Database Assessment for PDMS

Abhishek Gaurav, Nayden Markatchev, Philip Rizk and Rob Simmonds
Grid Research Centre, University of Calgary.
http://grid.ucalgary.ca

1 Introduction

This document describes the database issues related to the Proactive Data Management System (PDMS) [6]. PDMS uses databases for a number of tasks. PDMS uses the Replica Location Service (RLS) [5], that is implemented using a database and can propagate updates between database servers. It also uses the Meta Catalogue Service (MCS) [7] schema, in another database. PDMS also currently uses the Reliable File Transfer (RFT) [3] service that also uses a database. Furthermore, PDMS uses a database to save its internal state so that its internal state is preserved between server restarts.

The rest of this document is organized as follows: Section 2 gives an overview of relational databases. Section 3 describes database issues for the main PDMS server. Section 4 describes the RLS service, how it is used by PDMS and its database dependencies. Section 5 does the same for the MCS service and Section 6 describes database issues with the RFT service. Section 7 concludes the document.

2 Relational Database Overview

All the databases used by the systems discussed in this document are relational database management systems (RDBMS). A RDBMS is a database system that uses the relational model based in predicate logic and set theory. Although there is some disagreement about what constitutes a relational database, in practice databases that represent views of data as rows and columns are considered relational databases. The separation of data in tables of rows and columns allows users to organize data so that it is easier to mine and easier to update. This facilitates the addition, modification and retrieval of data.

All current RDBMSs use the Structured Query Language (SQL). SQL is a declarative computer language that operates on sets of data and is used to create, modify, retrieve and delete data from a relational database management system. SQL was adopted by as a standard by the ANSI (American National Standard Institute) in 1986 and since then there have been a few minor revisions of the standard.

Open source RDBMSs tend to be favoured by middleware developers because they are available at no cost and can be modified if required. Commercial RDBMS tend to have more

sophisticated management features and enhanced capabilities for replicating their contents, or for clustering groups of servers that appear as a single RDBMS. Another difference among RDBMS is their compliance with the ANSI SQL standard. For example the SQL:2003 standard which introduced XML related features and defines new data types, currently is only fully implemented in PostgresSQL. Other RDBMS, both commercial and open source, only implement a subset of these data types. Other differences include restrictions to the number of columns in a table, restrictions to the size of the query, restrictions to the maximum length of a table row and support for standard defined built-in functions and vendor specific functions.

2.1 Open Source RDBMS

The two most widely used open source RDBMS packages are MySQL and PostgreSQL. MySQL stores data in separate tables rather than recording all the data in a single logical unit. This approach makes data much quicker to access and modify, easier to administrate and easier to describe. PostgreSQL is another open source RDBMS that is similar in many ways to MySQL but the two systems differ in their technology and their purpose of use.

PostgreSQL implements database logic that supports views, triggers and stored procedures. Stored procedures allow processing to move from the application layer to the database system. Unlike MySQL, Postgres also allows user defined data types. MySQL is optimized for fast execution of simple streams of queries. This makes MySQL more suitable as a lightweight storage engine for web site and cache type environments, while PostgreSQL's extra functionality makes it more applicable to enterprise data centres and mainstream business environments. Table 1 summarizes some of the most important difference between the two RDBMS.

2.2 Commercial RDBMS

There are a number of commercial RDBMS such as Oracle, IBM's DB2, and Microsoft SQL Server, that provide similar features to the ones implemented in PostgreSQL and MySQL. In addition to the basic features provides by open source RDBMS, these commercial systems implement more sophisticated replication mechanisms, provide more user friendly administration and monitoring tools as well as more advanced backup strategies. However, few applications have these advanced features as requirements, therefore open source RDBMS are sufficient for most tasks. In addition to this, commercial software requires expensive licenses making the adoption of an open source RDBMS even more appealing.

Microsoft's SQL server [2] provides database mirroring and failover clustering as well as .NET framework integration. Although such features may be desirable in a business environment because of the high level of support a commercial vendor could offer, they are not very practical in educational environment because vendors are unwilling to release the source code of their products. Another restriction of the Microsoft SQL server is the requirement to run on Windows operating system.

Table 1: MySQL and PostgreSQL features comparison

Feature	MySQL 4.1.0	Postgres 7.3.3
SQL:2003 Datatypes support	Partial	Full
ODBC 3.0-specific datatype support	Full	Partial
Function CHAR_LENGTH	Not supported	Supported
Function CONCAT (ODBC-3.0)	Supported	Not supported
Function INSERT (ODBC-3.0)	Supported	Not supported
Non ANSI SQL92 group functions support	Full	Partial
Number of tables in JOIN method	31	+64
Join update with many tables	Supported	Not Supported
Binary strings	Not supported	Supported
Maximum return string size from function	1048576	+8000000
Maximum constant string size in SELECT	1048565	16777207
Column name length	64	+512
Table name length	64	+512
Table renaming support	Supported	Not supported
Maximum text or blob size (bytes)	1048543	+8000000
INSERT with Value list	Supported	Not supported
Maximum number of columns in a table	2699	1600
Maximum table row length	65502	103279
Maximum query size (bytes)	1048574	16777216

The Oracle database has a very advanced clustering support. It allows users to run any package of custom applications across a cluster of servers without modifying the applications, creates and operates database clusters with a set of built-in clustering services and provides automatic workload management. Oracle also provides a sophisticated and very high performance database backup utility called Oracle Recovery Manager (RMAN). Traditional database backup operations required the database administrator to record the specifics for each backup operation. On the other hand, with RMAN database backup and recovery could be as simple as a single command.

