A replicated information system to enable dynamic collaborations in the Grid

E. Torres*,†, G. Moltó, D. Segrelles, I. Blanquer, and V. Hernández

Institut d’Instrumentació per a Imatge Molecular. Universitat Politècnica de València, Camino de Vera S/N, 46022 Valencia Spain

SUMMARY

The main advantage of Grid computing over other distributed computing paradigms is its capability to coordinate the access to data and resources in a virtual multi-institutional environment. To this end, the information system plays a decisive role in selecting the services that meet the applications needs. This paper presents an information system for the Grid that provides transparent and scalable group communication services to standard Grid applications, with the objective of supporting dynamic collaborations that could help address problems that involve only some participants of a Virtual Organization. In particular, it enables more flexible delivery mechanisms, which allows applications to select the appropriate services before sending their data to the information system. This significantly enhances the protection of data from unauthorized access, and avoids the transmission of unnecessary messages over the network. The proposed information system is based on the use of XML technologies and replication. It introduces several new advanced features that are not currently supported as a whole by any Grid middleware, such as: several entry points to the information, persistent capabilities, support for advanced queries based on XQuery and support for the industrial standard WS-Policy. The information system has been stress-tested under realistic workloads in a Grid infrastructure with 50 sites. Scalability has been evaluated in up to 1000 messages that can be up to 10 Kbytes in size each, updated with a frequency of 5 minutes. Copyright © 2011 John Wiley & Sons, Ltd.

Received …

KEY WORDS: Grid; Information systems; Service-oriented architectures; Replication

1. INTRODUCTION

Computational Grids are currently used to tackle major challenges in scientific research. For example, the Worldwide LHC Computing Grid (WLCG) [1] distributes and analyses hundreds of terabytes of data, serving a community of several thousands of users around the world. To this end, different tools have been developed, enabling users to execute applications in ubiquitous distributed computational resources, and providing transparent and secure access to distributed data. gLite [2], Globus Toolkit [3] and UNICORE [4] are examples of open-source Grid middlewares that are widely used in various fields of science and engineering.

The main advantage of Grid over other distributed computing paradigms is its capability to coordinate the access to data and resources across different administrative domains, in a virtual multi-institutional environment. Grid supports Virtual Organizations (VO) created to address the problems of a particular area. For example, the Spanish National Grid Initiative (ES-NGI) [5] provides a virtual, distributed computing infrastructure that uses Grid technologies to interconnect
approximately 20 computational resource centers in Spain. A special characteristic for the ES-NGI is a strong collaborative and cooperative link with the Portuguese NGI through the IBERGRID agreement. This infrastructure is connected to the European Grid Infrastructure (EGI) [6]. The IBERGRID VO for Life Science applications (life.vo.ibergrid.eu) is an example of VO that has been created in the context of EGI to support the research activity in the life science community. This VO enables international collaboration in medical imaging, bioinformatics and drug discovery.

However, dynamic collaborations present a challenge to Grid information systems, since they do not provide a simple way to define special access rights agreed by part of the members of a VO. For example, this problem occurs when two or more groups of the VO agree to collaborate on the achievement of a particular objective. Although this can be tackled through the use of thematic VO subgroups, this approach has the disadvantage of requiring the approval of an administrator of the VO, and requires the reconfiguration of the infrastructure to support differentiation among groups. Another possible approach could be to adapt the applications to the collaboration, but this is not effective, since the management of Grid applications is a complex and tedious task. This especially affects the effectiveness of dynamic collaborations. Because of their dynamic nature, they have limited lifetime, in contrast to permanent collaborations. This problem is exacerbated in secure environments, for example hospitals or enterprise, where there exist additional requirements with respect to privacy or confidentiality. A possible solution could be to provide Grid services with the means for creating dynamic groups on the basis of the need of several partners in accomplishing one or more specific tasks. To achieve this goal, additional research is needed in the field of information management for the Grid.

In Grid, the information system is used by the applications to select the services that meet their needs. For example, the information is used in the allocation of computational resources for the execution of jobs, and to select the storage elements that contain the necessary data for the job and where the results will be written.

