An ActOn-based Semantic Information Service for EGEE

Wei Xing [#1], Oscar Corcho [#2], Carole Goble [#3], Marios D. Dikaiakos [*4]

[#]*School of Computer Science, University of Manchester*
Oxford Road Manchester, UK
[1]wxing@cs.man.ac.uk
[2]ocorcho@cs.man.ac.uk [3]carole.goble@cs.man.ac.uk

[*]*Department of Computer Science*
University of Cyprus, CYPRUS
[4]mdd@cs.ucy.ac.cy

***Abstract*— We describe an information service that aggregates metadata available in hundreds of information sources of the EGEE Grid infrastructure. It uses an ontology-based information integration architecture (ActOn), which is suitable the highly dynamic distributed information sources available in Grid systems, where information changes frequently and where the information of distributed sources has to be aggregated in order to solve complex queries. These two challenges are addressed by a metadata cache that works with an update-on-demand policy and by an information source selection module that selects the most suitable source at a given point in time, respectively. We have evaluated the quality of this information service, and compared it with other similar services from the EGEE production testbed, with promising results.**

I. Introduction and Motivation

EGEE [1] provides a production quality grid infrastructure spanning more than 30 countries with over 160 sites to a myriad of applications from various scientific domains, including Earth Sciences, High Energy Physics, Bioinformatics and Astrophysics. In such a large-scale Grid system, there are thousands of heterogeneous, loosely coupled resources, services, and applications, which are distributed geographically in a wide range. The current EGEE production testbed includes over 200 sites, 35,000 CPUs, 13 Petabytes of storage space in hundreds of storage elements, and an average of 40,000 concurrent jobs per day on behalf of 100 Virtual Organisations (VOs).

Having information about those heterogeneous entities is critical for the EGEE gLite middleware [2]. This information is used for tasks such as resource discovery, workflow orchestration, meta-scheduling, and security. Such information is normally aggregated and provided by information services, which can be defined as "databases of attribute metadata about resources" [3][1] Examples of information services are BDII [4] and MDS [5], focused on hardware and software resources; and RGMA [6], focused on jobs, services and running environments.

The main limitations of existing information services are that they do not provide enough information about large-scale distributed systems like EGEE, since they only focus on a few specific aspects of such systems, and that they do not always provide accurate information about the actual status of the Grid resources that they refer to.

To overcome these two limitations, we propose the creation of an information service that aggregates information from different information services in the EGEE production testbed, using an ontology-based information integration architecture [7]. The aggregation of distributed information poses the following **challenges**, due to the dynamic and heterogeneous nature of Grids:

- Metadata of a Grid entity consists of multiple attributes, whose values can be normally obtained from heterogeneous and geographically-distributed information sources. In a large-scale Grid system, several information sources can provide the same piece of information about a resource. And it may be difficult to identify and locate the most suitable (and available) information source for a specific information need.
- Metadata about most Grid entities may be updated frequently, so as to reflect the current status (capability and availability) of the services and resources that it refers to. This makes it hard to create and maintain up-to-date metadata about all the resources available in a Grid. For instance, the usage level of a CPU, storage space, and network connection may change every few minutes.
- Different information sources or services may provide overlapping views of the Grid state, in different schemas and formats, and with different characteristics of their information provenance (update frequency, quality-related).

These challenges are addressed in our ActOn-based information service. ActOn (Active Ontology) [8] is an ontology-based information integration approach that can be used to generate and maintain up-to-date metadata for a dynamic, large-scale distributed system.

First, ActOn uses ontologies to describe the domain for which information will be aggregated. This provides an expressive model to describe that information, which can be exploited with query languages and use for validation purposes

[1]In the rest of the paper, we will use the terms information and metadata interchangeably.

(e.g., to detect inconsistencies in the aggregated information) and for deriving new information. It also provides an extensible data model where changes in the descriptions of resources and services, or in the information sources (update frequency, information quality, etc.) are automatically reflected in the behaviour of the system.

Second, ActOn incorporates two modules that are not commonly found in other ontology-based information integration architectures: a cache, which provides fast access to information that has been already integrated and materialised and which is still valid, and an information source selector, which is used during the generation of the execution plan for retrieving information from the information sources and allows the system to adapt to changing conditions of the infrastructure and to add new information services easily.

The remaining of this paper is organised as follows. Section 2 introduces related work, namely existing Grid information services. Section 3 presents the architecture of ActOn, focusing on its different knowledge and software components, and on the main interactions between them, and describing how each of them are instantiated for the implementation of our EGEE Grid information service. Section 4 gives the results of our evaluation on the information quality of our approach, and compares them with other two EGEE Grid information services. Finally, Section 5 provides conclusions, and describes open issues and our planned future work.

