Available online at www.sciencedirect.com
SCIENCE DIRECT®
J. Parallel Distrib. Comput. 65 (2005) 1533–1541
Journal of Parallel and Distributed Computing
www.elsevier.com/locate/jpdc

Introduction of semantic matchmaking to Grid computing

Simone A. Ludwig[a,*], S.M.S. Reyhani[b]

[a] School of Computer Science, Cardiff University, Cardiff CF24 3AA, UK
[b] Department of Information Systems and Computing, Brunel University, Uxbridge UB8 3PH, UK

Received 11 January 2004; received in revised form 23 May 2005; accepted 31 May 2005
Available online 14 July 2005

Abstract

The fundamental problem the Grid research and development community is seeking to solve is how to coordinate distributed resources amongst a dynamic set of individuals and organizations in order to solve a common collaborative goal. The problem of service discovery in a Grid environment arises through the heterogeneity, distribution and sharing of the resources in different virtual organizations. This paper proposes a service discovery framework which is based on semantics. It gives an example of the Grid Job Submission Service written in DAML-S in order to show how service ontologies are implemented. This semantic approach allows a more flexible and dynamic matching mechanism based on semantic descriptions stored in ontologies.

Keywords: Ontology; Matching mechanism; Grid service discovery

1. Introduction

The computational speed of individual computers has increased by about 1 million times in the past 50 years. However, they are still not fast enough for more and ever more scientific problems. For example, in a few physics applications, data is produced by the fastest contemporary supercomputer. The analysis of this data would need much more computational power than presently available.

In the mid 1990s, Ian Foster and Carl Kesselman proposed a distributed computing infrastructure for advanced science and engineering, which they called "The Grid". The vision behind the Grid is to supply computing and data resources over the Internet seamlessly, transparently and dynamically when needed, such as the power grid supplies electricity to end users. The Grid originated from trying to solve the information and computational challenges of science [10].

Locating a service or a device on demand is a challenging task. A variety of service discovery systems exist that enable an application to discover a service. Most of these systems support an attribute-based discovery as well as a simple name lookup, also called matching, to locate a service. Usually only a set of primitive attribute types, such as string and integer, are used to characterize a service. The service discovery process is therefore primarily done by type matching, based on string or integer comparison. The existing service discovery systems lack the ability of inexact matching. In Grid environments where so many different implementations of services are available, that might vary in name and functionality, a more powerful matching process is desirable. The need for semantic matching becomes increasingly important as more and more services are developed.

HEP experiments such as ALICE [1], ATLAS [4], CMS [7] and LHCb [13] require huge distributed computational infrastructures to satisfy their data processing and analysis needs. These High Energy Physics (HEP) experiments want to access the Grid in order to process their petabytes of data necessary for their experimental evaluations. The HEP experiments are different independent applications which want to make use of the Grid. Therefore, there is a need for semantic service discovery in a Grid environment.

* Corresponding author.
E-mail addresses: Simone.Ludwig@cs.cardiff.ac.uk (S.A. Ludwig), smsreyhani@ieee.org (S.M.S. Reyhani).

0743-7315/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2005.05.022

Semantic Matching focuses on the problem of locating services on the basis of the capabilities that they provide. The solution to this problem requires a language to express the capabilities of services, and the specification of a matching algorithm between service advertisements and service requests, one that recognizes when a request matches an advertisement. The term ontology is useful in the Grid services context, where it describes the need for the provider of a service and the user of that service to share a common understanding of what capabilities the service offers and how they can be put to use [15]. This is described in more detail in the next section.

This paper is organized as follows. Section 2 gives an introduction to the term ontology and its development. In Section 3, the service discovery framework including the matching mechanism is presented. Furthermore, part of the implementation of the ontology for the Grid Job Submission Service written in DAML-S is described. Section 4 presents an account of work related to this subject. Finally, Section 5 concludes this paper with a summary of the proposed approach containing a comparison with the related work.

2. Background to ontology

When two or more parties seek a common understanding of something, they must work together to ensure that there is a high degree of correlation and similarity between the details of their respective descriptions and definitions of what they are trying to agree on [23]. This implies that shared understanding requires shared definitions. As an example, the day-to-day human interactions are made possible by the fact that our society's members share common understanding and common values. This sharing of common understanding is categorized as the science of ontology, which involves the study of the general concepts and abstractions that make up the fundamental aspects of our world. Over the years, the term "ontology" has grown beyond its original philosophical use and its definition has been blurred slightly when applied to some areas of computing. The term is used in Artificial Intelligence, for example, to describe the fundamental components used to model the worlds that robots understand. Robots can understand only what can be represented as allowed by their ontologies.

