Ninth IEEE International Symposium on Parallel and Distributed Processing with Applications

Extending the Globus Information Service with the Common Information Model

I. Díaz, G. Fernández, P. González, M.J. Martín, J. Touriño
Computer Architecture Group
Department of Electronics and Systems, University of A Coruña, Spain
Email: {idiaz, gfernandez, pglez, mariam, juan}@udc.es

***Abstract*—The need of task-adapted and complete information for the management of resources is a well known issue in Grid computing. Globus Toolkit 4 (GT4) includes the Monitoring and Discovery System component (MDS4) to carry out resource management. The Common Information Model (CIM) provides a standard conceptual view of the managed environment. This work improves the MDS4 functionality through the use of CIM, with the aim of providing a unified, standard representation of the Grid resources. Since a practical CIM model may contain a large volume of information, a new Index Service that represents the CIM information through Java instances is presented. In addition, a solution that keeps data in persistent storage has also been implemented. The evaluation of the proposed solutions achieves encouraging results, with an important reduction in memory consumption, a good scalability when the number of instances increases, and with a reasonable response time.**

I. Introduction

The management and monitoring of both static and dynamic resources is a key issue in Grid environments. Grid infrastructures group heterogeneous, geographically distributed computing resources. This inherent heterogeneity and the convenience of interoperability between different Grid middleware solutions motivate the use of a common representation to allow the exchange of resources information both inside and across Grids.

The use of information models allows an abstract, structured representation of the software and hardware aspects of Grid resources. Information models are computational abstractions of the real world. Generally, they do not require any particular instrumentation and are used for data exchange through different domains. Among the standard information models, the *Common Information Model* (*CIM*) [1], defined by the *DMTF* (*Distributed Management Task Force*), is an object-oriented, user-extensible model that provides a common representation for systems, networks, applications and services management. In addition, it is supported by a set of DMTF standards that facilitate its use. Among these standards, *WBEM* (*Web-Based Enterprise Management*) [2] defines a series of specifications to create and manage the CIM information.

Globus Toolkit 4 (GT4) [3] includes the *Monitoring and Discovery System* component (*MDS4*) to carry out resource management. However, while MDS4 defines an interoperable mechanism to manage heterogeneous resources in a uniform way, the information that is subscribed into it is quite limited. Another drawback of MDS4 is that it presents limitations in querying flexibility. This is due to the use of XPath 1.0 which does not support data type queries and it is inflexible in compound queries due to its lack of predicate control directives. For example, there is no simple way to make a query for a specific service where the resource attribute falls in a specified typed range, such as a float value or an enumeration.

Lately many efforts have been invested in improving the information subscribed to the Index Service of GT4 by increasing the types of information sources [4]. The last versions of the toolkit have included a series of Information Providers for the most popular monitoring and management tools (Ganglia, Nagios, Condor, etc.). In this way, the management information about the GT4 services like GRAM or RFT is completed with information coming from several sources. Many of the Information Providers included in MDS4 use the GLUE (Grid Laboratory for a Uniform Environment) Schema [5] as a common representation for their data. GLUE is an information model that represents the Grid or cluster resources for their access and it is integrated with Globus Toolkit since version 2.

The main focus of this work is the extension of MDS4 through the integration of a complete information model like CIM. Combining MDS4 with the CIM model improves resource management, allowing more expressive queries. CIM is also extensible, so new management domains can be added in a seamless fashion. Its standard nature also means that it can be understood and manipulated by tools from many different vendors.

In a previous work [6] a preliminary CIM-based query service for resource management information in Grid systems was described. This work completes the integration of the CIM model with the GT4 middleware, with the improvement of its scalability and functionality through the use of Java instances instead of representing the CIM information in XML format. Additionally, a solution that maintains the CIM data in persistent storage is implemented to be used in case of memory consumption constraints.