The use of hierarchical models is generally accepted as the solution to share this information in the Grid. The Monitoring and Discovery System (MDS) [7] is an example of information system for the Grid that provides access to the information organized in a tree hierarchy. MDS defines a hierarchical structure of nodes, where the information can follow two directions: (1) from the nodes to the root, and (2) from the root to the nodes.

In this model, if a node fails then the hierarchical information starting from this node is lost, since no replication capabilities are available. This issue has been addressed by the authors in [8], which presents a new topology with two entry points to the information system.

Other example of hierarchical system is the Berkeley Database Information Index (BDII) [9], used in EGI to provide information about the resources and their status. BDII is based on MDS. Several BDII servers are deployed at EGI forming a hierarchical topology of three levels: top level BDII, Grid Index Information Services (GIISs) and Grid Resource Information Services (GRISs). At the top level, BDII collects information from the site BDII or GIIS. Each Grid site runs a GIIS that aggregates information from the local GRISs, which are resource-level BDII. On each resource, a GRIS publishes dynamic and static information about that resource.

Although this topology consists of multiple services, only the top level BDII have access to the information of the entire system, and therefore, clients have to query a top level BDII in order to find the information that they require. To cope with this bottleneck, there can be multiple instances of the top level BDII.

In addition to BDII, the Relational Grid Monitoring Architecture (R-GMA) [10] is a different information system that has been used for a limited number of applications in EGEE [11], the precursor to EGI. Two of the most important are the accounting of the resources, and the user-level monitoring. The main advantage of R-GMA over BDII is in the use of the relational model, whereas BDII relies on the LDAP data model [12]. In R-GMA, information is presented as though it were in a global distributed relational database (although there are some differences) [13]. This model supports more advanced query operations than the BDII does, and it is also easier to modify the schema in R-GMA, making it more suitable for user-level information. On the other hand, this
flexibility of query reduces the performance of R-GMA, and at the same time makes it more complex and hard for administration.

Therefore, the majority of the information that describes resources and jobs in the Grid is provided by hierarchical systems, principally MDS and BDII. However, this model has several limitations related to the lack of flexibility in the types of relationships that can be modeled with this approach. In particular, the rigid structure of the hierarchical relations between the nodes in the MDS makes their dynamic reorganization difficult. For example, it is very difficult to define a relation $1:N$ where a node sends information to a subset of nodes of the MDS. This kind of relations is very important to facilitate communication between dynamic groups and other types of collaborations.

This paper presents a novel information system for the Grid that provides group communication services to standard Grid applications, transparently and in scalable manner. To this end, the proposed approach enables applications to select the information services to which the information is intended to be delivered. This brings two main advantages: (1) from the viewpoint of security, the protection of data from unauthorized access can be significantly enhanced by the use of trusted information services, and (2) from the viewpoint of the performance, it avoids the transmission of unnecessary messages over the network. According to our knowledge, these features are not present in the currently available Grid middleware.

A practical example could be a set of different centers, which hold different parts of the data. For instance, project CVIMO [14] has provided an e-infrastructure for health research that uses a Grid approach to securely share oncology medical images among different hospitals. In this context, physicians of different hospitals collaborate, improving diagnosis and treatment. To this end, dynamic collaborations are created between two or more hospitals, with the objective of analyzing complex cases. In this collaboration, the hospital responsible for the treatment of the patient will share the necessary information with the hospitals that participate in the collaboration. In the same way, the hospitals will share their findings with the responsible hospital. This must be performed in a secure context, ensuring that the data remained confidential.

In this way, the responsible hospital will store the information of the patient on an encrypted storage placed on a trusted domain, and it will use the information system for distributing decryption keys and other sensitive data among the participants of the collaboration. This approach guarantees that only the users of the organizations that receive the decryption keys through the information system will access the data, in contrast to traditional approaches, where any user with a valid proxy certificate is allowed to access the data.

This use case describes a scenario where the information system is used to implement a catalog service that enables Grid services to publish their capabilities (both static and dynamic attributes, such as security and performance) and applications to specify their requirements. This approach aims at facilitating the creation of dynamic groups within the VO, to enable more flexible collaboration among institutions in autonomous Grid systems.