II. Related Work: Grid Information Services

Currently, there are several well-known and widely-used Grid information services: Monitoring and Discovery System (MDS), Berkeley DB Information Index (BDII), and RGMA [5], [4], [6]. These services are deployed in most Grid systems, such as Europe Data Grid, Crossgrid, NASA Grid, and Open Science Grid [9], [10], [1], [11], [12], and widely used by Grid middleware and applications running on them.

MDS [5] is the information service component of the Globus platform. In MDS2.x, information about Grid resources is extracted by "information providers", which are software programs that collect and organise information from individual Grid entities, either by executing local operations or by contacting third-party information sources (e.g., the Network Weather Service, SNMP, etc.). Extracted information is organised according to the LDAP (Lightweight Directory Access Protocol) data model in LDIF format and uploaded into LDAP-based servers of the Grid Resource Information Service (GRIS). GRIS servers can register themselves in the Grid Index Information Services (GIIS) in order to aggregate directories, using a soft-state registration protocol called Grid Registration Protocol (GRRP). One of the disadvantages of MDS is that it is based on the LDAP data model, which is too rigid to be adopted or to represent the heterogeneous information in/on Grids. It also lacks in the ability of supporting complex queries.

BDII [4] is an improvement of MDS [5], designed to improve its query performance. It uses the MDS information model and access API and caches information with the Berkeley DB. An update process is used to populate LDAP-based servers. It consists in obtaining LDIF, either by doing an ldapsearch on LDAP URLs or by running a local script that generates LDIF. Then the LDIF is inserted into the LDAP database. BDII has the same problems as MDS for information expression and query.

RGMA [6] is a framework that combines monitoring and information services based on a relational model, which is implemented with XML. It implements the Grid Monitoring Architecture (GMA) proposed by the Open Grid Forum. GMA models the information infrastructure of the Grid using three core types of components: (i) producers, which provide information; (ii) consumers, which request information; and (iii) a single registry, which mediates the communication between producers and consumers. RGMA implements two additional properties over GMA. First, consumers and producers handle the registry in a transparent way; thus, anyone using RGMA to supply or receive information does not need to know about the registry. And second, all the information appears as one large relational database and can be queried as such (anyway, in the current implementation, the database is centralised). RGMA can be accessed using the RGMA API. The main drawback of RGMA is that it cannot easily manage the dynamic information about time-sensitive Grid resources, due to its architecture that comprises a central registry and distributed information servlet-based information producers.

III. Active Ontology (ActOn) and the EGEE Information Service

ActOn (Active Ontology) [8] is an ontology-based information integration approach that can be used to generate and maintain up-to-date metadata for a dynamic, large-scale distributed system. In this section we will describe the main characteristics of this approach and its architecture, and will use as a running example the details of the EGEE information service that we have built with this approach.

The development of ActOn was based on a list of requirements that are based on the actual information integration needs that were identified in dynamic, distributed systems like the EGEE Grid, Crossgrid, and Unicore [1], [10], [13].

- We need to deal with frequent changes of parts of the metadata, caused by the dynamic features of the entities of a large-scale distributed system.
- We need to have an efficient and economic way to avoid a continuous metadata update process, which is expensive for a large-scale distributed system.
- We need to be able to select the most suitable information source from a set of geographically-distributed and heterogeneous ones, which provide overlapping pieces of information, in different formats, and which can be available or unavailable at a given point in time.
- We need to create/update the metadata that captures only those aspects that we are interested in.

Although these requirements arise in the context of developing an aggregated information service for the EGEE Grid infrastructure, similar requirements can be also found

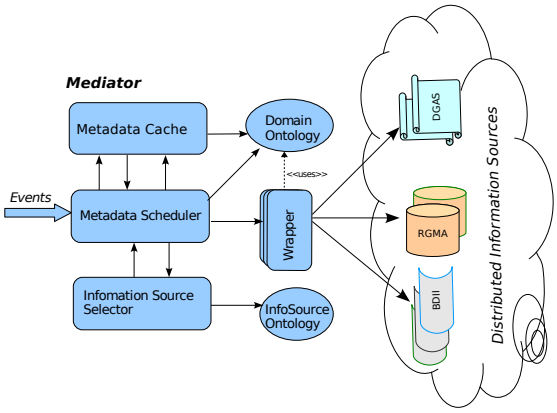

Fig. 1. Overview of the Active Ontology architecture [8]

in other application domains (e.g., the stock market, currency exchange, etc.). Therefore, ActOn provides a generic solution that can be easily adapted to different application domains.

ActOn is comprised of a set of knowledge components, which represent knowledge from the application domain and from the information sources; and software components, such as a metadata scheduler (MSch), an information source selector (ISS), a metadata cache (MC), and a set of information wrappers. Figure 1 shows how these components are interrelated and how they are related to the corresponding information sources where data is taken from.

