Письма в ЭЧАЯ. 2008. Т. 5, № 3(145). С. 328–333

SERVICES FOR REPLICA CONSISTENCY HANDLING IN DATA GRIDS

M. Ciglan, M. Babik, L. Hluchy

Institute of Informatics, Slovak Academy of Sciences, Bratislava

The grid technology is a popular framework for collaborative, resource sharing computing infrastructure for many scientific domains. As wider scientific communities are engaged in the use of the grids, the view on the grid technology changes from the traditional high-performance computing perspective to a wider context. Data management is becoming one of the main areas of research interest in the scope of grid systems. We present services for replica update propagation and consistency handling, which try to address on the absence of software tools for maintaining the consistency among distributed data resources in the grid environment. We describe the architecture of the services as well as several important implementation details. We discuss the possibility of integration of consistency handling services with legacy software and grid middleware services. This gives us the possibility to benefit from legacy software functionality and, at the same time, to increase the data availability and fault-tolerance by means of data replication.

Для многих научных областей грид-технология является популярной средой для создания общей вычислительной инфраструктуры с распределенными ресурсами. Вовлечение широкого научного сообщества в использование грид-технологии значительно расширило содержательное представление о ней как только о перспективе традиционных высокопроизводительных вычислений. Управление данными становится одной из основных областей исследований в рамках грид-систем. В работе представлены сервисы для передачи обновлений копий и для обработки целостности, которые призваны восполнить отсутствие программных продуктов для поддержки целостности среди распределенных ресурсов данных в грид-среде. Описана как архитектура сервисов, так и некоторые важные детали реализации. Обсуждается возможность интеграции сервисов по обработке целостности с лицензионным программным обеспечением и грид-сервисами. Это дает возможность использовать функциональные преимущества лицензионного программного обеспечения и в то же время повышает доступность данных и отказоустойчивость посредством копирования данных.

PACS: 02.70.-c; 02.90.+p

INTRODUCTION

Grid technology is a form of distributed system which is focused on sharing and coordination of computer resources such as computing power, data, storage, applications, and network resources.

Resource sharing nature of grid technology encourages and provides means to enhance the cooperation in the scientific communities. Institutes that share a common goal can form a virtual organization, and they can share also their resources by using grid technology.

The concept of the computing grid arose from the need to share computing power, mostly for the jobs that use read-only data sets as inputs (data output from scientific experiments).

Because of this historical reason, the data management tools for grid computing were primarily designed to manage read-only data sets. This simplification encouraged the adoption of the data replication concept as a way to increase the data availability and to provide fault-tolerance. As the grid technology is gaining popularity among wider scientific communities, the importance of updatable data resources becomes evident. Tools that would allow data replication and updating of those data could increase the cooperativeness and make easy the data sharing within the community. Services for grid-data consistency handling are needed to achieve this goal. In this paper, we present service for replica update propagation that serves as the base block for consistency services and a consistency service built on the top of the later that ensures consistency of distributed replicas.

The paper is organized as follows: in Sec. 1, we present information on related work. Although most of the cited work deals with consistency models rather than with update propagation mechanics, we believe they can provide the reader with valuable information.

In Sec. 2, we present replica update propagation service, its architecture, and functionality description. In Sec. 3, we describe a consistency service built on the top of update propagation service and in Sec. 4, we discuss the method for integration of presented software with legacy applications and central grid services; we discuss possible benefits of such an approach.

1. RELATED WORK

Interesting discussion about data-consistency services for the grid environment was presented in [1]. The authors discuss the data consistency from the perspective of grids. A strict approach guarantees that all replicas are always 100% in sync and thus fully consistent. Due to the locking overhead of keeping huge amounts of distributed data in sync, 100% consistency is a very impractical solution for the grid environment. Thus, if knowledge about the data and user requirements (use cases) is available, one can relax this strict consistency requirement and allow certain parts of the data to be out of sync for a particular amount of time. For instance, a site A in a data grid may explicitly define that newly created files at other sites B, C, and D have to be transferred to the site A within two days. This means the replica creation process can be done within a 48 h time frame. Within this period the state of physical files can be inconsistent. Another example is that writable replicas have to be updated and synchronized every 10 min. The authors have proposed that grid consistency services should support different levels of data consistency, which could be used by the users.

In [2] authors propose architecture for data-consistency handling for a grid data-sharing service which decouples consistency handling from fault-tolerance management. This approach allows the consistency protocol and the replication strategy to be designed independently, while only a small interaction has to be defined.

In [3] authors discuss that the synchronization protocols used in homogeneous systems cannot be implemented in heterogeneous grid environment and propose a synchronization protocol for heterogeneous grid architecture based on quorum system able to handle multiple network partitioning.

