowledge acquisition process. In addition the format of a dictionary which contains terms, their definite description and relations between the terms reduces the effort to transfer this domain-specific piece of knowledge from a human-readable into a machine-processable form. Apart from the dictionary command-line parameters from interfaces and APIs of available scheduling systems have been considered as input as well as potential new terms which evolved since the dictionary has been created.

3.2 Semantic Markup Languages and Tools

In general machine processable ontologies are created using semantic markup languages. Compared to a markup language the semantic markup language includes additional information attempting to encode the meaning of the content described using the markup language. These languages have evolved over time each new one adding another layer with higher abstraction and thus more ease of use and more functionality (see Fig. 2). As depicted, XML and XML Schema are providing the basis for all semantic markup languages by defining the syntax rules for markup languages.

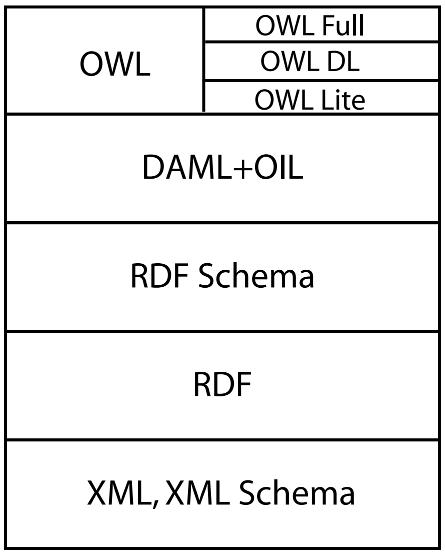


Figure 2: Semantic markup languages stack

On top of the XML/XML Schema layer RDF [9] is build which is the simplest of the markup languages. An RDF document is basically made of statements consisting of subject, predicate, object, also called triples, and attribute value pairs. As RDF was difficult to use for less experienced users, the RDF Schema specification [10] (RDFS) was created on top of RDF providing a formal definition of RDF and adding classes and properties.

In RDF/RDFS everything that is named with a Uniform Resource Identifier (URI) is a resource and may be accessed through this URI. RDF resources can be web pages, computer codes, data, hardware, algorithms, research groups, etc. On top of RDFS several other languages are build, each defined in terms of the respective lower layer:

- DAML+OIL [11] is a combination of the DARPA Agent Markup Language (DAML) ontology language (DAML-ONT) and OIL, which is the ontology inference layer for RDFS.
- The Web Ontology Language [12] (OWL) is based on DAML+OIL and is a W3C recommendation since 2004. OWL provides three increasingly expressive sublanguages, as there are OWL Lite, OWL DL and OWL Full.

Our language of choice is OWL DL since it is based on Description Logics and therefore allows automated reasoning over an ontology. Due to its richer language OWL Full does not allow this, while the OWL Lite syntax is too simple to fulfil the requirements of the Grid Scheduling Ontology.

While it is possible to write RDF XML (or DAML+OIL or OWL) using a common text editor this work is tedious and feasible only for small and simple ontologies. To overcome this limitation a number of tools have been developed supporting the creation and maintaining of complex DAML+OIL or OWL ontologies: OntoMat, Chimaera, PC Pack, Protégé, and others, including commercially licensed products. For the Grid Scheduling Ontology we decided to use Protégé [13] (see Fig. 3) from the University of Stanford as an authoring tool for several reasons:

- Protégé supports the creation of OWL ontologies.
- It is extensible through plug-ins.

- It is based on HP Lab's Jena package which might later be used as triple-store with query functions.
- Protégé has built-in support for the RACER [14] inference engine and reasoning system.
- It is written in Java and available under an Open Source license.