The EGEE information service that has been developed using ActOn uses Globus Toolkit 4 (GT4) [14] and the S-OGSA Semantic Binding Service [15]. The latter is used to bind semantic metadata with the ontologies it refers to and with the resources that the metadata describes, so that metadata can be managed as a resource, with its own lifetime, authorisation policies, etc. All the source code of ActOn and of the information service that we have described is available under Open Source license at the OntoGrid CVS [16].

A. ActOn Knowledge Components

The knowledge components used in ActOn include a (set of) domain ontology(ies) and an ontology of the information sources. Domain ontologies describe the metadata information model in the form of domain concepts and properties for which instances will be generated, and restrictions about them. In our service these are resources, components, services, and applications of the EGEE Grid. The Information Source Ontology provides information about the characteristics of information sources, which are used for the information source selection process. In our service they describe information services deployed in EGEE. The two ontologies are related by means of mappings that specify which domain concepts and which of their properties can be generated by which information sources, as we will explain later.

1) Grid Domain Ontologies: These domain ontologies define the global information model used to represent metadata, hence they are completely application dependant. ActOn does not put any constraint about the language to be used to implement these ontologies, although in our current implementation we assume that ontologies are described either in RDF Schema [17] or OWL [18].

We have created OWL ontologies that describe Grid entities, resources, capabilities and the relationships among them. These ontologies are based on the one described in [19] and extend the Grid ontology described in [20], which include descriptions about virtual organisations, users, applications, middleware services, computing and storage resources, networks, and usage policies. Besides the core Grid ontology, we have different ontologies for each specific Grid infrastructure. For example, the EGEE Grid Ontology describes the EGEE infrastructure and its entities, including concepts like Computer Element, Storage Element, User Interface, Worker Node, Resource Broker, Logging and Booking Service, and Site.

2) Information Source Ontology: This ontology assists in locating suitable information sources for a specific information need. It describes the features of the information sources to be used by the system and is divided into a domain-independent part, with five classes and forty properties, and a domain-specific part that contains descriptions of the types of information sources that can be used in an application, as well as specific instances of those classes.

The most important class in the domain-independent part of the ontology is `InformationSource`, which is described with four properties:

(i) `accessAPI`: it defines the information model and the information access methods to be used. For instance, the information model of BDII is LDAP, and its accessAPI can be "ldapsearch" in C and "JNDI" in Java;
(ii) `accessPoint`: it defines the server and port names to be used to obtain the information from. For instance, the CERN BDDII server can be described as "ldap://prod-bdii.cern.ch:2170";
(iii) `belongToMiddleware`: it specifies the middleware infrastructure (e.g., EGEE) where the information service is available, since depending on the middleware type and release being used the information access methods will be different;
(iv) `withSchema`: it indicates the kind of information that an information source provides. For instance, the EGEE BDII servers use the Glue Schema.

The domain-dependent part for our service contains descriptions of the following four main EGEE information providers: BDII (with the class `BDIIIP` being used to represent distributed BDII servers), RGMA, GridICE, and Unix-scripts. All of them are subclasses of the class `InformationSource`. Besides, we have defined 36 instances of BDIIIP, 10 instances of RGMA, 5 GridICE, and 10 Unix-script.

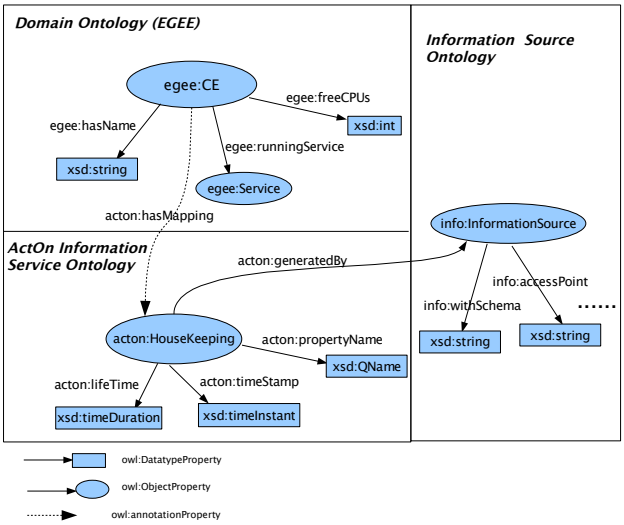

Fig. 2. Graphical overview of the association between domain and information source ontologies

An example of the information contained in one of the BDIIIP instances is:

* server name: ldap://prod-bdii.cern.ch
* server port: 2170
* access API: BDIIRet.class
* information schema: glueschema
* grid middleware: gLite middleware

As shown in Figure 2, the association between domain and information source ontologies is expressed by means of *housekeeping mappings*. Each domain ontology class or property is connected to the `HouseKeeping` class with the property `hasMapping`[2]. The property `generatedBy` represents the means to be used to extract information from the source and transform it into the domain ontology(ies) components. This is expressed with the class `InformationSource`. Each of the mappings specifies, as well, the timestamp and lifetime of the information retrieved from the information sources. This information is used by the Metadata Scheduler to control the Metadata Cache, as explained later.