Additionally, the information system presented in this paper supports WS-Policy [15] with the objective of ensuring interoperability with several major web service frameworks that also support this standard, such as Apache Axis2 [16] and .NET Framework [17]. The proposed information system also provides persistent storage the information, which is not present on other systems, such as the MDS of the Globus Toolkit.

The remainder of this paper is structured as follows. First, section 2 presents an information service for the Grid that provides applications with scalable access to collections of XML documents distributed across the members of a collaboration group, using a native XML representation with support for advanced queries. Next, section 3 covers a case study that reveals the benefits and limitations of using this information service. Finally, section 4 summarizes the paper and points out to future work.

2. ARCHITECTURE OF THE INFORMATION SYSTEM

This section presents the architecture of the information system that has been designed to provide group communication services to standard Grid applications, transparently and in scalable manner.
The architecture heavily relies on the use of XML technologies and replication. In particular, it provides access to a collection of XML documents globally distributed over the Grid, and supports advanced queries based on XQuery. This approach aims at offering similar flexibility to the relational model, without losing the performance.

2.1. Data Model and Support for Groups

This section describes the data model that supports the information system, which plays an essential role in the efficient transportation, storage and management of information in the Grid. The tasks required for assembling these processes into Grid applications include transporting the data from its origin to its destination, and storing, searching and transforming the information. The most widely used information systems in production Grid infrastructures (e.g. MDS, BDII) use distributed storages based on XML for this purpose.

Searches and write operations are the most challenging aspects of managing Grid information, while transportation relies heavily on replication. The use of an intermediate data representation based on LDAP is generally accepted as the solution to facilitate performance. However, searches in this approach are limited to the attributes defined in the LDAP directory schema for the computational resources and the storage elements, which are normally defined using the GLUE Schema [18]. On the other hand, these services avoid disk access, storing the data in-memory. This approach has the disadvantage that when a new element joins the system (e.g. after a server crash), it receives a complete copy of the information from its parent in the tree hierarchy, causing a large amount of traffic in the network.

In contrast, native XML databases (XMLDBs) efficiently support the storage and retrieval of XML documents, preserving the structure of the data in the XML document. For this reason, there is no need to explicitly map the information to other data structure (although some implementations make this mapping implicitly), thus reducing the overhead of transforming data from and to XML.

XMLDBs implement most of the mechanisms commonly found in other database systems, such as indexes, transactions or logging, providing support for a wide variety of data access models. In particular, they have been found to perform well at handling collections of documents with complex data structures and to reduce the computational overhead [19, 20]. Additionally, XMLDBs supports XQuery, a language that is used for querying the XML data.

Figure 1 describes the architecture of the information system. The figure shows sites A, B, C, D, E and F that authorize access to the members of a given VO. Sites A, B, C, D and E have created a collaboration, in which F is not involved. The figure also shows the mechanism that uses A to communicate with the participants of the collaboration, which consists on a hierarchical tree structure with A at its root that distributes information from root to leaves.

At the global level, the information is stored in collections of XML documents distributed over the set of replicas of the information service. Each replica manages a local XMLDB that is updated...
with the information sent from the applications to the information system. Collections contain WS-Policy documents, which provide the basis for searches and other operations.

WS-Policy is a World Wide Web Consortium (W3C) recommendation that provides a mechanism to share policies among different administrative domains. It provides a general purpose model and syntax to describe and communicate the policies of a Web service. This specification allows service providers to advertise their policies, and service consumers to specify their requirements.

Several different policies that can be applied to the interaction of a client with a Web service have been standardized (e.g. WS-Security, WS-Reliability), and others can be customized depending on the needs of each application. For example, in Grids, clients can use policies to express their requirements, constraints and preferences, and organizations to advertise the capabilities that support their resources. Currently, the systems that perform the match-making of requirements to allocate the most appropriate resources for the execution of jobs in the Grid, such as gLite WMS [21] and GridWay [22], implements their own policy interface and protocols, which difficult the interoperability and portability of Grid applications.

In order to enforce policies, systems need to provide the underlying mechanisms that process and perform the actions needed to implement a policy at runtime. For this purpose, the next section proposes a framework that uses XQuery to filter the relevant policies from a collection of generic WS-Policy documents stored in a XMLDB. The rest of the process consists on matching client policies to server policies.