Because we pay little attention to the ontology problem in our day-to-day human interactions, we usually take for granted our shared understanding and shared values that allow us to interact with relative ease. We tend not to consider it with respect to computing problems such as computing services.

Particularly, for Grid environments where service discovery is a significant issue, the need to share a common ontology becomes very important.

2.1. History of Ontology Development

A prerequisite for widespread use of ontologies is a joint standard for their description and exchange. RDF(S)

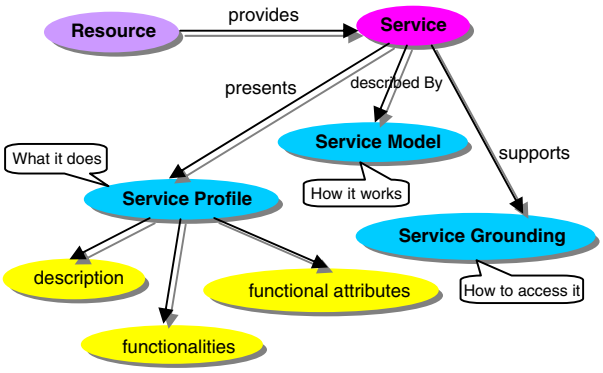

Fig. 1. Upper ontology for services.

(Resource Description Framework Schema) itself is an ontology/knowledge representation language which contains classes and properties (binary relations), range and domain constraints (on properties) and subclass and subproperty (subsumption) relations. RDF(S) is a relatively primitive language, however, more expressive power would clearly be necessary and desirable to describe resources in sufficient detail. Moreover, such descriptions should be amenable to automated reasoning if they are to be used effectively by automated processes [5].

These considerations led to the development of the Ontology Inference Layer (OIL) [9] and later to the design of DAML+OIL [8]. DAML+OIL is a more recent proposal for an ontology representation language that has emerged from work under DARPA's Agent Markup Language (DAML) initiative along with input from leading members of the OIL consortium. DAML+OIL is based on the original OIL language, but differs in a number of ways. DAML+OIL provide a greater interoperability on the semantic level. In this way, DAML+OIL extends the RDF(S) basic primitives for providing a more expressive ontology modeling language and some simple terms for creating inferences. In particular, DAML+OIL has moved away from the original frame-like ideas of OIL. It is an alternative syntax for a Description Logic (DL).

As part of the DARPA Agent Markup Language program, an ontology of services, called DAML-S [2] has been developed, which provides a set of basic concepts and relations for declaring and describing services, by utilizing the ontology structuring mechanisms provided by DAML.

The DAML-S service is characterized in three types which are service profile, service model and service grounding [17]. This is shown in Fig. 1. The service profile describes what the service does. It provides the type of information needed by a service requester to determine whether the service has the desired capabilities. The service model, also named process model, describes how the service works, i.e. how it is composed and what happens when the service is executed. A service grounding specifies the details of how a service can be accessed. The profile provides the information needed for the discovery of a service. The service model and grounding

Table 1
Description Properties of the Service Profile

Description properties	
serviceName	The name of the service.
intendedPurpose	A high-level description of what constitutes (typical) successful.
textDescription	A brief, human readable description of the service, summarizing what the service offers or what capabilities are being requested.
role	An abstract link to actors involved in the service execution.
requestedBy	A sub-property of *role* referring to the service requester.
providedBy	A sub-property of *role* referring to the service provider.

describe how service requesters and providers can access and interoperate with each other.

Service profiles consist of three types of information:

1. A human readable description of the service.
2. A specification of the functionalities that are provided by the service.
3. A list of functional attributes that provide additional information and requirements about the service that assist when reasoning about several services with similar capabilities.

Service functionalities are represented as a transformation from the inputs required to the outputs produced. For example, a news reporting service would advertise itself as a service that, given a date, will return the news reported on that date. Functional attributes specify additional information about the service, such as what guarantees response time or what accuracy it provides or the cost of the service. Table 1 and 2 list the properties defined by the service profile as described above.

3. Service discovery framework

Computational Grids, emerging as an infrastructure for the next generation computing, enable the sharing, selection and aggregation of geographically distributed heterogeneous resources for solving large-scale problems in science, engineering and commerce. As the resources in the Grid are heterogeneous and geographically distributed with varying availability and variety of usage and cost policies for diverse users at different times, priorities as well as goals of both users and owners vary with time. The management of resources and services in such a large distributed environment is a complex task. Therefore, service discovery plays an important role in such environments.

