Abstract

In this paper, we describe the GlobData project development which consists of a middleware architecture for maintaining the consistency of objects replicated, which are persistently stored into an RDBMS. Several consistency levels are offered, as well as a set of ad hoc optimistic consistency protocols. It is also emphasized the project experience obtained from the collaboration among different European research groups, specialized in distributed systems, and European enterprises interested in the development of this kind of applications. This project was granted by the European Union inside the Fifth Frame Program for the Information Society Technology program.

1. Introduction

The increasing business globalization, and the growing diffusion of Internet and of Wide Area Networks (WANs) in general, creates for corporations and institutions the need to store very large amounts of data in distributed databases that span geographically. These data must be accessed by users, also geographically scattered, that must consult or work with them, very often in a concurrent way. This sharing and interaction is not well handled by traditional database methods, and requires new capabilities in terms of behavior and availability. Nowadays applications must be implemented with the highest degree of data security, consistency, efficiency, reliability and availability.

Replication techniques are used in order to achieve the former requirements. Information usually is located in different departments and delegations, therefore remote transactions must be started. If these data accesses are performed locally, over the replicated repository data, on the one hand, the application performance is increased, and, on the other hand, site failures may be properly handled through different nodes. Enterprises are very interested in these topics, as well as node recovery whenever a node has been down for a while.

These topics have lead our respective research groups along with some international research groups and some European industrial firms to develop an architecture that satisfies these requirements. It provides persistent data replication following the object oriented paradigm, besides it supplies an interface which can be easily used by Java applications too. A wide range of consistency protocols implemented in this architecture allows to applications to switch to the one that best suits to them and to maintain data coherence [7, 9, 10].

Our industrial are used to develop applications following the object oriented paradigm, although data source for these applications are stored in RDBMSs. One of the requirements of the enterprises is to store data on RDBMS, thus porting existing data to new applications devised in COPLA would be easily done. This is one restriction imposed by the industrial partners, instead of using an OODBMS, which may be more efficient, although RDBMS are more familiar to enterprises, final users and all the community, in general. Distributed applications making use of distributed databases, as occurred with centralized DBMS, may start multiple remote transactions with a given transaction isolation level. Often, these application needs additional overhead to manage possible remote site failures so as to provide high-availability applications. Besides, the replication facility featured in this architecture provides better performance due to the fact that the former remote transactions may be performed locally and, therefore, increasing its performance.

Object state storage in an RDBMS leads to an additional problem. There is an imbalance between the entity relational model and the object oriented paradigm that must be solved. We have defined a translation pattern for an schema, defined in COPLA, to its equivalent relational model, performed by its respective compiler [2, 1].

The rest of the paper is organized as follows: Section 2 deals with a review of the project goals; Section 3 explains the system architecture; key issues addressed by research and industrial partners are outlined in Sections 4 and 5; the collaboration experience is depicted in Section 6; and, finally conclusions ends the paper.
2. Globdata project

The overall objective of GlobData is to design and produce an efficient software development tool and support system called COPLA (Common Object Programmer Layer Access), to provide application developers with a global view of an object database repository. It provides a transactional access to geographically distributed persistent object repositories, no matter where they are physically located. Therefore, it reduces the traditional bottleneck of remote access, allowing application developers to efficiently work against a single logical object environment, although the actual objects are geographically distributed. This work has been supported by the European Union under the Fifth Frame Program for the Information Society Technology program and continues with our research groups with the collaboration of the Spanish Government.

This research project is structured in three different levels:

- **Strategic objective.** So as to solve the problem of efficiently access large amounts of information geographically distributed. Several existing solutions have been exhaustively studied until the given solution has been developed.

- **Technological objective.** We must create an innovative, transparent and portable platform for global application development requiring at the same time minimum changes in user methodologies. The goals here are:
  
  - Implementation of an architecture and API for building applications for geographically distributed environments. We are focusing our efforts in developing a global solution for data replication and its availability following a complete object oriented approach.
  
  - Implementation of an architecture to integrate data sources to support applications. This results in a definition of an object definition language and an object query language so object repositories may be defined and accessed [1]. It has been also developed a library to simplify object state access to application developers (proxies).
  
  - Implementation of the security and consistency mechanisms needed for an efficient data interchange among the geographically disperse sites. The main problem of data replication is guaranteeing data consistency, so the system introduces an additional overhead for maintaining data consistency. Besides, applications making use of replication must implement additional pieces of software in order to manage the access to distributed resources, thus incrementing their development complexity.

- **Business objective.** Disseminate the use of COPLA through a sound exploitation plan so it can become widely available.

3. COPLA software architecture

COPLA architecture is divided into three different layers as it is shown in Figure 1. These layers are implemented in Java and may reside in different machines since they use an ORB to interact among them. In the following subsections are outlined the different components of this architecture.

![Figure 1. COPLA Architecture.](image)

Following a top to bottom approach the first layer is the library. It provides an object oriented interface to interact with objects defined following the ODMG standard [1, 8], which are persistently stored in an RDBMS. The COPLA manager layer is the key component of the architecture, it manages an object cache and the consistency among different replicas. Therefore, it needs an specific protocol that defines some specific rules so as to update replicas following the same order. It has to determine the existence of conflicts between different nodes trying to concurrently access or modify the same object.