The structure of the paper is as follows: Section II briefly describes the Common Information Model (CIM) and the Monitoring and Discovery System (MDS4). Section III details the proposed extension of the GT4 Information Service for its integration with the CIM model. Section IV presents the evaluation of the implemented strategies in terms of memory consumption, scalability and response time. Section V covers related work. Finally, conclusions and future work are discussed

978-0-7695-4428-1/11 $26.00 © 2011 IEEE
DOI 10.1109/ISPA.2011.62

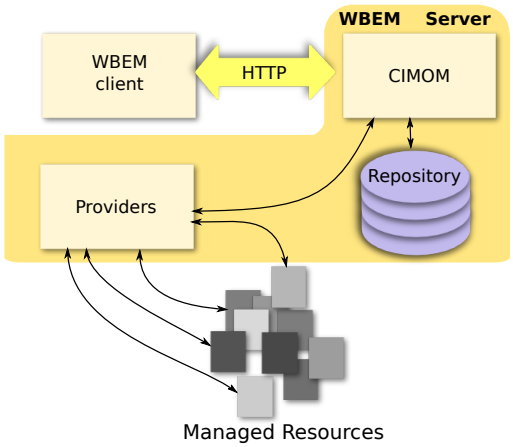

Figure 1: WBEM instrumentation

in Section VI.

II. Background

A. Common Information Model

The *Common Information Model* (*CIM*) is a management information model which provides a conceptual framework for describing management data. CIM makes it possible to exchange and interpret semantically rich management information by different systems and applications. It unifies and extends existing instrumentation and management standards (SNMP, DMI, CMPI, etc.) following the object-oriented paradigm. CIM is not bound to a particular technology or implementation. It is composed of a Schema and a Specification. The CIM Schema includes models for systems, networks, devices and applications amongst others. It enables developers of different platforms to describe data in a standard format so that it can be shared by a variety of applications. The CIM Specification is the language and methodology for describing management data.

The CIM model is the basis for several DMTF standards. Among these, WBEM defines a set of technologies to unify the management of distributed computing environments. WBEM provides the ability to exchange CIM information in an interoperable and efficient manner, defining protocols, query languages, discovery mechanisms, mappings, and anything else needed to exchange information. One of the specifications included in WBEM is *CIM-XML*, which defines a serialization of the CIM data in XML format.

WBEM adopts the client-server paradigm, as portrayed in Figure 1. The WBEM client does not have direct access to the managed resources. Instead, it sends requests to the CIM Object Manager (CIMOM), using CIM over HTTP. The CIMOM is the central component of the WBEM server and handles all communications with the client. It delegates requests to the appropriate providers and returns responses. Providers act as plugins for the CIMOM. They are responsible for the actual implementation of the management operations for a managed resource. Therefore, providers are implementation-specific. The repository is the part of the WBEM server that stores the definitions of the CIM Schemas.

There exists a wide range of WBEM implementations such as OpenPegasus [7], OpenWBEM [8], SBLIM [9] or WMI [10], which lead to widely used management tools. Other CIM-based tools and frameworks can be used as information sources, such as the AdCIM framework (`http://adcim.des.udc.es`), partly described in [11]. AdCIM is a model driven, CIM-based platform which can extract, interrelate, represent and persist the configuration ad management data of distributed systems.

B. Monitoring and Discovery System

The *Monitoring and Discovery System* (*MDS4*) component [12] is the information service included in GT4. It is in charge of gathering, publishing and accessing management information about resources and services on the Grid. As most of the components of GT4, MDS4 is Web Services-based, using standard interfaces defined within the *WS-Resource Framework* (*WSRF*) [13] and *WS-Notification* (*WSN*) [14] families of specifications to provide query and subscription interfaces to manipulate resource data in XML format. MDS4 provides standard protocols for information access and delivery, and relies on standard schemas for information representation. MDS4 interfaces with different local information sources, translating their diverse schemas into an appropriate XML schema. Various tools and applications can be constructed that take advantage of the uniform Web Services query, subscription, and notification interfaces to access those information sources that MDS4 implements.

The *Index Service* is the component of MDS4 which publishes management information of the elements in a *Virtual Organization* (*VO*) and provides a unified location for querying it. It is similar to a UDDI registry [15], except that it not only stores the location of data providers, but also caches data using a lifetime management mechanism. GT4 provides a simple default implementation of this Index Service, which obtains the management information from predefined information sources and maintains it in memory available for querying. This solution lacks scalability when dealing with large amounts of data, even more having in mind that the functionality of the MDS4 system should not impede other computing tasks. Thus, we propose the use of a persistent storage backend to store CIM data and avoid memory restrictions.

III. CIM Information Service

This section details the design and implementation of a solution to improve the performance of the GT4 Information Service through the use of CIM information. The development of the *CIM Information Service* can be divided in the following components:

- *CIM Information Provider*: new information provider developed to provide CIM data to MDS4.
- *CIM Index Service*: a customized Index Service backend which stores the available information in an external relational database.