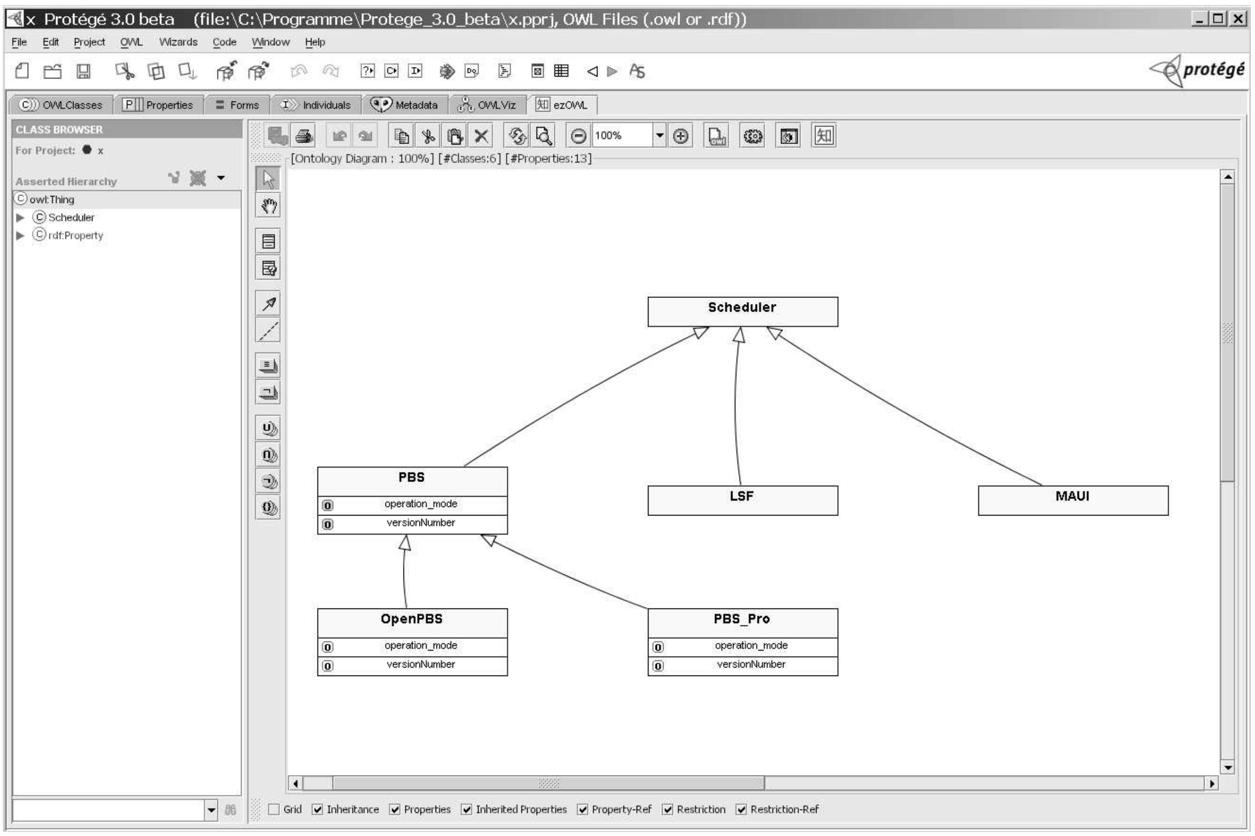


Figure 3: The Protégé ontology editor

4 Grid Scheduling Ontology

The question to be answered before naming the objectives for defining a Grid Scheduling Ontology is "What is an ontology?" This may be an easy discipline for people involved in Semantic Web or Semantic Grid work, but experience teaches that the term ontology is in general unknown to domain experts providing the knowledge to create one. There is no universally valid definition, therefore some widely-used are listed here:

- "... for them [Artificial-intelligence and Web researchers] an ontology is a document or file that formally defines the relations among terms. The most typical kind of ontology for the Web has a taxonomy and a set of inference rules." [15]
- "A specification of a representational vocabulary for a shared domain of discourse – definitions of classes, relations, functions, and other objects – is called an ontology." [16]
- " 'Ontology' the term used to refer to the shared understanding of some domain of interest which may be used as a unifying framework ..." [17]

4.1 Challenges and Objectives

As Grids are more and more comprising heterogeneous hardware and operating systems while stretching across different administrative domains, scheduling of a single resource has already become a challenging task. Automatic scheduling of several resources with different properties to execute a complex work flow with temporal dependencies is not possible if the resources' capabilities and constraints are not known in advance. This knowledge has several facets: the site-dependent way of describing resources, the differences between local scheduling systems in describing and managing resources and the methods a meta-scheduler uses to map the single user request to the individual requests addressed to these local scheduling systems.

Resources and resource requests today are usually described using descriptive and constraint languages. Different technologies available in Grid systems are used to ensure that the exposed capabilities of a resource match the ones requested: exact syntactic matching, as done in Condor [18], syntactic translation, as done in Globus [19], or database lookup and mapping, as done in UNICORE [20]. The major advantage of such approaches is the little overhead as the knowledge is already built in the system. Disadvantages are:

- The development of Resource Provider space and Resource Requester space descriptions (see also Section 2) must be synchronised.
- The low flexibility to react on changes in RP or RR.