The following is a description of a proposed service discovery framework for Grid environments. The framework relies on an ontology description, that allows

Table 2
Functional attributes of the service profile

Functional attributes	
geographicRadius	Geographic scope of the service, either at the global scale (e.g. e-commerce) or at a regional scale (e.g. pizza delivery).
degreeOfQuality	Quality qualifications, such as providing the cheapest or fastest possible service.
serviceParameter	An expandable list of properties that characterize the execution of a service, such as averageResponseTime or invocationCost.
communicationThru	High-level summary of how a service may communicate, e.g. what communication language is used (e.g., KQML, SOAP).
serviceType	Broad classification of the service that might be described by an ontology of service types, such as B2B, B2C etc.
serviceCategory	Categories defined within some service category ontology. Such categories may include products, information services etc.
qualityGuarantees	Guarantees that the service promises to deliver, e.g. guaranteeing to provide a response within 3 minutes, etc.
qualityRating	Industry-based ratings, such as the "Dun and Bradstreet Rating" for businesses.

semantic matching and it is based on the concepts of the (LARKS) Language for Advertisement and Request for Knowledge Sharing matchmaker proposed by Sycara et al. [8]. The matching mechanism comprises of three matching stages. These are context, syntactic and semantic matching, whereas the service ontology provides the knowledge-base.

The difference between the LARKS and the Grid service discovery framework is that the Grid service discovery relies on DAML-S and its ontologies for the matching, while LARKS only uses the set of filters that progressively restrict the number of advertisements which are candidates for a match. The Grid service discovery framework achieves semantic matching as it relies on the three matching stages and also allows the flexibility of close matches by providing an ontology knowledge-base.

The description of the matching process in the service discovery matching framework is described below, and is followed by a description of the framework.

3.1. Matching Mechanism

An advertisement matches a request, when the advertisement describes a service that is sufficiently similar to the service requested [16]. The problem of this definition is to specify what "sufficiently similar" means. Basically, it means that an advertisement and a request are "sufficiently similar" when they describe exactly the same service. This

definition is too restrictive, because providers and requesters have no prior agreement on how a service is represented and additionally, they have very different objectives. A restrictive criterion on matching is therefore bound to fail to recognize similarities between advertisements and requests.

It is necessary to allow matching engines to perform flexible matches, those that recognize the degree of similarity between advertisements and requests in order to provide a softer definition of "sufficiently similar". Service requesters should be allowed to decide the degree of flexibility that they grant to the system. If they allow little flexibility, they reduce the likelihood of finding services that match their requirements, which means, they minimize the false positives, while increasing the false negatives. On the other hand, by increasing the flexibility of a match, they achieve the opposite effect, that is, they reduce the false negatives at the expense of an increase of false positives.

An additional problem related with performing flexible matches is that the matching engine is open to exploitation from advertisements and requests that are too generic in the attempt to maximize the likelihood of matching. For instance, a service may advertise itself as a provider of everything, rather than to be precise with what it does and similarly, the requester may ask for any service, rather than specifying exactly what it expects. The matching engine can reduce the efficiency of these exploitations by ranking advertisements based on the degree of a match supplied with the request.

Considering all these criteria, the matching engine should satisfy the following [16]:

(a) The matching engine should support flexible semantic matching between advertisements and requests based on the ontology available to the service and the matching engine.
(b) Despite the flexibility of the match, the matching engine should minimize false positives and false negatives. Furthermore, the requesting service should have some control on the amount of matching flexibility it allows to the system.
(c) The matching engine should encourage providers and requesters to be precise with their descriptions at the cost of either not being matched or being matched inappropriately.
(d) The matching process should be efficient which means that it should not burden the requester with excessive delays that would prevent its effectiveness.

The algorithm proposed aims to satisfy all four requirements. Semantic matching is based on DAML-S ontologies. The advertisements and requests refer to DAML-S concepts and the associated semantics. By using DAML-S, the matching process can perform implications on the subsumption hierarchy leading to the recognition of semantic matches despite their syntactical differences and difference in modeling abstractions between advertisements and requests. The use of DAML-S also supports accuracy, which means that no matching is recognized when the relation between the advertisement and the request does not derive from the DAML-S ontologies used by the registry. Furthermore, the semantics of DAML-S descriptions allow the definition of a ranking function which distinguishes multiple degrees of matching.