The objective of focusing the data model on WS-Policy containers is to decouple it from the implementation. The storing of WS-Policy documents in the database makes the standardization of common tasks possible, which involve database maintenance and replication. On the other hand, the custom XML documents included in these WS-Policy documents can be accessed using different approaches.

The set of replicas sharing a particular collection defines a group in the information system. For example, the collaboration shown in Figure 1 is created when an application sends information that is exclusively delivered to the replicas A, B, C, D and E, but not to F. To this end, the information system provides an API that allows applications to create groups. Each replica that receives information from a group creates a collection to store and manage the information of that group. The replicas support search operations using their local XMLDBs. Additionally, the information system supports insert, update and delete operations that are executed locally at one replica and replicated over the service.

2.2. Support for Advanced Searches using XQuery and WS-Policy

The information system uses an approach that combines XQuery and WS-Policy to manage collections of XML documents of various types. Figure 2 shows a diagram that describes how the information is mapped in the database, using the example of the jobExecutionGuarantee definition that allows applications to negotiate parameterized service guarantees for the execution of jobs in the Grid [23]. Each collection in the database defines the group for the WS-Policy documents that stores. Similarly, the WS-Policy document defines the subject for the information. For example, the subject can be a computational resource, a storage element, an application or a user, or a combination of them. The collection defines the membership of the subject to a group.
WS-Policy documents can be divided into blocks to include several different types of XML documents. Each XML document that is contained in a block of the WS-Policy document describes a specific attribute of the subject, while each XML element corresponds to a particular detail of the information, such as the response time of a computational resource. This model provides more flexible searching options than the LDAP model.

For example, all the information that describes the capabilities (e.g., security, performance, status) of a computational resource or storage element in the Grid can be expressed through different, specialized XML documents, and they can be included in a single WS-Policy document. The information system uses a different copy of the WS-Policy document for each group that involves the subject. The XMLDB internally handle groups through collections of XML documents. Thus, in the particular case of dynamic collaborations where each collection represents a specific collaboration and the documents stored in the collection represent the subjects involved in the collaboration, the same resource may participate in different collaborations, having different capabilities at different groups (because it can use different WS-Policy documents to advertise its capabilities).

Figure 3 describes the flow of information within the information system. The left side of the figure shows a service or application that creates one or more XML documents with the objective of describing the capabilities of a resource. The structure of each document is specified in an XML schema. These documents are packaged in a WS-Policy document to store them in a collection.

The same part of the figure shows a client that describes a series of requirements using several XML documents, and it stores them in a WS-Policy document. The information system supports search operations based on this information.

Figure 4 shows an example of query that uses XQuery. The query shown in the figure selects all the XML documents from a specific collection that match the requirements of the client. To this end, the search selects the WS-Policy documents that include at least one XML document that fits the XML schema on which the requirement is defined (jobExecutionGuarantee). In this way, the search is limited to the subjects of the group that define all the attributes requested by the client. Additionally, the example shows how to specify the parameters for attributes matching certain criteria. For instance, the query shown in the figure specifies a maximum value for the response time (responseInMilliseconds) and minimum value for availability (availabilityInPercent).

```xml
for $item in collection('container_group1.xml')/wsp:Policy/
  wsp:ExactlyOne/wsp:All/qos:jobExecutionGuarantee
where $item/qos:endpoint/qos:localpart = $localpart
and xs:long($item/qos:responseInMilliseconds)
  <= xs:long($responseInMilliseconds)
and xs:float($item/qos:availabilityInPercent)
  >= xs:float($availabilityInPercent)
return $item
```

Advertise (service) Example A1

<wsf:ExactlyOne>
  <wsf:All>
    <qos:jobExecutionGuarantee>
      <qos:responseInMilliseconds>200</qos:responseInMilliseconds>
    </qos:jobExecutionGuarantee>
    <qos:availabilityInPercent>98.0</qos:availabilityInPercent>
  </wsf:All>
</wsf:ExactlyOne>

Requirements (client) Example R1

<wsf:ExactlyOne>
  <wsf:All>
    <qos:jobExecutionGuarantee>
      <qos:responseInMilliseconds>500</qos:responseInMilliseconds>
      <qos:availabilityInPercent>90.0</qos:availabilityInPercent>
    </qos:jobExecutionGuarantee>
  </wsf:All>
</wsf:ExactlyOne>