In our work, we aim to build on existing (or at least on emerging) standards. We briefly present standardization efforts related to the problematic. Grid community, inspired by the movement in the field of Web Service technologies, produced Open Grid Services Architecture (OGSA), which provided the specification for state full web services. From the following

discussion with Web Services community, the Web Services Resource Framework (WSRF) standard [4] arose which keeps the concept of statelessness for web services, but provides means to model the access state full resources. Another relevant standardization effort is undertaken by Database Access and Integration Services — Working Group (DAIS-WG) of Global Grid Forum (GGF) standardization body. As the working group's name suggests, it is formulating standards for database access and integration services in the grid environment. OGSA-DAI [5] is a grid data management middleware that makes available in the grid system, via web services, data resources of different types (relational databases, XML databases, and files). OGSA-DAI provides functionality to query, update, transform and deliver data and allows to chain multiple activities to form a complex data manipulation procedures needed by the client applications for data processing and data integration. OGSA-DAI is intended to form a reference implementation of emerging DAIS-WG standard for data access and integration in the grid environment that is being developed by GGF [6].

2. REPLICA UPDATE PROPAGATION SERVICE

OGSA-DAI is currently most advanced tool for data access and integration for grid data resources, compliant with emerging GGF standards. This makes this middleware environment the best choice for building higher level data grid services. Our Replica Update proPAGATION service (RUPAGATION) uses OGSA-DAI framework as the basic operational environment. It is built as a set of modules, which can be plugged in OGSA-DAI data services. OGSA-DAI provide web service based interface to access the system functionalities, it is compliant with WSRF standard (as well as other standards of multiple flavors are available). It is a middleware that is operational on heterogeneous operating systems and supports vast variety of relational database systems. The aim of RUPAGATION is to form the base stone for higher level grid consistency services, which will provide automatic and autonomous synchronization of replicated data resources in the distributed, heterogeneous grid environment.

RUPAGATION provides common base functionality for different possible consistency services and approaches. It is able to catch and store updates submitted to the specified data resources, propagate those updates to the other replica sites and apply updates to distinct data replicas. RUPAGATION can provide its functionality for all types of resources supported by OGSA-DAI. This means that if we have an implementation of a consistency model that is built on the top of RUPAGATION service, we can use the same implementation to ensure consistency for a large number of relational databases, XML databases, and file resources. This virtualization provided for consistency services is, in our opinion, the most important aspect of this work. RUPAGATION was decoupled to several modules with clearly defined functionalities which can be deployed separately, according to the system requirements and needs. From the technical point of view, modules are set of OGSA-DAI activities.

RUPAGATION is composed of the following modules: (a) update catcher module, (b) update manager module, (c) update transfer module, (d) update applier module. Update catcher module is responsible of catching incoming updates on specified data resource. Submitted updates are catched and subsequently processed by update manager module. Update manager module provides functionality for creating new sets of updates and maintaining previously created update sets. Update transfer module is responsible for transferring update sets between

distinct grid sites and registering incoming update sets in update applier module. Update applier module performs updates of a data replica and keeps information about updates performed over specified data resource.

Update catcher module is the first interaction point for incoming update statements. It is responsible for catching those commands for later processing by update manager module. By default, update catcher module forwards incoming update statements to SQLUpdate activity (provided by OGSA-DAI), and they are executed against underlying relational database. Update catcher can be configured not to do so, if the consistency maintenance algorithm from higher level consistency service requires such a behavior. After receiving each update statement, module sends a notification to update manager that can trigger an update operation, in case that certain defined condition is fulfilled.

Update manager component is responsible for creating new update statements packages from newly arrived updates. It is also responsible for maintaining previously created update sets. After an update set is created, module sends a notification to defined consistency service, which can then react on this event. Update manager module also provides information about replica state, about updates applied on locally managed replica. Update transfer module is a chain of data delivery activities (mostly provided by OGSA-DAI middleware) that is able to transfer data files (update sets) to remote grid node operating RUPAGATION service. The transferred updates are registered within appropriate update manager module.

Update applier module is a module executing updates on replica stored in underlying relational data resource.

3. PCUM CONSISTENCY SERVICE

We used the functionality of the RUPAGATION service to implement consistency service for primary copy update model (PCUM). In primary copy update model, one replica of a data source is labeled to be a primary copy. For each nonprimary replica, only read operations are allowed from users or application. Update operations are applied only on primary replica. Changes performed on primary replica are then propagated and applied on nonprimary replicas.

The service creates an update set for each incoming update transaction. The update set is then propagated and applied to all defined replica sites.

Some of the replicas may not be accessible in the time of the update propagation of the new update set because of resource, service, or network problems. Those error situations must be handled. Consistency service at nonprimary replica site contacts primary replica consistency service upon startup, to retrieve identifiers of update sets that were not applied on local data resource. If such update sets exist, they are retrieved and applied. Another mechanism for handling update sets is functionality of the nonprimary replica services that check periodically the state of the primary replica and synchronize the data if several update sets were not applied.

More serious problem arises when the primary replica service is in error state (hosting site is down/consistency service is not started or malconfigured). In this case, the whole system will be dysfunctional, because no updates of the data resource will be possible. In order to overcome this error state, consistency services at nonprimary replica must be able to detect the failure of primary replica service and must be able to assign new primary replica site.