3.2. Service discovery matchmaker

Three components are necessary for the Grid service discovery matching. These are service provider, service requester and service discovery matchmaker. The sequence of interactions is as follows:

1. The service provider registers its service description in the service registry database.
2. The Grid application requests a Grid service and sends the request to the service discovery matchmaker.
3. The matchmaker returns the matches to the service requester.
4. The service requester decides then which service to use depending on the client's need and contacts the service directly.

Grid services are services that are provided in a Grid environment. These services are, for example, authentication, authorization, job submission, distributed job scheduling, resource optimization, data management, wide-area data transfer, file replication, resource management and resource monitoring. Grid services can be defined as a bundle or a collection of "simple" services. These services are registered in the service registry database.

The service requester consumes services offered by Grid service providers in the system. A request for any Grid service has to be sent to the matchmaker. In a Grid environment, service requesters are the applications which use the Grid services. These are e.g. the LHC-HEP experiments which want to access the Grid services in order to process their Petabytes of data necessary for their experimental evaluations.

The service matchmaker mediates between service requesters and service providers for some mutually beneficial cooperation. Each provider must first register with a registry, also called matchmaker. Service provider advertises their capabilities (advertisements) by sending some appropriate messages describing the kind of service they offer. Every request a matchmaker receives will be matched with its actual set of advertisements. If the match is successful, the matchmaker returns a ranked set of appropriate service providers to the requester.

In contrast to a broker, a matchmaker does not deal with the task of contacting the relevant providers, transmitting the service request to the service provider and communicating the results to the requester. This avoids data transmission bottlenecks, but it might increase the amount of interactions between service requesters and the matchmaker.

The matchmaker processes a received request in the following three basic steps:

(a) Every pair of request and advertisement has to go through several different matching stages during the

matching process where a kind of comparison of the request with all advertisements is performed.
(b) Decision of the service provider whose capabilities match best with the request depending on the specified algorithm and the defined service ontology.
(c) Providing information to the service requester by sending a contact address and related capability descriptions of the relevant service provider.

The ontology of a matchmaker is not equal to the union of local domain ontologies of all service providers who are actually registered at the matchmaker. Thus, a matchmaker has only partial knowledge and might not be up-to-date with the actual time of processing incoming requests. This is due to the fact that, for efficiency reasons, changes in the local ontology of a service provider might not be propagated immediately to all matchmaker it is registered at. These are the implications of a centralized service discovery model for the Grid service discovery matchmaker. A better solution is a decentralized service discovery model following the distributed nature of the Grid. In distributed systems components may fail, messages may be lost or services expire, hence a management for the lifetime of services must be provided to allow a secure service discovery process [14].

The architecture fulfils the matching criteria listed in Section 3.1 as follows. The Grid service discovery matchmaker supports flexible semantic matching between advertisements and requests based on the ontology available. Minimizing false positives and false negatives is achieved with three matching stages in combination with the well-defined ontology. The matching stages restrict the false positives and the ontology restricts the false negatives. The matching engine should encourage providers and requesters to be precise with their descriptions. To achieve this, the service provider follows an XML-based description to advertise its services and the service requester generates a query in a specified format. Defining the ontology and the matching stages precisely allows the matching process to be efficient.

3.3. Component description of the matchmaker

The matching process of the matchmaker is designed with respect to the criteria listed in the first section. The matching process is organized as a series of matching stages. This means that matching a given request into a set of advertisements consists of the following three matching stages:
(1) Context matching — Selects those advertisements in the context database that can be compared with the request in the same or similar context.
(2) Syntactic matching — This stage matches the request syntactically with any advertisement selected by the context matching.
(3) Semantic matching — This final stage matches the request from the context matching using semantic descriptions of service properties.

The context matching selects the appropriate context of an application domain. This stage is very important especially in a Grid environment where the different applications want to match Grid services and specify the service request in their own "application" language. Additional parameter are attached and forwarded to the following matching stage. The syntactic matching helps to find services which are well known and where the service requester either knows exactly the service name or where the service request is translated into the Grid services context via the context matching. In order to find the service a request to a registry is made where the service details are returned from. The semantic matching stage performs a match based on an ontology where the concepts and semantics of the services are described. This knowledge is inferred by a reasoning engine and the matched values are returned to the service requester. The actual service details can be queried from the registry again.