Advertise (service) Example A2

<wsf:ExactlyOne>
  <wsf:All>
    <qos:jobExecutionGuarantee>
      <qos:responseInMilliseconds>300</qos:responseInMilliseconds>
    </qos:jobExecutionGuarantee>
    <qos:availabilityInPercent>90.0</qos:availabilityInPercent>
  </wsf:All>
</wsf:ExactlyOne>

Requirements (client) Example R2

<wsf:ExactlyOne>
  <wsf:All>
    <qos:jobExecutionGuarantee>
      <qos:responseInMilliseconds>400</qos:responseInMilliseconds>
    </qos:jobExecutionGuarantee>
  </wsf:All>
</wsf:ExactlyOne>

Figure 5. Output examples based on the query shown in Figure 4 with different input documents.

This kind of query allows the information system to extract XML documents directly from the database, in contrast to extract WS-Policy documents from the database and then to extract XML documents from them. In particular, they are useful to handle pieces of information that identifies the subject. For example, in this particular case the endpoint element identifies the resource to which the information corresponds. Figure 5 shows several output examples based on the query shown in Figure 4 with different input documents. This approach has the advantage of simplicity, since the output documents do not need additional processing, and they can be used directly with other services (e.g. Grid meta-scheduling services).

Additionally, the information system supports searches based on the WS-Policy framework. In particular, the intersect operation provides developers with the means for comparing two different policies and finding the alternatives that are supported on both policies. For example, the result of intersecting the document that advertises the capabilities of the resource in Figure 5 with the document that specifies the requirements of the client coincides with the result shown in the figure.

This approach allows overriding the implementation of the intersect operation in the information system to adapt it to particular cases. For example, this is useful to implement a catalog to provide efficient access to information structured in a schema that is known beforehand (e.g. GLUE Schema).

On the other hand, in the case of using WS-Policy to search the database, it is necessary to extract the documents from the database before executing the query. For example, this could be done by using a variation of the XQuery expression shown in Figure 4 to select WS-Policy documents instead of XML documents. However, this will increase the overhead of XML processing, which could have a negative effect on the performance of the system.

2.3. Communications in the Information System

Figure 1 shows how the components of the information system cooperate to storage and update information in the Grid. Each replica of the service consists of a Grid service linked to a local XMLDB. Section 3 presents a case study of the information system that is based on the Web Services Resource Framework (WSRF) [24].

Copyright © 2011 John Wiley & Sons, Ltd.
Prepared using cpeauth.cls DOI: 10.1002/cpe
The Globus Toolkit 4 (GT4) [25] is an open source software toolkit used for building Grids that conform to the Open Grid Services Architecture (OGSA) [26]. OGSA describes an architecture for service-oriented Grid computing that is based on Web service technologies.

WSRF is a widely used infrastructure for OGSA, which is completely implemented in GT4. It provides support for modeling and deploying stateful resources using Web services. A stateful resource maintains its state between method calls. However, the Web service standard explicitly defines that Web services should be stateless. For the provision of the state, WSRF defines the WS-Resource, a combination of a Web service and a data container that stores the stateful data, while keeping the conformance to the Web service standard.

Authors of [25] provide more details regarding the development of Grid services with GT4. As the development framework is quite complex, there is an initiative underway to port GT4 services on modern Web service development tools [27].

In particular, the implementation of the case study of section 3 uses a replicated Grid service that groups several WS-Resources to manage the database. Each WS-Resource handles the collection of documents of a specific collaboration. The replicated service updates the documents every time a change is made to the information of the collaboration. To this end, it select the collection that corresponds to the group and, when necessary, the WS-Policy document that corresponds to the subject, to execute operations.

The operations are transmitted over the network using the most effective mean. For example, updates are transmitted over the network as XQuery sentences to avoid sending the complete WS-Policy document, with the objective of saving bandwidth.

The information system updates at regular intervals. Write operations that apply to a single replica are blocked by the information system until the next scheduled replication event. Every replica that participates in a group can modify the information of any subject of the group.