In current implementation, when a nonprimary replica site detects primary replica service malfunction (while performing periodical check for primary replica update state), it notifies all other replicas consistency services. The replica with the most actual update set is chosen to be the primary replica. In the case of the conflict (several nonprimary replica sites have the most actual updates), the IPs of the conflicting services, sites are compared lexicographically, and service with the lexicographically greatest IP value is appointed to be a primary replica service.

If an update arrives to a nonprimary replica site (that might be primary in the past) the update set is redirected to primary replica consistency service.

This naive consistency service provides us with the behavior similar to Read-One-Write-All (ROWA) consistency model [7].

4. EMPOWERING GRID MIDDLEWARE AND LEGACY SOFTWARE

In this section we argue that proposed services PCUM consistency and RUPAGATION service, are beneficial not only to the application that operates over replicated updatable grid data but also to the legacy grid middleware services, especially for the central grid services. The failure of a central grid service can have a significant impact on the operation of the whole grid system. Metadata services, services for replica location as well as security services for course-grained access control policies definition (such as CAS [9]) are typically implemented as central services in todays grid middleware. The distribution of those services with replicated and synchronized data can bring important improvement in the fault-tolerance of the grid.

We have chosen Metadata Catalog Service (MCS) [8] as a legacy central grid service to be integrated with our update propagation system for testing purposes of this approach. To integrate those two systems, only simple, syntactical changes were required in the source code of two classes of MCS. The call for JDBC update statements was replaced by invocation of update operation on (remote) primary replica data resource using web service interface of OGSA-DAI data service.

Modified version of MCS was deployed on nonprimary replica sites and the database structures of MCS were made available in data resources with RUPAGATION service deployed. Each deployed copy of MCS performed read operations on local database, the update operations were send to primary replica copy. PCUM consistency service then applied those updates to all deployed MCS's data replicas. The time efficiency of update operations decreased, but the load of read-only operation was distributed among the deployed copies of MCS. The read-only operations performed at site A were read at site A, but changes made by users/applications at site B were also applied to the data resource of the site A.

There are, in our opinion, two important points about this approach:

First is that even this approach is not a valid option for write-intensive applications, it might be considered for use in applications which perform large amount of read-only requests and only relatively few write operations. Using this approach, we can benefit from functionality provided by legacy applications and services and, at the same time, take advantage of the higher data availability provided by the concept of replication.

The second interesting feature is that, using our approach, we can turn noncollaborative legacy application into a utility that shares data with other deployed copies of the application

and allows users to produce new data which is instantly shared, in the scope of application, with other members of a (possibly distributed) community. (This is not the case for MCS, as this software product is already collaborative in its nature.)

5. FUTURE WORK

Presented software is (in the time of writing the paper) still in prototype stage. Work in the near future will be focused on refining and optimizing the implementation. We also plan to implement other consistency models on the top of RUPAGATION system, such as quorum based consistency protocols.

CONCLUSIONS

In this paper, we have presented a set of software modules that forms an update propagation service for relational data resources in the grid environment and a consistency service that uses the functionality of update propagation service. Described software modules are implemented as a collection of activities pluggable into OGSA-DAI framework for grid-data access and integration. OGSA-DAI environment was chosen because of the effort of its authors to stay compliant with recently emerging standards for data access in grids. Proposed system is intended to provide grid-data consistency handling. We have also described a possibility to integrate proposed update propagation system with legacy applications, and we have highlighted possible benefits of this approach.

Acknowledgements. This work is supported by projects MediGrid EU 6FP RTD GOCE-CT-2003-004044, VEGA No. 2/6103/6.

REFERENCES

1. *Dullmann D. et al.* Models for Replica Synchronization and Consistency in a Data Grid // 10th IEEE Symp. on High Performance and Distributed Computing (HPDC-10'01). 2001.
2. *Antoniu G., Deverge J.-F., Monnet Sb.* Building Fault-Tolerant Consistency Protocols for an Adaptive Grid Data-Sharing Service. Research Report No. 5309. Syst. Commun. Project PARIS. 2004.
3. *Goel S., Sharda H., Taniar D.* Replica Synchronization in Grid Databases // Intern. J. Web and Grid Serv. 2005. V. 1, No. 1. P. 87–112.
4. WSRF standard. http://www.globus.org/wsrf
5. OGSA-DAI. http://www.ogsadai.org.uk
6. DAIS-WG and OGSA-DAI. http://www.ogsadai.org.uk/about/ogsa-dai
7. *Ceri S. et al.* A Classification of Update Methods for Replicated Databases. TR STANCS-91-1392. Stanford Univ., 1991.
8. *Singh G. et al.* A Metadata Catalog Service for Data Intensive Applications.
9. *Canon S. et al.* Using CAS to Manage Role-Based VO Subgroups // Proc. of Comp. in High-Energy Physics'03 (CHEP'03). 2003.