The service registry database contains all Grid services with the DAML-S service profile structure. The service ontology database contains the DAML-S service ontology that was described in Section 2. The service registry database is linked to the service ontology. The main benefits of providing a local ontology are as follows. The user can specify in more detail what is being requested or advertised, and particularly the matchmaker is able to make automated inferences on these additional semantic descriptions. This improves the overall quality of the matching process.

The semantic matching stage is responsible for matching the request semantically. This is performed as follows. The Grid services ontology is parsed by a DAML parser. The attributes and classes of DAML-S describe the concept of the ontology. It characterizes the service for advertisement, discovery and matching. The service request is being matched semantically by parsing the ontology. The DAML-S code facilitates effective parsing of service capabilities through its use of generic RDF(S) symbols compared to DAML-S specific symbols. With a defined set of rules, an inference engine reasons about the parameters parsed from the ontology. The result parameters of the inference process are forwarded to the registry where a lookup is performed.

3.4. Implementation of the ontology written in DAML-S

The Grid Job Submission Service (JSS) was implemented as part of the Grid ontology written in DAML-S and is described in detail as follows. JSS is responsible for the actual job management operations (submission, cancellation and monitoring) once the Resource Broker (RB) has chosen a suitable Computing Element for running the job. The RB is a middleware that supplies distributed clients with job execution in a heterogeneous computing environment such as the Grid. Client applications are provided with a set of interfaces for sending requests and receiving responses to and from the RB. The RB is responsible for carrying out tasks to satisfy the client requests. These tasks include e.g. interacting with the Replica Catalog and performing the job submission via the JSS.

```
<rdf:RDF
<daml:Ontology>
<daml:versionInfo>
$Id: JobSubmission-Service.daml, v0.1 01/11/2002 Simone Ludwig $
</daml:versionInfo>
<rdfs:comment This ontology represents DAML-S service description for the example of a Grid job submission.>
...
</daml:Ontology>
<service:Service rdf:ID="JobSubmissionAgent">
<!-- Reference to the Job Submission Profile -->
<service:presents rdf:resource="&js_profile;#Profile_JobSubmissionAgent"/>
<!-- Reference to the JobSubmission Process Model -->
<service:describedBy rdf:resource="&js_process;#JobSubmission_Process"/>
</service:Service>
</rdf:RDF>
```

Fig. 2. Code Fragment of JobSubmission-Service.daml.

```
<rdf:RDF
  <daml:Ontology about="">
    <daml:versionInfo>
      $Id: JobSubmission-Profile.daml, v0.1 01/11/2002 Simone Ludwig $
    </daml:versionInfo>
    <rdfs:comment>
      DAML-S Coalition: This profile description represents an example of a Grid job submission.
    </rdfs:comment>
    ...
  </daml:Ontology>
  <!-- #    Instance Definition of JobSubmission Example Advertisement   # -->
  <service:ServiceProfile rdf:ID="Profile_JobSubmissionAgent">
    <!-- reference to the service specification -->
    <service:isPresentedBy rdf:resource="&js_service;#JobSubmissionAgent"/>
    <!-- reference to the process model specification -->
    <profile:has_process rdf:resource="&js_process;#JobSubmission_Process"/>
    <profile:serviceName>JobSubmissionAgent</profile:serviceName>
    <profile:textDescription>
        This service provides the job submission service for the Grid environment.
    </profile:textDescription>
    <profile:providedBy>
      <profile:ServiceProvider rdf:ID="JobSubmission">
        <profile:name>EDGJobSubmission</profile:name>
      </profile:ServiceProvider>
    </profile:providedBy>
    ...
    <!-- Descriptions of the parameters -->
      <profile:input>
        <profile:ParameterDescription  rdf:ID="CreateJDL">
          <profile:parameterName> CreateJDL </profile:parameterName>
          <profile:restrictedTo rdf:resource="&concepts;#JDL"/>
          <profile:refersTo rdf:resource="&js_process;#createJobDescriptionLanguage_In"/>
        </profile:ParameterDescription>
      </profile:input>
      ...
      <profile:input>
        <profile:ParameterDescription  rdf:ID="UpdateBookkeeping">
           <profile:parameterName> UpdateBookkeeping </profile:parameterName>
           <profile:restrictedTo rdf:resource="&concepts;#Bookkeeping"/>
           <profile:refersTo rdf:resource="&js_process;#updateBookkeeping_In"/>
        </profile:ParameterDescription>
      </profile:input>
      <!-- The consequence of the jobsubmission is that the certificate is valid -->
      <profile:effect>
        <profile:ParameterDescription rdf:ID="ValidCertificate">
          <profile:parameterName> ValidCertificate </profile:parameterName>
          <profile:restrictedTo rdf:resource="&concepts;#HaveValidCertificate"/>
          <profile:refersTo rdf:resource="&js_process;#HaveValidCertificate"/>
        </profile:ParameterDescription>
      </profile:effect>
  </service:ServiceProfile>
</rdf:RDF>
```