- *CIM Operation Provider*: new operation provider which exposes new and more complex query operations to the CIM Index Service.

Figure 2 shows a general schema of the Information Service modified to interoperate with CIM. The added components are shown in dark blue. The *CIM Information Provider* collects the management information about the resources from a CIM model-based information source, such as the AdCIM framework or WBEM-based systems, and provides this information to the *CIM Index Service*. The clients can query the CIM data to the *CIM Index Service* through the operations exposed by the *CIM Operation Provider*.

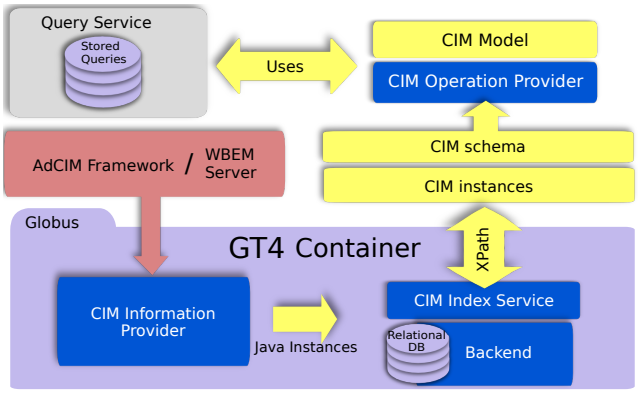

Figure 2: Overview of the new CIM Information Service extending MDS4

A. CIM Information Provider

The first aspect to tackle is the development of an information provider capable of querying information sources and making CIM data available inside MDS4. GT4 provides the following mechanisms to create a new information provider:

- The *Aggregator Sources*, which are defined inside of the *Aggregator Framework*, collect information from various sources and provide it to consumers called *Aggregator Sinks* (e.g. Index Service). MDS4 includes the following Aggregator Sources:
  - *Query Aggregator Source*: polls WSRF services for resource property information.
  - *Subscription Aggregator Source*: collects data from WSRF services via the subscription/notification operations of WS-Notification.
  - *Execution Aggregator Source*: executes scripts periodically to collect information in XML format.
- The *UsefulRP* subsystem or *RPProvider Framework* [16] included in recent versions of MDS4 provides a framework for the information gathering through multiple sources and the automatic transformation of the obtained data in an MDS4 readable format. UsefulRP provides mechanisms for both publishing WSRF properties as XML data, and to represent XML data from arbitrary sources as WSRF properties in Java instances. Though by default data

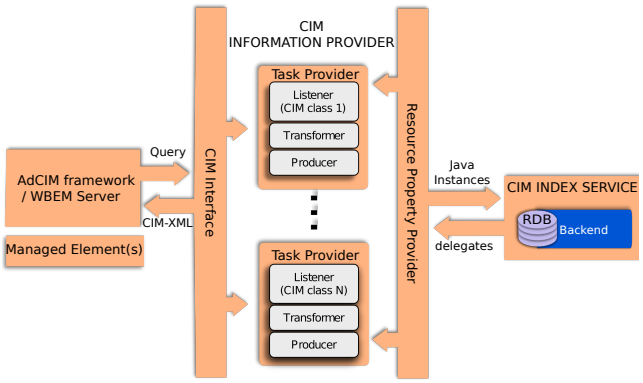

Figure 3: General Schema of CIM Information Provider

are transformed using the GLUE 1.1 schema, UsefulRP supports customized XML Schemas to introduce new information models (as will be seen later). This mapping can be triggered by special events, or scheduled periodically. XML input data can be injected from plain files, HTTP GET operations, or by the spawning of a child process (as the Execution Aggregator does). This new functionality simplifies the development of customized Information Providers.

Since there is no direct Globus support for the CIM information model, the management information in CIM is provided by non-WSRF Querying APIs. The choice is thus limited to use either the Execution Aggregator Source or the UsefulRP subsystem. The former only provides valid XML documents to Aggregator Sinks, whereas the latter can transform these XML data into Java objects, which can be managed in a programmatic way. Therefore, UsefulRP was used to implement the CIM Information Provider due to its extensive functionality. In this manner we have improved our previous work [6], where an Execution Aggregator Source was used for the CIM data subscription.

Due to the novelty of the UsefulRP system and the comparative lack of documentation, our implementation of the CIM information provider follows the Ganglia information provider [17] as a rough guide. This information provider is part of GT4; it collects information from sources encoded with the XML mapping of the GLUE Schema [5].