B. ActOn Software Components

We will now describe the software components that comprise the ActOn architecture, as shown in Figure 1.

1) Metadata Scheduler (MSch): It is designed to apply an update-on-demand policy to cache metadata. That is, the cached metadata is not updated until it is stale when being queried, so as to avoid unnecessary updates. We adopt event-driven mechanisms to cope with that policy. We have defined three types of events that can trigger the update process, though we have only implemented the first one in our service. They are:

(i) Application-specific events. They are application-based lifetime control events. The MSch can force an update process based on specific application requirements. For instance, an external application may require to update a specific piece of metadata at a given point in time.

(ii) Query events. They are raised when metadata is being queried. As we will show below, if the metadata being queried is available in the metadata cache and valid, the information sources are not contacted. If not, then we contact them to get fresh metadata[3].

(iii) System-related events. They can cause changes of the Grid entities that the metadata refers to. A typical example is a *job-finished event*, which can cause the change of the value of the `runningJob` property of an instance of the class `JobQueue`.

The MSch acts upon receiving events. When the metadata scheduler receives a query event that involves retrieving metadata that has never been retrieved before or that is not valid since its expiry time has passed, or when it receives any of the other types of events, the metadata scheduler follows three steps: 1) it contacts the Information Source Selector to select the most suitable information source where to obtain the metadata from; 2) it retrieves the metadata from the selected sources, using the corresponding wrappers; and 3) it updates the metadata cache, assigns a time-stamp to the retrieved information and sends back the results to the requester.

An example can illustrate a typical procedure of MSch workflow. When a query event is triggered that requests metadata for the Computing Element ce101.cern.ch, the MSch will first check the time-stamp of its associated metadata, which is stored by the Metadata Cache, and compare it with its lifetime. If it is valid, then it will just give back the results. If it is out of date, then it will invoke the Information Source Selector service to select a suitable information source (i.e., one EGEE region or site BDII server) for updating the Computing Element metadata. After getting the information about a suitable information source (for example, lxb2086.cern.ch or prod-bdii.cern.ch), it invokes the corresponding Information Wrapper service to fetch the information with an ldapsearch query, and then invokes the Metadata Cache to update (refresh) the metadata by modifying the values and time-stamp of the relevant properties. At the same time the new metadata is sent back to the metadata requestor.

Our approach has clear advantages over others that update metadata on a regular time-scale basis, such as Globus MDS and gLite BDII. These systems keep updating all their metadata every 6-8 minutes. This approach is too expensive and imprecise, particularly in large-scale distributed systems. On the one hand, there are many useless updates: a lot of updated metadata is most likely not being used (queried) in hours

[2]When using OWL to implement the ontologies, we use an OWL annotation property so as not to interfere with the domain and information source knowledge representation.

[3]In the case that the latency is bigger than the update time of the information source, this will still provide out-of-date metadata, but in the rest of cases data will be always up-to-date

although it is updated every few minutes. On the other hand, some of the metadata may not be accurate in the case that the values of the metadata change more frequently than the regular update time. In fact, some of the dynamic metadata of BDII, such as freeCPU number, runningJobs or networking bandwidth, is usually incorrect as it is never updated on time.

2) Information Source Selector (ISS): The Information Source Selector (ISS) is used to find the most suitable information source from the set of available sources, which are described as instances of the Information Source Ontology. Information sources can be any system (database, file, service, etc.) that contains relevant information. In Grid systems there are many redundant and geographically-distributed information sources available. For example, over 20 region BDII servers can be used to fetch information about the EGEE Computing Elements.

The selection is based on a set of retrieval conditions, including the actual information needed (specified as a SPARQL query), and other aspects like the geographical proximity of the source. For example, in our prototype we have defined the class `ComputingElement` that represents EGEE computing elements. This class has a property `freeCPU` that is `generatedBy` the information source BDII.

Since in our ontology we have defined over 30 BDII servers (as instances of the class `BDIIIP`), the ISS service sends a query to select the most suitable one for fetching the needed value. The query is done in SPARQL, and retrieves those instances of `BDIIIP` that `belongToMiddleware` EGEE Grid, whose `schema` is GlueSchema and whose `version` is 3.0. Also the middleware is *gLite*, and the release version *3.1.5*. Below is a SPARQL query for a BDIIIP instance in our implementation:

```
PREFIX  onG: <http://www.cs.man.ac.uk/img/ontogrid/>
FROM    <EGEEGridInfo.v0.3.owl>
SELECT  ?BDIIIP
WHERE   { ?x onG:runningService bdiiip?   .
          OPTIONAL { ?x onG:belongTo "EGEE"   .
                     ?y onG:installedOn   ''gLite''   .
                     ?z onG:withSchema   ''GlueSchema'' . }
```

The selected BDIIIP instances are ranked according to their geographical proximity, quality of the service, and the capabilities of the BDII server machine.