Fig. 3. Code Fragment of JobSubmission-Profile.daml.

```
<rdf:RDF
  <daml:Ontology about="">
    <daml:versionInfo>
      $Id: JobSubmission-Process.daml, v0.1 01/11/2002 Simone Ludwig $
    </daml:versionInfo>
    <rdfs:comment>
      DAML-S Coalition: JobSubmission Example - Process Model
<!-- Instance Definition of Job Submission -->
  <process:ProcessModel rdf:ID="JobSubmissionAgent_ProcessModel">
    <service:topLevelProcess rdf:resource="#JobSubmission_Process" />
    <service:isImplementedBy>
      <service:Service rdf:resource="&js_service;#JobSubmissionAgent"/>
    </service:isImplementedBy>
  </process:ProcessModel>
<!-- JobSubmission (ATOMIC) -->
  <rdfs:Class rdf:ID="JobSubmissionSteps">
    <rdfs:subClassOf rdf:resource="&process;#AtomicProcess" />
  </rdfs:Class>
   ...
  <rdf:Property rdf:ID="createJDL_In">
    <rdfs:subPropertyOf rdf:resource="&process;#input"/>
    <rdfs:domain rdf:resource="#JobSubmissionSteps"/>
    <rdfs:range rdf:resource="&concepts;#Job"/>
  </rdf:Property>
   ...
  <rdf:Property rdf:ID="verifyJob_In">
    <rdfs:subPropertyOf rdf:resource="&process;#input"/>
    <rdfs:domain rdf:resource="#JobSubmissionSteps"/>
    <rdfs:range rdf:resource="&concepts;#Job"/>
  </rdf:Property>
  <rdf:Property rdf:ID="updateBookkeeping_In">
    <rdfs:subPropertyOf rdf:resource="&process;#input"/>
    <rdfs:domain rdf:resource="#JobSubmissionSteps"/>
    <rdfs:range rdf:resource="&concepts;#Bookkeeping"/>
  </rdf:Property>
</rdf:RDF>
```

Fig. 4. Code Fragment of JobSubmission-Process.daml.

The JSS comprises the following steps:

1. Check certificate validity.
2. Create (JDL) Job Description Language for the job.
3. Submit the job using a net process or any more complex procedure.
4. Monitor the load via network.
5. Verify correct job execution and progress.
6. Periodically check the log files such as `stdout` and `stderr`.
7. Verify correct job termination and output.
8. Update bookkeeping of processed data.

The code fragments in Figs. 2–4 describe the implementation of the service, profile and process ontology of the Grid JSS. Fig. 2 shows the service model which describes the job submission service and refers to the service profile and the service process. The service profile, shown in Fig. 3, defines the specification of the functionalities. These are provided by the service together with a list of functional attributes which offer additional information and requirements about the service. Fig. 4 shows the process model. It describes how the service works, i.e. how the service is composed and what happens when it is executed. There are atomic, simple and composite processes which can be modeled with DAML-S. The job submission is an atomic process identified by the `& process` attribute.

4. Related work

During the past few years lots of effort and research have been placed in the field of knowledge representation and semantic matching mechanisms which are described in the following subsections. The different approaches are based on enhanced service discovery methods, description logics, the usage of ontology-based knowledge representation and Grid-based resource selection.

4.1. Bluetooth Service Discovery Protocol

The Bluetooth Service Discovery Protocol (BSDP) supports a sophisticated matching mechanism that uses semantic information associated with services and attributes to decide the success or failure of a query. The enhanced version of BSDP (presented by Saikanth Avanch et al. [3]) supports semantic matching and provides service registration. The first version used the RDF/RDF-S data model and the second and current version uses DAML.

4.2. DReggie

The project DReggie is an attempt to enhance the matching mechanisms in Jini and other service discovery systems.