Figure 3 shows a diagram of the proposed information provider. Inside it, several components assume different responsibilities:

- The *CIM Interface* is in charge of obtaining the information from external sources. In the Ganglia implementation, this functionality is integrated into the producer (explained below). In our case, due to the differences among the querying APIs of several CIM-based query services, this functionality is kept separated. Thus, the information provider is decoupled from the concrete CIM information source used.
- The *Producers* serve as mediators between clients and

the information provider, decoupling them. They allow to transform the data obtained from the client into the format required by both listeners and transformers. GT4 provides the `ResourcePropertyElementProducer` interface for this purpose, which returns a Java DOM representation of an XML Element class (`org.w3c.dom.Element`). Due to the memory requirements imposed by the DOM tree representation, an information producer (`WBEMElementProducer`) is defined for each CIM class to obtain its instances. This limits the size of the received XML documents and avoids performance bottlenecks.

- The *Transformers* perform the transformation of CIM-XML into instances of Java object classes defined inside GT4. GT4 defines the `ResourcePropertyElementTransform` interface for this functionality. It is implemented through the `CIMInstanceElementTransform`, which has to be indicated in the provider's configuration. The transformation is made transparently by the UsefulRP framework if the XML document is defined with an XML Schema. Additionally, these components can apply an initial XSLT transformation to the XML document. We have defined such an XML Schema based on the specification for the representation of CIM in XML [18]. This Schema represents CIM instances and their relations, including properties and references, and involving some modifications needed for the representation of persistent entities such as the insertion of extra fields. This schema supports strongly typed queries which classify entities in an inheritance tree and allows class membership checks, not supported by the default MDS4 mechanism. We have also defined an XSLT stylesheet to remove redundancy and normalize data.
- The *Listeners* are notified periodically of the information from the sources to be updated in MDS4. This component is represented by the `ResourcePropertyProviderListener` interface. The implementation provided by GT4 maintains the subscripted information in one object attribute and defines methods to query it. Each listener receives the CIM information from a producer and delegates its storage in the CIM Index Service.
- The *Task Providers* link each listener with a producer and invoke them periodically. They are implemented by the `ResourcePropertyProviderTask` class, which also implements the `TimerListener` interface. We have defined our own task provider (`CIMRPProviderTask`), which provides producers with the necessary formatted parameters to obtain management information and also notifies the corresponding listener of new CIM instances.
- The *Resource Property Provider* is a front-end class that defines an API to access the information obtained by the information provider. It keeps a list of listeners that manage this information. The GT4 implementation is the `BaseResourcePropertyProvider` class. We have defined a CIM specialization with the `CIMResourceProperty` class, which access the available information using the backend component (described in Section III-B).

B. CIM Index Service

As stated before, the MDS4 Index Service collects information from Grid resources and publishes it in a single location. The default GT4 implementation of the Index Service (`DefaultIndexService`) stores the collected data in memory. Maintaining this information in memory entails both performance and scalability problems when large amounts of CIM data are subscribed.

The MDS4 Index Service is based on the *WSRF* specifications. It is represented as a *WS-ServiceGroup* in which each entry represents a subscription from an information source. The default implementation of the Index Service adapts its behavior according to the information source: it manages the storage of the information provided by the Aggregator Sources, but it delegates to the UsefulRP providers the access to the information that they gather.

In order to provide a single access point to the maintained data, we have defined a new backend for the Index Service that follows the *Data Access Object* design pattern. This component is in charge of managing the access to the stored data, comprising from its storage and update to its query. The proposed Index Service delegates data storage to this backend component.

Since the use of CIM representation entails a large volume of data, two implementations of the backend component have been developed: one of them stores the information in a database and the other maintains the information in memory, like the Globus default solution does. *Java Persistent API* (*JPA*) [19], a Sun Microsystems specification, was used for the database access. JPA describes an object/relational mapping API to manage relational data in Java applications. More specifically, we have used the Apache implementation of the standard, *OpenJPA* [20]. This choice has been motivated by the integration of Globus Toolkit with this implementation since version 4.2, and because OpenJPA provides configurable support for a broad range of databases.