As it is shown in Figure 1, there is another module, inside the same layer, that implements the given consistency protocol. It is exclusively responsible for the communication among the different replicas (or nodes) which COPLA is composed of. This module will propagate the updates performed on a node to the rest of nodes, and monitor the behavior of the rest of nodes. Therefore, node failures or joins may be detected and the respective recovery protocol will be responsible for these tasks. The final performance of the system highly depends on the kind of consistency protocol implemented. Actually, it has been implemented several optimistic consistency protocols [7, 9, 10]. Replication techniques are classified depending on the time where update propagation takes place, so we have two different kinds of protocols:

- **Lazy protocols.** The updates are propagated once the
original transaction has successfully finished. Thus, the response time is minimized, although the abort ratio may increase since transactions may often access outdated objects.

Eager protocols. Update propagation takes place before the original transaction finishes. Thus, sequential consistency is guaranteed, but, on the other hand, more subtle communication protocols are needed in order to propagate the updates.

The last layer, called Uniform Data Store (UDS) [2], is in charge of providing an object oriented approach to the COPLA manager in behalf of the RDBMS used to persistently store the state of an object. Therefore, it hides all “relational issues” to the rest of the system. The features provided by this layer are:

A transactional object repository access. It creates a transaction inside the database for each session opened in the COPLA architecture. This defines a transaction with a given transaction isolation level according to the application needs.

It supports different kind of consistency protocols and provides persistence mechanism for metadata needed by them. This is an important feature, since several protocols may be switched inside our architecture. The UDS stores metadata needed by them in order to properly work.

Compilation and execution of object oriented queries. A query compiler has been designed and implemented to translate object oriented queries to their equivalent SQL queries according to our translation proposal [1]. Great feedback has been given here from our industrial partners.

Objects updates are transparently performed on the RDBMS. The UDS is responsible for properly updating the state of an object in the underlying RDBMS. It mainly translate object requests and updates into SQL sentences according to our mapping proposal [1, 2].

4. Key issues addressed by the research partners

The research groups (Instituto Tecnológico de Informática [6], Fundação da Faculdade de Ciências da Universidade de Lisboa [3] and Universidad Pública de Navarra [11]) have to address the following key issues:

Efficient communication protocols to ease information and control data interchange. We have developed group communication mechanisms and monitor memberships for this architecture. We have proved several group communication primitives varying from total order multicast passing by uniform multicast to the simple unicast.

Mechanisms and protocols for correct synchronization of the different application parts. COPLA usage by an application implies the addition of special software tools so as to properly work and concurrently access over a shared object repository. Moreover, these repositories are replicated throughout all the nodes, so special synchronization must be taken into account for object state consistency. Besides, object state must be shown to the application in a given programming language, in our case Java.

Mechanisms for secure data exchange. Some validation mechanisms, such as user and password validation, is provided. Moreover, information exchange between components is encrypted utilizing the secure socket layer.

Protocols for coherence maintenance in shared distributed storage. Several protocols have been designed and implemented in COPLA in order to maintain data consistency, as it is depicted in [7, 9, 10].

Long-term transaction handling. We have defined the session term which may include several transactions on the underlying RDBMS, and several session modes, such as: plain, checkout and transaction modes [2, 7].

Applications building mechanisms based on distributed object request standards. We have defined an object definition language, and its respective object query language, so as to create, access and modify an object repository [1]. As it was outlined before, it follows the ODMG standard [8].

Interoperability issues related to the possible use of different distributed computation standards. We follow the CORBA standard, so several IDL interfaces have been defined to interact with the rest of COPLA components. We have also implemented proxies in Java language to represent an object state in the COPLA architecture.

5. Key issues addressed by the industrial partners

On the other side, industrial partners (GFI Informática [4], Investigación y Desarrollo Informático [5]) key issues are:

Easy adaptation to support different commercial languages and databases. Applications developed in this architecture may be implemented in different object oriented programming languages, that is why we have chosen CORBA as the intercommunications mechanism. They also work with several commercial databases, such as Oracle, we intended the persistence mechanism may easily be ported to several RDBMS.

Ease of use by programmers with no special skills requirements. Costs associated to acquire skills on this architecture must be scarce, so the design and implementation of applications must be sound to final users. Some special administration features must be designed and implemented by the industrial partners so as to port existing software. Solutions offered to their clients may be ported to this new architecture at minimum training costs.

Scalability and efficiency. These are the main drawbacks of this kind of architectures some solutions scale well but lack of efficiency and viceversa. Some implemented consistency protocols fit well for a few nodes but they loose their efficiency once some more nodes are added. They have compared previous existing solutions with ours so that all possible drawbacks are quickly found and fixed.

Adaptation to their ongoing projects. There are interesting commercial projects our industrial partners were developing: logistics, telecommunication services billing, management of employees, etc. These projects can be improved
and enhanced with our replication architecture proposal. Mainly, because it integrates features that usually implies the integration of different software solutions.