Using ontologies certainly holds overhead creating and maintaining the ontologies (although once published, an ontology might be used without additional effort in other environments, too). On the other hand there are several advantages in the usage of an ontology:

- The technology developed for the Semantic Web can be exploited.
- The development of RP and RR space ontologies may happen independently.
- The high flexibility to react on changes in RP or RR.

This comparison of advantages and disadvantages of the different approaches shows that ontologies come up as potentially useful instruments which provide a sophisticated approach to categorise and draw relationships between the various ways resources and services are described today. The objective of a Grid Scheduling Ontology and the services exploiting it is to introduce knowledge exploitable by machines into the scheduling process in Grids. Recapturing the definitions cited above we can rephrase the objective of introducing a Grid Scheduling Ontology as to:

Drive the evolution of a shared understanding of the Grid scheduling domain usable for various forms of machine and human interactions.

When such an ontology will be shared among the resource brokers, resource management systems, schedulers and the corresponding user agents or clients of the scheduling systems, the processes of finding resources or services and mapping user requests to resources in the Grid may become easier. While there are already several examples in different projects for using ontologies in the process of detecting resources in the Grid and determine their capabilities [21] less has been done dealing with the resources of the scheduling and resource management domain itself. The Grid Scheduling Ontology aims to fill this gap and will allow negotiation between sites with different resources available under different scheduling policies to become more distinct and carried out automatically by a meta-scheduler.

4.2 Realisation

One driving force for the development and usage of ontologies, languages and tools was the idea of a Semantic Web. Originating from a Web (Service) environment ontologies are easy to integrate into the future versions of service-oriented and WSRF-based Grid systems like upcoming implementations of UNICORE or the Globus Toolkit. At the same time local schedulers, meta-schedulers, and resource management systems deployed as Web – or Grid Services will become available as results of different projects.

After the meta-scheduler receives a request to schedule a job comprising multiple resources the meta-scheduler will start querying the individual local schedulers about their capabilities. So if the inference engine returns for example "NQS and OpenPBS are schedulers not capable of doing advance reservation" the meta-scheduler is able to decide that a remote scheduling system (controlling a resource needed for a job) that is exposed as "NQS" is not suitable to schedule a component of an application that has to run in parallel with other components using other resources. The RACER inference engine will be used to find out for each scheduler to which class of schedulers it belongs and whether the scheduler has the necessary capabilities. Based on this knowledge the meta-scheduler starts negotiations with the appropriate local schedulers. The set of appropriate schedulers may be empty, of course, because none of the resources available has a local scheduler with the necessary capabilities, e.g. advance reservation or interactive use of the resource.

4.3 Example Use Case

The first use of the ontology will be in the framework of the VIOLA project [3] where a meta-scheduler makes use of the ontology to identify remote scheduling systems and their capabilities. VIOLA's first generation Meta-Scheduler architecture focuses on the scheduling functionality while minimizing changes to the UNICORE system (for a description of the basic UNICORE architecture please refer to [20]). As depicted in Fig. 4 in light grey, the system comprises of the Agreement Manager, the Meta-Scheduler itself [22], the Scheduling Ontology and a Meta-Scheduling plug-in (which is part of the client and not pictured separately). Before submitting a job to a Usite, the Meta-Scheduling plug-in and the Meta-Scheduler exchange the data necessary to schedule the resources needed. The Meta-Scheduler is then (acting as a Agreement Consumer in WS-Agreement terms [23]) contacting the Agreement Manager to request a certain level of service, a request which is translated by the Manager into the appropriate commands of the local scheduler once the remote resource management system/scheduler is recognized and its capabilities meet the requirements to schedule a component of the managed distributed job. The ontology is used for identification of the appropriate remote system and its capabilities. At a later stage VIOLA will make use of the ontology to translate the job request into the format requested by a local scheduler not accessible via UNICORE. Once all resources are reserved at the requested time the Meta-Scheduler notifies the UNICORE Client via the Meta-Scheduling plug-in to submit the job. This framework will also be used to schedule the interconnecting network, but potentially every resource can be scheduled if a respective Agreement Manager is implemented and the Meta-Scheduling plug-in generates the necessary scheduling information.