In particular, the WSRF-based implementation uses the replication library introduced by authors of [28] to propagate the updates to the information system. This library provides WSRF-based Grid services with automatic and transparent replication of WS-Resources. It operates in a multi-primary passive replication scheme, in which any replica can process a request and the state of the modified WS-Resource is propagated to the other replicas. Additionally, it supports a replication mechanism that uses a hierarchical structure to organize the communications in the Grid. This approach has been successfully used previously in MDS and BDII.

The replication library can use a ring-based topology, as depicted in the left side of Figure 6, or a Leaf-to-Root Complete Binary Tree topology, as depicted in the right side of the figure. A Complete Binary Tree is a binary tree in which every level, except possibly the last, is filled, and the nodes in the last level are all to the left. This topology allows concurrent parallel update streams through the branches of the tree. Since a Complete Binary Tree is always perfectly balanced, this state update process requires a time in the order of $\Theta(\log_2(N))$ for a tree with $N$ replicas, thus resulting in a significantly faster approach than using a ring-based topology, for a moderate number of replicas.

Figure 6. Interaction diagram of the Grid service replication library. The left-hand side shows a ring-based topology while the right-hand side depicts a Leaf-to-Root Complete Binary Tree topology.
Table I. Initial and final setup of the experiments E1, E2 and E3, for the number of documents, the document size (in Kbytes) and the update frequency (in minutes).

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Number of docs.</th>
<th>Doc. size (Kb)</th>
<th>Update frec. (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Initial</td>
<td>Final</td>
<td>Initial</td>
</tr>
<tr>
<td>E1</td>
<td>100</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>E2</td>
<td>100</td>
<td>1000</td>
<td>1</td>
</tr>
<tr>
<td>E3</td>
<td>100</td>
<td>100</td>
<td>1</td>
</tr>
</tbody>
</table>

Authors of [28] provides more details about the distributed algorithms employed, the protocols for joining a group, the fault-tolerant capabilities of the library and its performance evaluation.

Additionally, the information system defines a mechanism that does not use communications to discard collaborations. This mechanism enforces the applications to register a lifetime for the collaborations. During the lifetime of the collaboration, the applications are responsible for updating the information system. Service replicas will discard the collaborations that have not received any update before the end of the life of the collaboration, independently and without communicating with each other.

In particular, the WSRF-based implementation uses the WS-ResourceLifetime specification to catch the event of the service container scheduling the destruction of the WS-Resource that handle the database. In this approach, the applications are responsible for updating the WS-ResourceLifetime.

### 3. CASE STUDY

This section presents three experiments that have been conducted to study the information system in a Grid environment. The study has been focused on analyzing the behavior of the information system when the workload increases due to an increase in collaboration activities, either due to the number of collaborations or the size of the messages transmitted over the network increases, or because the updates occur more frequently. The objective of this study is to have an idea of the limitations of the information system, and consequently to identify the applications that can benefit from the approach presented in this paper.

The experiments consist on measuring the time invested in replicating the status of collaborations, using an implementation based on WSRF. In this particular case, each communication among replicas of the information service includes an XML document.

Several different setups have been prepared to measure the influence of the number of documents, the document size and the frequency of update, in the performance of the system. Table I shows how the experiments were prepared.

The experiments E1 and E2 have conducted a series of measurements changing the size of the documents (1, 3, 5, 7, 9 and 10 Kbytes) and the number of documents (100, 300, 500, 700, 900 and 1000), respectively. The number of WS-Resources created in each replica coincides with the number of documents transmitted over the network, since each WS-Resource corresponds to a specific collaboration and each document includes the information that is intended to update the status of the collaboration. The frequency at which the system updates the status of the collaborations was the same in all the experiences of these experiments (10 minutes).

Similarly, the experiment E3 measures the influence in the frequency of update (10, 9, 7, 5, 3 and 1 minutes), fixing the other parameters.

The experiments have measured the time elapsed since the system triggers an update event that consists on sending the XML document associated with the collaboration to all the replicas of the information service, until the last replica receives the update. This time measures the latency of replication, which is especially important to study the scalability of the system.