C. CIM Operation Provider

The concept of *Operation Providers* was introduced in GT3 as a delegation-based approximation to add functionality to services. Unlike inheritance, it allows operation reuse by many services without code modifications. Java WS Core (the GT4 implementation of the WSRF and the WS-Notification family of standards) also supports this functionality. In GT3 operation providers had to implement a specific interface. In Java WS Core no such interface is required. In fact, an operation provider is not different in any way from a standard web service. That means that any web service can be used as an operation provider. There is a series of operation providers defined inside GT4 that are used broadly by different services of the middleware (`GetRPProvider`, `DestroyProvider`, `ServiceGroupRegistrationProvider`, etc.).

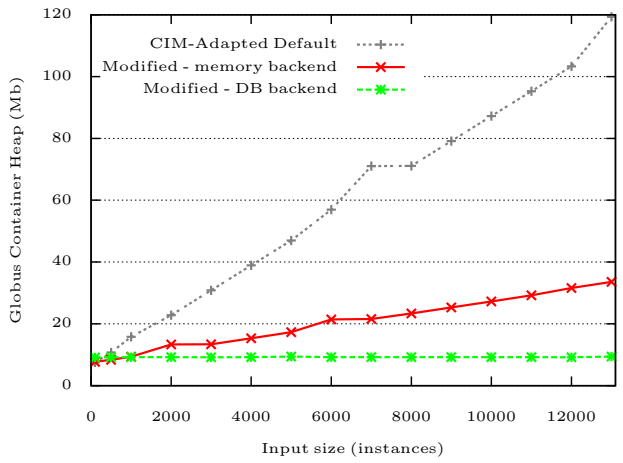

Figure 4: Experimental results: memory usage

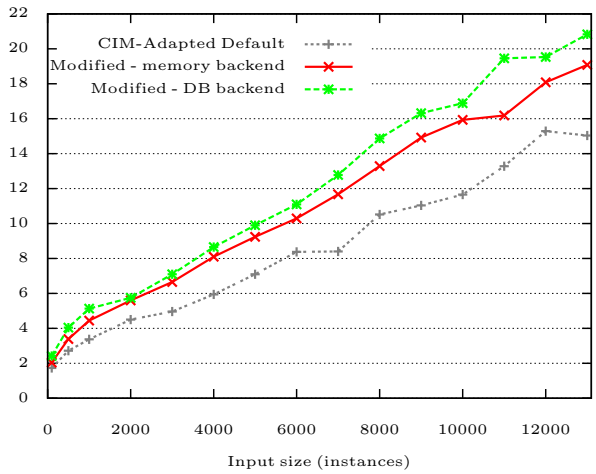

Figure 5: Experimental results: response time

To improve the integration of the CIM model with MDS4, a new operation provider was developed, the *CIM Operation Provider*. It allows to query CIM information subscribed in the CIM Index Service in a simpler and more efficient way than the default GT4 one.

The *CIM Operation Provider* incorporates the following operations, which can be used by the clients to query available CIM instances:

- Obtaining all instances of a particular CIM class.
- Obtaining all instances of a particular CIM class filtered by some property values.
- Obtaining all instances of a particular CIM relation with certain values in some of its references, so CIM associations can be navigated.

IV. Experimental Results

Some experiments were carried out to show the scalability and efficiency of the proposed CIM Information Service. The experiments were run on a Grid in which each node had a 3GHz Intel Core Duo E8400 and 2GB RAM, with Linux kernel 2.6.27 under the Ubuntu 8.10 (Intrepid) distribution. Globus Toolkit 4.2.1 and SBLIM, a WBEM implementation for Linux systems, were also used.

To ensure that the measures were taken in all cases under the same conditions, the Globus container was executed always with the minimum number of services and without performing unnecessary background tasks. Three different scenarios were considered:

- CIM-Adapted Default Index Service: In this case, the CIM information is obtained and queried using the default mechanisms provided by GT4. The data subscription is carried out with the *Execution Aggregator Source*, which executes scripts periodically and provides the resulting XML data to the Index Service. The querying of this information is done with the *wsrf-query* client, which uses the predefined `QueryResourceProperties-Provider` operation provider to make XPath queries over the information available in the Index Service. This scenario represents the approach of our previous work [6].
- Modified Index Service with memory backend: This service obtains the information from our *CIM Information Provider* and transforms it via the UsefulRP framework. The information queries are carried out with the *cim-query* client, which was implemented to query the information to the *CIM Operation Provider*.
- Modified Index Service with database backend: The container is executed with the implemented *CIM Index Service*. The difference with the previous scenario is that in this case the information is maintained in a database through the use of JPA.