3) Information Wrappers: After an information source is selected, the Metadata Scheduler contacts the corresponding Information Wrapper in order to retrieve the relevant up-to-date information. Normally there is an Information Wrapper per type of information source accessed (that is, one for MDS, another one for BDDII, etc.). We have developed four kinds of wrappers: the BDII server wrapper, the RGMA server wrapper, the GridICE wrapper, and the Unix-script wrapper.

The wrappers are used to fetch information from different information sources. First, the Information Wrapper gets information from the information source ontology about the data model of the specific source to be accessed, and about its access API and access point. Then it fetches the information from its source. For instance, a BDIIIP information source can be queried using an LDAP query based on the information from a BDII individual, such as "ldapsearch -x -H ldap://prod-bdii.cern.ch:2170 -b mds-vo-name=CERN-PROD,o=grid". Once the query is answered, the results are transformed into instances of the concept `ComputingElement` of the domain ontology.

ActOn does not impose any specific technology for generating Information Wrappers. They can be generated in an ad-hoc manner, by hard-coding the access to the information source and the transformation into the application domain ontology. They can be also generated with generic wrapper-generation languages and technologies, such as WSL [21], D2R [22], R2O [23], etc.

4) Metadata Cache (MC): The Metadata Cache (MC) stores and manages the metadata obtained from the information sources, together with its timestamp and lifetime information, so that it can check whether such property values are still valid or not (e.g., lifetime control) when it receives a query event that involves them.

The metadata cache uses the domain ontologies as its information model. For instance, in our service the MC caches information about Computing Elements (CE), Storage Elements (SE), Virtual Organisations (VO), etc. As commented above, the MC uses the S-OGSA semantic binding service implementation in order to store the values together with their timestamp and lifetime, using the mappings shown in Figure 2.

IV. Information Quality Evaluation and Comparison

In our evaluation we want to know whether the results provided by our service conform to the expectations of the users, and how it compares with the other available services. We are interested in knowing whether all information services obtain the same results when answering the same query, given the same conditions in the EGEE production testbed. We also want to check how many of those answers are correct and how many of the existing answers are actually retrieved. To check this, we have selected two metrics, commonly used in information retrieval: precision and recall. Below we provide their definitions and the formulae used to calculate them:

Precision: The proportion of relevant information retrieved, out of all the information retrieved.

$$\text{Precision} = \frac{(relevant\ information) \cap (retrieved\ information)}{retrieved\ information} \quad (1)$$

Recall: The proportion of relevant information that is retrieved, out of all the relevant information available.

$$\text{Recall} = \frac{(relevant\ information) \cap (retrieved\ information)}{relevant\ information} \quad (2)$$

A. Experiment setup and design

We have designed a set of experiments for measuring the information quality criteria selected. Measurements are taken on a real Grid testbed, the EGEE production testbed, which at the time of the experiments, has gLite 3.0.1 installed as its middleware. The user interfaces used to access the EGEE Grid

are the UI machines at the University of Manchester[4], United Kingdom, and at the Institute of Physics of Belgrade[5], Serbia.

To carry out the experiments and record their results, we have developed a set of Java-based client software and Unix shell scripts, available at the IST OntoGrid project CVS [16].

The key aspects upon which we compare different information services are: i) the information model that each information service adopts; and ii) the expressiveness of its query language. In order to evaluate these two features, we have proposed six representative queries that cover a wide range of Grid systems, including Grid hardware resources, software resources, middleware environment, services, applications, etc., and show increasing complexity. These queries can be normally issued by middleware systems like schedulers, resource brokers or by more complex applications:

- Query 1: Find all the Computing Elements (CEs) that support the BIOMED Virtual Organisation (VO).
- Query 2: Find all the CEs that support the BIOMED VO and have more than 100 CPUs available.
- Query 3: Find all the CEs that support the MPI running environment.
- Query 4: Find all the CEs that support the BIOMED VO, have more than 100 CPUs available, and support the MPI running environment.
- Query 5: Find all the CEs where GATE (Geant4 Application for Tomographic Emission) can be run.
- Query 6: Find all the CEs that support the BIOMED VO, have more than 100 CPUs available, and where GATE can be run.