The experiments have used 50 replicas of the information service, organized in a Leaf-to-Root Complete Binary Tree Topology. This setup reproduces a Grid infrastructure with 50 different sites.
geographically distributed, where each replica acts as a local cache of the information system for the site where it is placed. The structure has been created before measurements, in order to simulate collaborations between the 50 sites.

The experimental testbed consists of 50 GT4-based Grid service containers, deployed in the same number of x86 servers (Intel® Xeon™, 2.80GHz, L2 512 Kbytes), equipped with 2 Gbytes of memory and running CentOS Linux 5.3. A local area network interconnects the servers. Real traffic patterns and load conditions have been simulated in the network to reproduce real conditions.

Figure 7 shows three graphs with the results of the experiments E1, E2 and E3, respectively.

Figure 7a shows the results of the experiment E1. The replication times measured in E1 have similar values, regardless of the size of the documents used in each setup. In these conditions, the average replication time among the 50 replicas does not exceed 1.5 seconds. This result indicates that the documents used in the experiment are far from reaching the limits where the document size becomes the bottleneck for scalability.

However, the experiment has not been redesigned to include documents of larger size because, according to our knowledge, the largest size used in the experiment (10 Kbytes) is appropriate in the majority of cases. Moreover, using optimization techniques focused on compressing the documents for transportation could possibly increase the amount of information in the documents.

Figure 7b shows the results of the experiment E2. Unlike the previous experiment, E2 has found a barrier for scalability in the number of documents. In this case, the replication time has an asymptotic growth smaller than 9 seconds. However, when the number of documents is around 800, the replication time shoots up. These results demonstrate the capability of the system to manage collections in the order of a thousand documents, keeping the total time invested in replication below 10 seconds. This is acceptable for the majority of Grid infrastructures currently available. For example, the Spanish-Portuguese joint Grid infrastructure IBERGRID currently includes researchers from approximately 30 thematic VO subgroups related to the principal applications supported by the infrastructure, which are organized in 7 VOs. These VOs have a structure that consists of subgroups with the form: /VO/Country/Application. In this way, each VO can be subdivided per country, so that for each application there can be a subgroup of Spain and other of Portugal. The results of the experiment E2 show that the information system can support up to 30 dynamic collaborations per thematic VO subgroup in IBERGRID.

Finally, Figure 7c shows the results of the experiment E3. This experiment has found that for values of frequency of update above 5 minutes, this parameter has no visible influence in the replication time. However, values of frequency of update below 3 minutes result in excessive traffic due to the numerous exchanges of messages among the replicas. This produces massive network congestion that results in an overall slowdown of network traffic, thus causing a steep increase in the replication time. This result is consistent with the frequencies of update that are used in the majority of Grid infrastructures currently available. For example, computing elements in EGI experiment a latency of approximately 30 minutes in the status of the resources reporting to the workload management system.
4. CONCLUSION

This paper has presented an information system for the Grid that provides group communication services to standard Grid applications, transparently and in scalable manner. The system has several advanced features that are not currently available in a single Grid middleware, such as: several entry points to the information, persistent capabilities, support for advanced queries based on XQuery and support for the industrial standard WS-Policy. In particular, the use of XQuery enables developers with flexibility to generate complex queries to search for information in the Grid, while the support for WS-Policy enhances the interoperability of the information services.

The architecture of the system has been designed to support dynamic collaborations that could help address problems that involve only some participants of the VO. It provides a service that manages the information in an XML backed, seamlessly replicated across the Grid.

The information system has been stress-tested under realistic workloads, in a Grid infrastructure with 50 sites. Scalability has been evaluated in up to 1000 messages that can be up to 10 Kbytes in size each, updated with a frequency of 5 minutes.

Future works include further experimentation and optimization of the proposed approach. There are plans to evaluate the applicability of the information system to other problems, such as the monitoring of resource utilization. In this particular case, the performance of the system needs to be adapted to deal with information that is much more dynamic.

ACKNOWLEDGEMENT

The authors wish to thank the financial support received from The Spanish Ministry of Education and Science to develop the project “ngGrid - New Generation Components for the Efficient Exploitation of eScience Infrastructures”, with reference TIN2006-12890. This work has been partially supported by the Generalitat Valenciana and the Structural Funds of the European Regional Development Fund (ERDF).

REFERENCES