The first experiment measures the memory consumption in each scenario. The memory consumed is measured as the size of the heap used by the Globus container. Figure 4 shows the results when the number of CIM instances increases. It can be observed that in the default solution (scenario 1) memory consumption increases sharply with the number of CIM instances. This can lead to a bottleneck in Grid systems in which the information is organized in a hierarchical form and nodes may have information from several others, collecting too many data. A meaningful improvement is achieved by the solutions that use the UsefulRP providers (scenarios 2 and 3). This is because the information is represented through Java instances instead of through an XML DOM tree. Therefore, in case of memory consumption constraints, any of these two solutions would perform well. The final solution, maintaining the CIM information either in memory or in a database, will be chosen conditioned to the requirements imposed by each concrete environment.

Another experiment was carried out to measure the penalties in the response time to query CIM information caused by the use of a database instead of maintaining data in memory. The obtained results are depicted in Figure 5. These results show that the response time to query the information increases when the CIM Information Provider is used to handle the CIM

information, especially when the information is maintained in a database. This is because both the search of the instances with certain values in their attributes and the querying of the database (with the consequent creation of the corresponding instances) cause performance penalties compared with the DOM tree access made by the `QueryResourcePropertiesProvider` operation provider of GT4.

This provider uses the *XPath 1.0 API* included in *Xalan-Java* to implement the XPath queries over the available information maintained in memory. This implementation of the API uses the *DTM* (*Document Table Model*) interface, which was designed specifically to optimize performance and minimize storage of the XPath and XSLT implementations. Nevertheless, the usage of XPath 1.0 means that values are not strongly typed, and operations such as searching value ranges in a data type are difficult to implement. Conversely, since UsefulRP uses an XML Schema to perform the transformation to instances, this representation based on Java objects allows typed queries with more complexity and functionality than XPath queries.

Summarizing, the use of the proposed CIM Information Service offers improved query semantics at the expense of slightly larger delays in queries. The tests have also shown a significant improvement in memory consumption.

V. Related Work

As Grid technology is increasingly being adopted, so it grows the interest of its integration with resource models (such as CIM) used in traditional IT environments. Several Working Groups of the Open Grid Forum (OGF) [21] point CIM as an interesting model for resource management and aim to promote the adoption of CIM as a resource model. Currently, the DMTF is starting to work closely with the Grid community through an alliance with the OGF [22]. Among the benefits of this alliance is the collaboration on the creation of a broader CIM model for grids that leverages the requirements of both the DMTF and the OGF Working Groups, resulting in a unified CIM model for industry-wide use.

However, currently the majority of the production Grid projects use some forms of the GLUE schema instead of CIM for the description of grid resources (e.g. EGEE [23], TeraGrid [24], OSG [25] ...). This is because they rely on Grid middleware that operates with this schema, such as Globus or gLite. One exception is the NAREGI project [26], that uses its own middleware (NAREGI Middleware) that supports an extension of the CIM schema.

Recently a new revision of GLUE, GLUE 2.0 [27] has been proposed as a recommended standard. Though it defines an abstract information model tailored to Grid systems, it does not offer mechanisms to complement this information with that of other sources. The recommendation also explicitly denies conceptual support for extensions. These drawbacks are addressed by the CIM model, and hence by our solution.

In recent years other efforts have been made to take advantage of the potential of the CIM model within the management of Grid systems. Some examples are the RMCS system [28] of Mao et al., the work presented by Ravelomanana et al. [29], or the schema presented by Tursunova et al. [30], in which applications for Grid systems using CIM were developed. Agarwala et al. [31] show a scalable approach for CIM information querying in large scale systems, not only grids. However, these approaches work independently of the management service provided by the Grid middleware, so they must be used separately.

An approach to include the CIM information in the Index Service of Globus is the proposal by Wang et al. [32], which implements a transformation from CIM format to the GLUE Schema, and, in a later work [33], also transform CIM data from VMWare virtual machines into GLUE to integrate their resource information into a Grid infrastructure. The same idea is followed by Nakada et al. [34] to achieve interoperability among different grid middleware that use different resource information schemas. This solution is quite limited since occasionally a correspondence between both models cannot be found, as the CIM model covers much more scope than the GLUE schema and allows its extension with definitions that represent a specific domain. The work of Andreetto et al. [35] adapts the gLite Grid middleware to support GLUE, BSE/JDSL and SAML.