The key idea in DReggie is to enable these service discovery systems to perform matching based on semantic information associated with the services [6]. Semantic service matching introduces the possibilities of fuzziness and inexactness of the response to a service discovery request. In the DReggie system, a service discovery request contains the description of an "ideal" service — one whose capabilities match exactly with the requirements. Thus, matching now involves comparison of requirements specified with the capabilities of existing services. Depending on the requirements, a match may occur even if one or more capabilities do not match exactly. Service description in DReggie system is marked up in DAML. The semantic matching process, which uses these descriptions, is performed by a reasoning engine. At the heart of DReggie is an enhanced Jini Lookup Service (JLS) that enables smart discovery of Jini-enabled services. DReggie retains the matching mechanism currently employed by the Jini lookup and the discovery infrastructure.

4.3. Description logics matchmaker

DLs are a family of knowledge representation formalism. They are based on the notion of concepts and roles, and are mainly characterized by constructors that allow complex concepts and roles to be built from atomic DLs [11] [18]. The main benefit from these knowledge languages is that sound and complete algorithms for the subsumption and satisfiability problems often exist. A DL reasoner solves the problems of equivalence, satisfiability and subsumption. The matching service provides three basic functionalities which are advertising, querying and browsing. The algorithm is a translation in DL terms of the ideas exposed, previously mentioned. The DL matchmaker achieves service discovery based on knowledge representation formalism.

4.4. Semantic search — SHOE

Simple HTML Ontology Extensions (SHOE) is an ontology-based knowledge representation language designed for the Web. The current version of the language is the result of an effort that started in 1996 and anticipated many of the features which were subsequently added to XML and RDF. SHOE uses knowledge oriented elements, and associates meaning with content by making each web page commit to one or more ontologies [12]. SHOE ontologies permit the discovery of implicit knowledge through the use of taxonomies and inference rules, allowing content providers to encode only the necessary information on their web pages, and to use the level of details that is appropriate to the context.

4.5. LARKS

LARKS [18] is used by middle or matchmaking agents to pair service-requesting agents with service-providing agents that meet the requesting agents' requirements. LARKS is expressive and capable of supporting inferences. It also incorporates application domain knowledge in agent advertisements and requests. Domain-specific knowledge is specified as local ontologies in the concept language Information Terminology Language (ITL). The LARKS matchmaking process employs techniques from information retrieval, artificial intelligence, and software engineering to compute the syntactical and semantic similarity among agent capability descriptions. The matching engine of the matchmaker agent contains five different filters for context matching, word frequency profile comparison, similarity matching, signature matching and constraint matching. The user configures these filters to achieve the desired tradeoff between performance and matching quality.

4.6. Resource selector

Tangmurarunkit et al. [19] have designed and prototyped an ontology-based resource selector that exploits ontologies, background knowledge, and rules for solving resource matching in the Grid to overcome the restrictions and constraints of resource descriptions in the Grid. Traditional resource matching, as done by the Condor Matchmaker [20] or Portable Batch System [21], matchmaking is based on symmetric, attribute-based matching. In order to make the matchmaking more flexible and also to consider the structure of VOs the framework consists of ontology-based matchmakers, resource providers and resource consumers or requesters. Resource providers periodically advertise their resources and capabilities to one or more matchmakers using advertisement messages. The user can then activate the matchmaker by submitting a query asking for resources that satisfy the request specification. The query is then processed by the TRIPLE/XSB deductive database system [22] using matchmaking rules, in combination with background knowledge and ontologies to find the best match for the request.

5. Conclusion

In a Grid environment where so many different implementations are available the need for semantic matching based on a defined ontology becomes increasingly important in order to provide close and customized service request matches. This paper introduced a semantic service discovery framework for Grid environments. It showed the new matching mechanism based on ontology knowledge. The Grid ontology for the JSS (written in DAML-S code) was presented in order to show how ontologies are defined and implemented. The proposed matching framework allows a better service discovery and close matches in a flexible way based on the defined ontology. Therefore, Grid applications are able to specify the criteria a service request should be matched with.

A link to other semantic and ontology representations such as the BSDP, DReggie, DLs, SHOE, LARKS and resource selector were accounted for. The related projects fall into the categories of enhanced service discovery methods, description logics, usage of ontology-based knowledge representation and Grid-based resource selection.

The proposed semantic service discovery framework has two close relations. The first is on the semantic side with the LARKS matchmaker, where the main concept of semantic matching was taken from; and the second is the relation to the Grid with the resource selector, where semantic matching was also introduced to Grid computing, however not for services but for resources. The framework proposed enabled a first step towards a semantic service discovery approach in a Grid environment. The main requirements which have driven the development of the semantic framework were high degree of flexibility and expressiveness, support for subsumption and datatypes and a flexible and modular structure. Special focus was given to an efficient matching mechanism which is based on semantic descriptions.