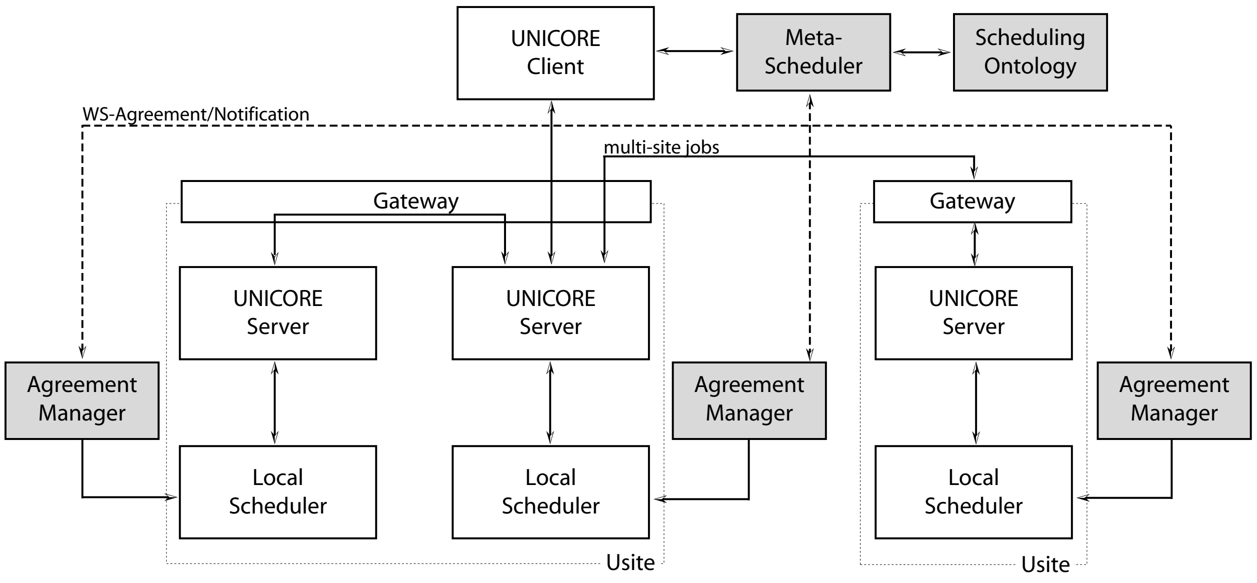


Figure 4: The VIOLA Meta-Scheduling architecture

5 Conclusions and future work

The work described here marks a starting point. We know about both the shortcomings of scheduling solutions in current Grid systems and the potential of ontologies. Based on that knowledge and the fact that no ontology for the

Grid scheduling domain exists we started to create one. Here we present a concept for introducing domain-knowledge into scheduling middleware making scheduling-specific parts of such knowledge exploitable encoded into a scheduling domain ontology. This ontology provides a common semantic understanding to be shared between the components involved in the scheduling process, it increases the automation level, and makes usage and administration of Grids easier. As stated above creating an ontology is a cyclic process that will evolve in several phases. In the first phase we have achieved the following necessary steps (see also Section 3) to create the ontology:

- We identified purpose and scope of the ontology.
- We selected OWL DL as the appropriate language.
- We evaluated and selected the development tools to be used in the process.
- We finalised the initial knowledge acquisition.
- We started with the conceptualisation of the ontology.

There are a number of open issues that will be tackled once the ontology's conceptualisation has been completed. Some of them are related to the contents of the ontology, others arise from the operational environment. It is necessary to examine existing resource ontologies (e.g. those from the DataTAG project [24]) with respect to their re-usability. This will also help to make sure the Grid Scheduling Ontology matches general requirements for a Grid resource ontology, a task which is likely to be carried out in cooperation with the respective group at GGF (see Section 1). Furthermore it is of interest to evaluate the Grid Scheduling Ontology's relationship to other ontologies which serve similar needs in a service-context (see e.g. [25], [26]). Regarding the operational aspects of the ontology it will be necessary to design and create adapters for schedulers that cannot be modified and integrate the respective components in existing scheduling and resource management systems.

At the end of the first phase the Grid Scheduling Ontology will be published and evaluated in a prototype scheduling environment. The results of this evaluation will serve as input to the next phases of enhancement and refinement.

References

[1] R. Menday and Ph. Wieder. GRIP: The Evolution of UNICORE towards a Service-Oriented Grid. In *Proc. of the 3rd Cracow Grid Workshop (CGW'03)*, Oct. 27-29, 2003.

[2] C. Catlett, W. Johnston and I. Foster. *Global Grid Forum Structure*. Grid Forum Document GFD.2, Global Grid Forum, 2002.

[3] VIOLA – Vertically Integrated Optical Testbed for Large Application in DFN. Project web site, 2005. Online: http://www.viola-testbed.de/.

[4] J. Schopf. Ten Actions when Grid Scheduling. In *Grid Resource Management* (J. Nabrzyski, J. Schopf and J. Weglarz, eds.), pages 15-23, Kluwer Academic Publishers, 2004.