With regard to the persistent storage of management information, Aloisio et al. [36] have developed, within the European GridLab project, a Grid information service (the *iGrid Information Service*) based on the MDS of GT2 that maintains the management information in a database as our proposal. While the prerequisites of performance and scalability are achieved, this system was designed for GT2, which has a very different architecture and uses different technologies than GT4.

VI. Conclusions and Future Work

The solution proposed in this paper makes it possible the use of information obtained from WBEM applications and CIM model-based information sources, such as our own AdCIM framework, for the management of Grids. The CIM Schema provides structures and models that make it possible to represent all the computational aspects of a Grid with a high level of detail. The developed implementation takes advantage of the facilities and standards provided by GT4.

A series of components for MDS4 which optimize the use of CIM inside GT4 have been presented. A new information provider was developed using the UsefulRP framework included in MDS4. From the experimental results it can be concluded that this framework is valuable in the development of new information providers, obtaining a good trade-off between memory consumption and response time. To improve the performance and scalability of the system, a new Index Service with persistent storage has been also implemented, which performs in a scalable manner when dealing with large amounts of CIM data.

As for future work, the focus on efficiency and scalability must be emphasized, since the management component of the Grid must not create significant overhead. With this goal in mind, further studies about possible optimizations in the Globus

integration will be done. A possible alternative to the solution proposed in this paper would be managing the information in XML format, storing it in a native XML database. In this case it would be necessary to study both approaches, considering both the optimization of the storage and the performance of the querying of information.

Recently, a new version of Globus has been released, the Globus Toolkit 5. MDS has not been included in this new version as there are plans to replace it by a new Crux-based IIS (Integrated Information Service). The adaptation of this work to the new IIS solution will be studied when implementation details are known.

ACKNOWLEDGEMENTS

This work was funded by the Ministry of Science and Innovation of Spain under Projects TIN2007-67537-C03 and TIN2010-16375.

REFERENCES

[1] DMTF Architecture Working Group. Common Information Model (CIM) Infrastructure. [Online]. Available: http://www.dmtf.org/standards/published_documents/DSP0004_2.5.0.pdf

[2] WBEM solutions. [Online]. Available: http://www.wbemsolutions.com/

[3] I. Foster, "Globus Toolkit Version 4: Software for service-oriented systems," *Journal of Computer Science and Technology*, vol. 21, no. 4, pp. 513–520, July 2006.

[4] Globus Alliance. Globus Toolkit 4.2.1 WS MDS information providers. [Online]. Available: http://www.globus.org/toolkit/docs/latest-stable/info/providers/

[5] Globus Alliance. Globus Toolkit 4.0 WS MDS: Cluster monitoring information and the GLUE resource property. [Online]. Available: http://www.globus.org/toolkit/docs/4.0/info/key/gluerp.html

[6] I. Díaz, G. Fernández, M. J. Martín, P. González, and J. Touriño, "Integrating the Common Information Model with MDS4," in *Proceedings of the 9th IEEE/ACM International Conference on Grid Computing*, Tsukuba, Japan, October 2008, pp. 298–303.

[7] The Open Group. OpenPegasus, C++ CIM/WBEM manageability services broker. [Online]. Available: http://www.openpegasus.org/

[8] OpenWBEM. [Online]. Available: http://www.openwbem.org/

[9] Standards Based Linux Instrumentation for Manageability. [Online]. Available: http://sourceforge.net/apps/mediawiki/sblim/

[10] C. Tunstall and G. Cole, *Developing WMI solutions: A guide to Windows Management Instrumentation*, ser. Microsoft Technology. Addison-Wesley Professional, December 2002.

[11] I. Díaz, J. Touriño, J. Salceda, and R. Doallo, "A framework focus on configuration modeling and integration with transparent persistence," in *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, IPDPS 2005, Workshop on System Management Tools for Large-Scale Parallel Systems*, Denver, USA, April 2005, p. 297a.

[12] J. Schopf, I. Raicu, L. Pearlman, N. Miller, C. Kesselman, I. Foster, and M. D'Arcy, "Monitoring and discovery in a Web Services framework: Functionality and performance of Globus Toolkit MDS4," Argonne National Laboratory, MCS Preprint 1315-0106, January 2006.