Providing semantics defined in a Grid services ontology allows a semantic service discovery matching process. Taking the semantic approach into account for service discovery shows that it allows a more flexible and interoperable way of finding the appropriate services.

References

[1] ALICE — A Large Ion Collider Experiment, http://alice.web.cern.ch/Alice/

[2] A. Ankolekar, et al., DAML-S: Semantic Markup for Web Services, International Semantic Web Workshop (SWWS), 2001.

[3] S. Anvancha, et al., Semantic Service Discovery in Bluetooth. Technical Report, Computer Science and Electrical Engineering, University of Maryland, 2001.

[4] ATLAS — A Toroidal LHC Apparatus, http://atlasinfo.cern.ch/Atlas/Welcome.html

[5] S. Bechhofer, C. Goble, Towards Annotation using DAML+OIL, K-CAP 2001 Workshop on Knowledge Markup and Semantic Annotation, Victoria B.C., 2001.

[6] D. Chakraborty, et al., DReggie: semantic service discovery for M-commerce applications, Workshop on Reliable and Secure Applications in Mobile Environment, 20th Symposium on Reliable Distributed Systems, 2001.

[7] CMS — Compact Muon Solenoid, http://cmsdoc.cern.ch/cms/outreach/html/index.shtml

[8] DAML. Darpa Agent Markup Language Program, http://www.daml.org

[9] D. Fensel, et al., OIL in a nutshell, Proceedings of EKAW-2000, LN AI, 2000.

[10] C. Goble, The Grid - From concept to reality in distributed computing, Bioinformatics World Article, 2003, http://www.bioinformaticsworld.info/feature3b.html

[11] J. Gonzalez-Castillo, et al., Description Logics for Matchmaking of Services, HP Labs Technical Report, 2001.

[12] J. Heflin, J. Hendler, Semantic Interoperability on the Web, Proceedings of Extreme Markup Languages 2000, Graphic Communications Association, 2000.

[13] LHCb — Large Hadron Collider, http://lhcb-public.web.cern.ch/lhcb-public/default.htm

[14] S.A. Ludwig, Comparison of centralized and decentralized Service Discovery in a Grid Environment, Proceedings of 15th IASTED International Conference, Parallel and Distributed Computing and Systems (PDCS), USA, 2003.

[15] S.A. Ludwig, et al., A grid service discovery matchmaker based on ontology description, Proceedings of the International EuroWeb 2002 Conference, Oxford, UK, 2002.

[16] M. Paolucci, et al., Semantic matching of web services capabilities. Proceedings International Semantic Web Conference (ISWC 02), 2002.

[17] T.R. Payne, et al., Advertising and Matching DAML-S Service Descriptions, Semantic Web Working Symposium (SWWS), 2001.

[18] K. Sycara, et al., Dynamic service matchmaking among agents in open information environments, J. ACM SIGMOD Rec. 1999.

[19] H. Tangmunarunkit, S. Decker, C. Kesselman, Ontology-based Resource Matching in the Grid — The Grid meets the Semantic Web, Proceedings of the First Workshop on Semantics in Peer-to-Peer and Grid Computing (SemPG03) in conjunction with the Twelfth International World Wide Web Conference 2003, Hungary, 2003.

[20] The Condor Project, http://www.cs.wisc.edu/condor

[21] The Portable Batch System, http://pbs.mrj.com

[22] The XSB Research Group, http://xsb.sourceforge.net

[23] W3C Working Draft, Requirements for a Web Ontology Language, http://www.w3.org/TR/webont-req/

Simone A. Ludwig is a Research Associate at Cardiff University, UK. She received a Ph.D. degree in Computer Science and an M.Sc. degree in Distributed Computing Systems Engineering from Brunel University, UK in 2004 and 2000, respectively. Her research interests include Grid Computing, Distributed Computing, Semantic Grid/Web, Knowledge Engineering and Software Development. Prior to that, she worked several years as a Software Engineer for Daimler-Chrysler, Stuttgart, Germany. She is a member of the IEEE.

S.M.S. Reyhani received his BEng (hons) degree in Electrical and Electronic Engineering from The University of Wales College of Cardiff, UK, and his MSc degree in Microelectronics systems design and Ph.D. in Computational Bio-electromagnetism from Brunel University in London, UK. His research interests include theoretical and computational bio-electromagnetism, numerical modelling techniques, signal processing, knowledge engineering and software development. He is a member of IEE and IEEE.