[5] J. Brooke, K. Garwood and C. Goble. Interoperability of Grid Resource Descriptions: A Semantic Approach. In *Proc. of the GGF 9 Semantic Grid Workshop*, 2003.

[6] C. Goble and N. Shadbolt. *Ontologies and the Grid*. Tutorial held at GGF 4, 2002. Online: http://www.semanticgrid.org/presentations/ontologies-tutorial/.

[7] M. Roehrig, W. Ziegler and Ph. Wieder. *Grid Scheduling Dictionary of Terms and Keywords*. Grid Forum Document GFD.11, Global Grid Forum, 2003.

[8] M. Roehrig, W. Ziegler and Ph. Wieder. Grid Scheduling Dictionary WG Charter, 2003. Online: https://forge.gridforum.org/projects/sd-wg/document/ggf-sd-charter-final.html/en/1.

[9] G. Klyne and J.J. Carroll. *Resource Description Framework (RDF) – Concepts and Abstract Syntax*. W3C Recommendation, Feb. 2004.

[10] M. Klein, J. Broekstra, D. Fensel, F. v. Harmelen and I. Horrocks. Ontologies and Schema Languages on the Web. In *Spinning the Semantic Web* (D. Fensel, J.A. Hendler, H. Lieberman and W. Wahlster, eds.), pages 95-139, The MIT Press, 2003.

[11] D. Connolly, F. v. Harmelen, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein. *DAML+OIL (March 2001) Reference Description*. W3C Note, Mar. 2001.

[12] D.L. McGuinness, F. v. Harmelen. *OWL Web Otology Language Overview*. W3C Recommendation, Feb. 2004.

[13] N.F. Noy, M. Sintek, S. Decker, M. Crubezy, R.M. Fergerson, M.A. Musen. Creating Semantic Web Contents with Protégé-2000. *IEEE Intelligent Systems* **16**(2):60-71, 2001.

[14] V. Haarslev and R. Möller. RACER System Description. In *Automated Reasoning : First International Joint Conference (IJCAR 2001)*. Volume 2083 of Lecture Notes in Computer Science, pages 701-?, Springer, 2001.

[15] T. Berners-Lee, J. Hendler and O. Lassila. The Semantic Web. *Scientific American.com*, 2001.

[16] T.R. Gruber. *Translation Apporach to Portable Ontology Specifications*. Knowledge Systems Laboratory Technical Report KSL 92-71, Stanford University, 2002.

[17] M. Uschold, and M. Gruniger. *Ontologies: Principles, Methods and Applications*. Artificial Intelligence Applications Institute Technical Report AIAI-TR-191, University of Edinburgh, 1996.

[18] D. Thain and M. Livny. Building Reliable Clients and Servers. In *The Grid: Blueprint for a New Computing Infrastructure* (I. Foster and C. Kesselman, eds.), Morgan Kaufmann, 2003.

[19] The Globus Alliance. Project web site, 2005. Online: http://www.globus.org.

[20] D. Erwin, ed. *UNICORE Plus Final Report – Uniform Interface to Computing Resources*. UNICORE Forum e.V., ISBN 3-00-011592-7, 2003.

[21] H. Tangmunarunkit, S. Decker and C. Kesselman. Ontology-Based Resource Matching in the Grid – The Grid Meets the Semantic Web. In *Proc. of the International Semantic Web Conference 2003*. Volume 2870 of Lecture Notes in Computer Science, pages 706-721. Springer, Sep. 2003.

[22] G. Quecke and W. Ziegler. MeSch – An Approach to Resource Management in a Distributed Environment. In *Proc. of 1st IEEE/ACM International Workshop on Grid Computing (Grid 2000)*. Volume 1971 of Lecture Notes in Computer Science, pages 47-54, Springer, 2000.

[23] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, J. Pruyne, J. Rofrano, S. Tuecke and M. Xu, *Web Services Agreement Specification*. Grid Forum Draft, Version 1.1, Global Grid Forum, 2004.

[24] DataTAG - Research & technological development for a Data TransAtlantic Grid. Project web site, 2005. Online: http://www.datatag.

[25] L. Li and I. Horrocks. A Software Framework For Matchmaking Based on Semantic Web Technology. In *Proc. of the Twelfth International World Wide Web Conference (WWW2003)*. 2003.

[26] C. Wroe, R. Stevens, C. Goble, A. Roberts and M. Greenwood. A suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data. *International Journal of Cooperative Information Systems special issue on Bioinformatics*. Mar. 2003.