As it has been early highlighted, feedback from our industrial partners has been essential to develop special requirements at the application level such as query mechanisms improvement and collection management.

6. Project experience

On the following we will develop a set of key issues that have been our hobbyhorse through the project deployment:

Levels of consistency. COPLA offers three different levels of consistency: plain, which offers no guarantee about up-to-date objects; checkout, this level is very similar to the concurrent development of software in a project; and, transactional, which is the one that provides one copy serializability. Enterprises greatly appreciate these different levels of consistency that may easily be ported to the traditional isolation levels featured in the relational model.

Object oriented approach. We have explained that enterprises are used to develop application following the object oriented paradigm, due to its code simplicity and modularity. Although, it is at a higher level of abstraction, once you are familiar, programming is a simple task. Enterprises appreciate this full object oriented approach, that allows to store the state of an object in an RDBMS and coexist with data of former applications.

ODMG standard. This is one of the most important topic we have dealt with enterprises. Applications (schemes) developed for COPLA architecture must follow the ODMG standard (in fact, we have implemented a subset of it). Data sources for existing enterprise applications follow the entity relational model, so porting existing data sources to COPLA is a non-trivial task. Along with the definition of a schema, COPLA provides a language to locate objects defined by means of object oriented queries based on the ODMG standard. This is a powerful feature that has not been taken fully advantage, enterprises has been using it as an extension of standard SQL queries. In other words, only a 10% of its capability has been used to fetch objects in a repository. Therefore, instead of using relationships between classes they still use key attributes as the matching parameter to locate objects. As it has been pointed out earlier, one of the enterprises requisites is that an object state must be stored in an RDBMS, thus porting existing data source to COPLA may be done without any additional effort, but this has not been as easy as it seemed at the beginning.

Consistency management. Along the development of this project we realized that our industrial partners were very interested in a serializable transactional behavior that guarantees an eager replication to all nodes. This approach which is correct and valid, is not the best approach to all applications they are interested in. Thus, some applications have worse performance than previous existing solutions. Since, they usually do not confide in lazy protocols or other consistency levels which may lead to a better performance of them. In short, they want to develop commercial applications that supports concurrency access, supporting ACID properties, guaranteeing at all time that all sites contain an up-to-date version of all objects.

Node recovery. This topic was of great interest to our industrial partners, they were very interested in topics dealing with node failures or node joins. We have developed a monitor membership so as to detect node failures and recoveries. They were interested in how the recovery process is performed by our consistency protocols, i.e., if we transfer the whole repository, the state of outdated objects, or SQL statements missed. They were really interested in these management tools.

High availability and failure detection. We are offering a high available replicated object oriented database, which is greatly appreciated by our industrial partners. Thus, they are not aware of node failures, a failure detector is responsible for that task. Besides, there is no need to perform regular data backups, because the implemented consistency protocols have their own recovery protocols. They are responsible for updating a node after a failure.

Technology investments. Enterprises utilize technology to develop applications that best match their customers needs. On the other hand, researchers focus on a technology for their own sake, after all academic research is rewarded for innovation and thoroughness. An enterprise does not research to find out how to develop a complete software architecture. It is very interesting and also of great value the effort done by our industrial partners as to develop and utilize a technology whose commercial exploitation is not immediate.

7. Conclusions

Replication of data is viewed as a way to increase performance and availability for distributed applications. Nevertheless, it implies the risk of inconsistency so accesses to data sources must be carefully treated, as well as techniques to recover from site failures and redirect applications to alive sites whenever a node may fail. Data replicated must be stored in a persistent storage media, such as Database Management Systems (DBMS), either relational (RDBMS) or object oriented (OODBMS). We have adopted the former although we use an object oriented approach to develop the replication software architecture. Additional implications must be taken into account in order to translate an object state to the entity-relational model. Therefore, we are offering a unique layer that offers object state replication that has been persistently stored in an RDBMS.

Several types of consistency levels are provided, so different applications may increase their performance, decreasing their read transactions response time, although increasing a little bit the abortion rate. Thus, along with the proper selection of a consistency protocol for a given application may even increase the behavior of it.

It is important to highlight that this project has involved a consortium composed by academic and industrial partners.
that has benefited both sides. Sometimes research topics are far away from currently commercial needs. And, on the other hand, enterprises does not enough invest in the research and development areas in order to discover new trends towards solutions to existing problems. In the paper we have developed the key issues of both parts, and outlined the feedback coming from both sides as it was highlighted in the project experience. This project has been developed inside the European Union Fifth Frame Program for the Information Society Technology.

This project continues its exploitation, thus COPLA has become a research platform to develop consistency and recovery protocols for replicated objects persistently stored in an RDBMS. Several applications have been designed and implemented for their commercial usage. Actually, we are also developing a second edition of this software, which will offer a more flexible approach towards consistency protocols usage, allowing application developers to work with several of them in a concurrent way. Thus, different applications will use different consistency protocols, so as to improve their own performance instead of choosing the best consistency protocol among all the possible applications used. This is currently accomplished as a research project supported by the Spanish Government under research grant TIC2003-09420-CO2.

References