Information Service	**Query1** (Find all the CEs that support the BIOMED VO)
BDII (LDAP search)	ldapsearch -x -H ldap://lcg-bdii.cern.ch:2170 -b mds-vo name=local,o=grid '(&(objectClass=GlueVOView) (GlueVOViewLocalID=biomed))' GlueCEAccessControlBaseRule
RGMA (SQL query)	Select GlueCEVOViewUniqueID, Value from GlueCEVOViewAccessControlBaseRule WHERE Value='VO:biomed'
ActOn Based (SPARQL query)	PREFIX egeeOnto: <http://www.cs.man.ac.uk/img/ontogrid#> SELECT ?ceid ?ceID ?VO WHERE ?ceid egeeOnto:CEUniqueID ?ceID . ?ceid egeeOnto:hasVO ?VO . OPTIONAL { ?ceid egeeOnto:VO ?ceID . FILTER (?vo = ``biomed")}

Fig. 3. An Example of the Query 1 in BDII, RGMA, and ActON

Each of these six queries has been translated into the query languages of the three information services. Figure 3 shows an example for Query 1. We use different clients to execute these queries and extract the results obtained (e.g., ldapsearch for BDII, the gLite RGMA client tools for RGMA and a Java-based ActOn client for the ActOn-based information service).

Not only queries are different, but also query results are obtained in different manners, due to the differences in the information models of each service. The result of a BDII query is a set of LDAP entries, of an RGMA query a set of table rows, and of an ActOn-based query a set of RDF triples. Figure 4 shows three different ways to show the same Grid resource in the three services evaluated (i.e., ce02.tier2.hep.manchester.ac.uk, an EGEE Computing Element). Even if they have different syntax and size, in our experiment we count them as one piece of information each. That is, we use each "Grid resource" obtained from a query as the basic unit for counting information, which will be used to calculate precision and recall, as described in Section IV-B.

Query results of BDII:

```
# biomed, ce02.tier2.hep.manchester.ac.uk:2119/jobmanager-lcgpbs-biomed, UKI-NORTHGRID-MAN-HEP, local, grid
dn: GlueVOViewLocalID=biomed,GlueCEUniqueID=ce02.tier2.hep.manchester.ac.uk:2119/jobmanager-lcgpbs-
biomed,mds-vo-name=UKI-NORTHGRID-MAN-HEP,mds-vo-name=local,o=grid
GlueCEAccessControlBaseRule: VO:biomed
```

Query results of RGMA:

```
+-----------------------------------------------------------------------------------+
| GlueCEVOViewUniqueID                                                  | Value     |
+-----------------------------------------------------------------------------------+
|ce02.tier2.hep.manchester.ac.uk :2119/jobmanager-lcgpbs-biomed/biomed  | VO:biomed |
+-----------------------------------------------------------------------------------+
```

Query results of ActOn:

```
| ceid                                          | ceID                                | VO       |
| <http://img.cs.man.ac.uk/ontogrid1234423456> | "ce02.tier2.hep.manchester.ac.uk"  | "biomed" |
```

Fig. 4. Results of BDII, RGMA, and ActOn for the the same Grid resource Computing Element at University of Manchester (ce02.manchester.ac.uk)

B. Experimental Results Measurement and Analysis

The experiment consists in examining the information retrieved for each of the six queries aforementioned, so as to estimate their corresponding precision and recall measures.

Precision is easy to determine, since it can be computed manually by looking at the results obtained from each query. In all cases, we assume binary relevancy of information, that is, each piece of information retrieved is either relevant or irrelevant for the issued query.

Recall is more difficult to determine, due to the fact that the amount of information available in the EGEE production testbed changes frequently in these systems and there is no way to get accurate information about the actual state of the Grid resources that are available without using the information services that we are evaluating. To get a good approximation, we execute each query 100 times, with a 4-minute interval between executions, monitoring the testbed during 400 minutes. Then we use the highest value obtained from this 100 executions as the total number of relevant information to be used to calculate recall.

Tables I, II and III provide the precision and recall measurements obtained after the execution of the experiments described above for the three information services selected: BDII, RGMA and the ActOn-based information service. The

[4]ui.tier2.hep.manchester.ac.uk
[5]ce.phy.bg.ac.yu

values provided in the tables show the average of executing the queries 100 times.

TABLE I
BDII RECALL & PRECISION MEASUREMENT (100 TIMES)

Query	*Retrieved Info.*	*Relevant Info.*	*Precision*	Recall
Q1	14,999	15,200	1	0.987
Q2	242,517	19,708	0.082	0.918
Q3	7174	7300	1	0.983
Q4	485034	4600	0.010	0.990
Q5	-	-	-	-
Q6	-	-	-	-

TABLE II
RGMA RECALL & PRECISION MEASUREMENT (100 TIMES)

Query	*Retrieved Info.*	*Relevant Info.*	*Precision*	Recall
Q1	3417	15200	1	0.225
Q2	6321	6321	1	1
Q3	6568	7300	1	0.900
Q4	11245	4914	0.437	0.563
Q5	-	-	-	-
Q6	-	-	-	-

TABLE III
ACTON RECALL & PRECISION MEASUREMENT (100 TIMES)

Query	*Retrieved Info.*	*Relevant Info.*	*Precision*	Recall
Q1	15200	15200	1	1
Q2	34100	34100	1	1
Q3	6568	7300	1	0.900
Q4	6568	7300	1	0.900
Q5	24	24	1	0.900
Q6	6	6	1	1

As a general comment about these results, we can highlight the fact that BDII shows in general poor results with respect to recall and precision, while ActOn and RGMA present better results. This is mainly related to the repository that BDII uses (LDAP), which is too lightweight and hence provides weak information process and query capabilities; while RGMA's is based on relational databases and ActOn's is based on RDF, which both have better query capabilities.