[13] T. Banks, "Web Services Resource Framework (WSRF) - primer v1.2," OASIS Tech. Rep., May 2006. [Online]. Available: http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf

[14] OASIS. Web Services Base Notification 1.3. [Online]. Available: http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf

[15] Online Community for the Universal Description, Discovery, and Integration OASIS Standard. [Online]. Available: http://uddi.xml.org/

[16] Globus Alliance. Globus Toolkit 4.2.1 WS MDS UsefulRP guides. [Online]. Available: http://www.globus.org/toolkit/docs/4.2/4.2.1/info/usefulrp/usefulrp.pdf

[17] M. Massie, B. Chun, and D. Culler, "The Ganglia distributed monitoring system: Design, implementation, and experience," *Parallel Computing*, vol. 30, no. 5-6, pp. 817–840, June 2004.

[18] Distributed Management Task Force, Inc. Specification for the representation of CIM in XML. [Online]. Available: http://www.dmtf.org/standards/published_documents/DSP0201_2.3.0.pdf

[19] R. Biswas and E. Ort, "The Java persistence API - a simpler programming model for entity persistence," Sun Microsystems, Inc., May 2006. [Online]. Available: http://java.sun.com/developer/technicalArticles/J2EE/jpa/

[20] Apache OpenJPA user's guide. [Online]. Available: http://openjpa.apache.org/documentation.html

[21] Open Grid Forum. [Online]. Available: http://www.gridforum.org/

[22] OGF/DMTF Work Register Version 1.2, May 2009. [Online]. Available: http://www.dmtf.org/about/register/OGF-DMTFWorkRegister1_2.pdf

[23] Enabling Grids for E-sciencE. [Online]. Available: http://www.eu-egee.org/

[24] The TeraGrid project. [Online]. Available: http://www.teragrid.org/

[25] Open Science Grid. [Online]. Available: http://www.opensciencegrid.org/

[26] National Research Grid Initiative. [Online]. Available: http://www.naregi.org/index_e.html

[27] GLUE specification v. 2.0. [Online]. Available: http://www.ogf.org/documents/GFD.147.pdf

[28] H. Mao, L. Huang, and M. Li, "Web resource monitoring based on Common Information Model," in *Proceedings of the 1st Asia-Pacific Services Computing Conference, APSCC 2006*, December 2006, pp. 520–525.

[29] S. Ravelomanana, S. Bianchi, C. Joumaa, and M. Sibilla, "A contextual GRID monitoring by a model driven approach," in *Proceedings of the 3rd Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services, AICT/ICIW 2006*, Guadeloupe, French Caribbean, February 2006, pp. 37–43.

[30] S. Tursunova, T. Son, and Y. Kim, "Grid resource management with tightly coupled WBEM/CIM local management," in *Proceedings of the 11th IEEE/IFIP Network Operations and Management Symposium, NOMS'2008*, Salvador da Bahia, Brazil, April 2008, pp. 983–986.

[31] S. Agarwala, L. Bathen, D. Jadav, and R. Routray, "Configuration discovery and monitoring middleware for enterprise datacenters," in *Proceedings of the 12th IEEE/IFIP Network Operations and Management Symposium, NOMS'2010*, Osaka, Japan, April 2010, pp. 639–646.

[32] L. Wang, M. Kunze, and J. Tao, "From CIM to GLUE: Translate resource information of virtual machines to computational Grids," in *GI/ITG Kommunikation und Verteilte Systeme, Fachgespräch "Virtualisierung"*. Universität Paderborn, February 2008, pp. 55–63.

[33] L. Wang, G. von Laszewski, D. Chen, J. Tao, and M. Kunze, "Provide virtual machine information for Grid computing," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40, no. 6, pp. 1362–1374, November 2010.

[34] H. Nakada, K. Saga, Y. Saeki, H. Sato, M. Hatanaka, and S. Matsuoka, "Job invocation interoperability between NAREGI middleware beta and gLite," in *Proceedings of the 9th IEEE International Conference on High Performance Computing in Asia Pacific Region*, Seoul, Korea, October 2008, pp. 298–303.

[35] P. Andreetto, S. Andreozzi, A. Ghiselli, M. Marzolla, V. Venturi, and L. Zangrando, "Standards-based job management in Grid systems," *Journal of Grid Computing*, vol. 8, pp. 19–45, 2010.

[36] G. Aloisio, M. Cafaro, I. Epicoco, S. Fiore, D. Lezzi, M. Mirto, and S. Mocavero, "iGrid, a novel Grid information service," in *Proceedings of the European Grid Conference, EGC 2005*, Amsterdam, The Netherlands, February 2005, pp. 506–515.