Now we will analyse with more detail some of the system behaviours over specific queries, and derive more conclusions from these values:

BDII has weak query capabilities. Table I shows that BDII has extremely bad precision results for queries 2 and 4, while the results for queries 1 and 3 are excellent. This is related to its weak query ability, as aforementioned. LDAP-based queries are string-based, and hence they cannot be used to support queries over numerical values, such as "greater than or lower than". If we want to improve this precision value, we need to fetch all the information about CE CPUs as a string value first (as we have done to get these results), and then post-process (filter) those results on the client side. RGMA and the ActOn-based information services do not have that problem, since their query abilities are better.

RGMA is not able to relate information available in different tables. Table II shows that RGMA has bad precision results in query 4. RGMA contains information to solve this query, but the information comes from two different tables (`GlueCE` and `GlueSubClusterSoftwareRunTimeEnvironment`), and the query language used by RGMA does not allow making a join of both tables. Hence the situation is similar to the previous case: this problem can be solved on the client side by post-processing the results that have been obtained from each separate query.

RGMA is very sensitive to the registering and availability of information providers at a given point in time. Table II shows that RGMA has bad recall results in query 1. This is because the amount of Computing Element producers that is available during the experiment is not always stable, due to the fact that either producers were not registered in the RGMA registry at that specific moment, or that the producers were not configured correctly or available at that point in time. BDII and the ActOn-based information service are more robust to this, due to the fact that they store information locally and do not depend on their information providers at the time of querying.

Some complex queries cannot be answered by one type of information service in isolation. Tables I and II show that BDII and RGMA can only answer the first four queries. They cannot answer queries 5 and 6 because their information providers cannot provide enough information and should be combined. This shows that the ability of BDII and RGMA to share their data resources is weak. On the other hand, the ActOn-based information service has the ability to adopt existing information sources as its information providers, and aggregate information from these information sources to answer such complex queries.

V. CONCLUSIONS AND FUTURE WORK

In this paper we have presented an information service for EGEE that is based on an ontology-based information integration approach, Active Ontology (ActOn). This approach overcomes some of the limitations of current similar approaches when dealing with highly dynamic, distributed and redundant information sources in the cases where information quality, availability and robustness, as well as response time, are important non-functional requirements.

We adopt a data warehouse approach to information integration, where we materialise relevant information from different information sources and assign it a lifetime based on the update frequency of the information sources where it is taken from. The materialised information acts as a metadata cache that is updated only when an information request is sent to the system and the materialised information has expired.

Besides, information sources are selected at run-time from a large set of sources that provide redundant information, based on criteria such as their information coverage, availability, geographical proximity, etc.

The results of the experiments executed to analyse the quality of metadata and the response time of our system are promising, suggesting that it can increase the metadata quality

and robustness of currently-deployed information systems, and decrease the cost of system resources.

In summary, our main contribution over the state of the art in Grid information systems is that we have proposed a Grid information service that performs an ontology-based integration of information from existing services, what allows creating automatically execution plans for retrieving information from sources that are overlapping in the information that they publish and have different provenance constraints, and maintain a cache of relevant information as long as it is valid given its lifetime constraints.

As for the integration of features from other systems, we plan to work on the integration and extension of (semi-)automatic wrapper generation systems like D2R and R2O (currently these systems are only available to access databases, but we plan to extend them for accessing information services such as those present in Grid systems), and on the integration of query reformulation and planning techniques, such as those of Theseus [24], with the metadata cache approach that we have proposed.

We also plan to take full advantage of following an ontology-based approach for information integration, allowing us to perform tasks that cannot be done easily with the services currently available, such as detecting inconsistencies in the metadata that is available or deriving new information. For example, a common problem with current information services is their level of trustiness. There are many cases where a computing element specifies that it gives support to MPI but does not comply with the requirements for running an MPI job, which are that it must be a CE server, must have an `sshd` service running on it, must have the libraries `mpirun` and `libmpi.so` in its file system, and must have at least two worker nodes. Similarly, we could derive that a computing element gives support to MPI if the previous conditions apply, since this is a necessary and sufficient condition.

ACKNOWLEDGEMENTS

This work is supported by the EU FP6 OntoGrid project (STREP 511513), by the Marie Curie fellowship RSSGRID (FP6-2002-Mobility-5-006668), and by the EU FP6 CoreGrid Network of Excellence (FP6-004265). We also thank Pinar Alper (IMG group), Antun Balaz and Laurence Field (EGEE), and Georges Da Costa and Anastasios Gounaris (CoreGrid WP2), for their helpful comments.

REFERENCES

[1] "Enabling Grids for E-sciencE (EGEE)," http://public.eu-egee.org/.
[2] "EGEE gLite," http://glite.web.cern.ch/glite.
[3] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, and J. V. Reich, *The Open Grid Services Architecture, Version 1.5*, gfd-i.080 ed., GGF, July 2006, http://forge.gridforum.org/projects/ogsa-wg.
[4] "Berkeley Database Information Index (BDII)," http://lfield.home.cern.ch/lfield/cgi-bin/wiki.cgi?area=bdiipage=documentation.
[5] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman, "Grid information services for distributed resource sharing," in *Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10)*. IEEE Press, August 2001.
[6] "EDG RGMA," www.marianne.in2p3.fr/datagrid/documentation/rgma-guide.pdf.
[7] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner, "Ontology-based integration of information — a survey of existing approaches," in *IJCAI–01 Workshop: Ontologies and Information Sharing*, H. Stuckenschmidt, Ed., 2001, pp. 108–117.
[8] W. Xing, O. Corcho, C. Goble, and M. Dikaiakos, "Active Ontology: An Information Integration Approach for Highly Dynamic Information Sources," in *Europe Semantic Web Conference 2007 (ESWC-2007)*, Innsbruck, Austria, June 2007, Poster.
[9] "European DataGrid," http://eu-datagrid.web.cern.ch/eu-datagrid/.
[10] J. Marco and et al., "First Prototype of the Crossgrid Testbed," in *Proceedings of First European AcrossGrids Conference (AXGrids 2003), LNCS 2970*. Santiago de Compostela, Spain: Springer-Verlag, 2003, pp. 67–77.
[11] "Globus Toolkit," http://www.globus.org/toolkit/.
[12] "The Spallation Neutron Source (SNS) project," http://www.sns.gov/.
[13] P. Wieder and D. Mallmann, "UniGrids - Uniform Interface to Grid Services," in *7th HLRS Metacomputing and Grid Workshop*, Stuttgart, Germany, April 2004.
[14] "Globus toolkit," http://www.globus.org.
[15] Ó. Corcho, P. Alper, I. Kotsiopoulos, P. Missier, S. Bechhofer, and C. A. Goble, "An Overview of S-OGSA: A Reference Semantic Grid Architecture," *Journal of Web Semantics*, vol. 4, no. 2, pp. 102–115, 2006.
[16] "OntoGrid CVS," http://www.ontogrid.net/ontogrid/downloads.jsp.
[17] D. Brickley and R. G. (editors), "RDF Vocabulary Description Language 1.0: RDF Schema." February 2004, http://www.w3.org/TR/rdf-schema/.
[18] P. Patel-Schneider, P. Hayes, and I. Horrocks, *OWL Web Ontology Language Semantics and Abstract Syntax*, World Wide Web Consortium, February 2004.
[19] W. Xing, M. D. Dikaiakos, and R. Sakellariou, "A Core Grid Ontology for the Semantic Grid," in *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2006)*. Singapore: IEEE Computer Society, May 2006, pp. 178–184.
[20] M. Parkin, S. van den Burghe, O. Corcho, D. Snelling, and J. Brooke, "The Knowledge of the Grid: A Grid Ontology," in *Proceedings of the 6th Cracow Grid Workshop*, Cracow, Poland, October 2006.
[21] H. Garcia-Molina, Y. Papakonstantinou, A. R. a. D. Quass, Y.Sagiv, J. Ullman, V. Vassalos, and J. Widom, "The TSIMMIS Approach to Mediation: Data Models and Languages," *Intelligent Information Systems*, vol. 8, no. 2, pp. 117–132, 1997.
[22] C. Bizer, "D2R MAP: A DB to RDF Mapping Language," in *12th International World Wide Web Conference*, Budapest, May 2003.
[23] J. Barrasa, O. Corcho, and A. Gomez-Perez, "R2O, an Extensible and Semantically based Database-to-Ontology Mapping Language," in *In Proceedings of the 2nd Workshop on Semantic Web and Databases(SWDB2004)*, Toronto, Canada, 2004.
[24] G. Barish, D. DiPasquo, C. A. Knoblock, and S. Minton, "Dataflow plan execution for software agents," in *Proceedings of the Fourth International Conference on Autonomous Agents*, C. Sierra, M. Gini, and J. S. Rosenschein, Eds. Barcelona, Spain: ACM Press, 2000, pp. 138–139.