Unlike Microsoft's SQL server and like Oracle database IBM's DB2 supports a wide range of hardware platforms and operating systems. One of the key strengths of DB2 is its speed. According to recent benchmark tests [1] DB2 outperformed Oracle database by 68% on SAP benchmark using half the number of CPUs. DB2 provides an easy integration in WebSphere/Java environment which is IBM's integration and application software.

3 PDMS Server Database

PDMS uses a database to record the system's internal state. Saving the system state allows PDMS to recover from the state it was before the restart, thus providing 'fault tolerance.

Table 2 and Table 3 describe the data PDMS records in order to preserve its state information.

Table 2 holds information about the replication jobs that are currently in progress. Each replication job is assigned a unique replication ID. The database keeps track of the total number of files to replicate as well as the number of transfers finished, active, retrying, failed, and pending. The status of the replication job is one of the states from Table 3.

Replication Job States
RECEIVED
DOES_NOT_EXIST
SERVER_ERROR
MCS_ERROR
RLS_ERROR
NO_MATCHING_LFNS
LFNS_RETRIEVED
PFNS_RETRIEVED
REPLICATING
FAILED
VAULTDIR_CREATED
VAULTDIR_ERROR
COMPLETE.

The credentials of the user initiating the replication request, the request itself and the RFT endpoint reference are recorded, so that PDMS can recover from a restart of the server. An endpoint reference is a construct that is part of the web services addressing specification [4] designed to support dynamic generation and customization of service endpoint descriptions. In an event of a PDMS restart, part of the restart process of the service is to check whether there were any outstanding replication jobs just prior to the service restart. If such jobs are present, the action depends on whether RFT was used as a transfer agent of the PDMS custom method is selected. In the case of the custom method, the user credentials and the replication requests are sufficient to restart the replication job. In the case of RFT, the RFT endpoint reference is also needed. The database is updated upon a change in the state of any of the transfers; for example a completion of a file transfer will decrement the number of active transfers and increment the number of finished transfers.

The default method of moving data in PDMS is to make calls to a RFT service. The functionality for PDMS to replicate files itself is not fully supported or tested. When PDMS does replicate files itself, it makes use of a table information similar to that in Table 3. Table 3 contains the state information maintained for a LFN that is being replicated. The replication ID refers to the replication job the LFN belongs to. The Bytestransfered field is updated every time an update for the current number of bytes transferred for the LFN is received from RFT or the GridFTP server. The size field holds the size of the physical file in bytes. The status of the replicating LFN is one of the following: PENDING, REPLICATING, FINISHED, FAILED.

Besides being read during a system startup, the PDMS's database is also consulted when a monitoring request is sent from the PDMS client to the server. Notification based updates are also supported. The PDMS database is also read from a monitoring web portal but it also could be easily integrated to provide information to a more general monitoring service.

3.1 PDMS Database Issues

PDMS is developed to use with MySQL RDBMS but it can be modified use any other RDBMS that has a Java Database Connectivity (JDBC) interface. JDBC is a technology that provides cross-DBMS connectivity for many SQL database management systems. PDMS uses the JDBC's APIs to communicate with the underlying database. In our test

Table 2: Replication Jobs Table

Entry	Type	Description
replication ID	Integer	Unique replication ID
totalLFNs	Integer	The total number of LFNs for the replication Job
finished	Integer	The number of finished transfers
active	Integer	The number of active transfers
retrying	Integer	The number of retrying transfers
failed	Integer	The number of failed transfers
pending	Integer	The number of pending files to be transferred
status	Varchar	The status of the replication job
cred	BLOB	Credentials of the user requesting the replication
replicationRequest	BLOB	The replication request
rftepr	BLOB	RFT endpoint reference

Table 3: LFN Replication State

Entry	Type	Description
repid	Integer	The ID of the replication job the LFN belongs to
lfn	varchar	LFN name
bytesTransfered	Big Integer	The total number of bytes transfered for the LFN
size	Bog Integer	The size of the PFN corresponding to the LFN
status	Tiny Integer	The replication status for the LFN

deployments the PDMS service and the MySQL server are located on the same physical machine, but the code is written so that the database system could reside on a remote host. The use of another JDBC compliant would require installation of another JDBC driver, as well as some code modification in the instantiation of a database connection. Other databases have not been tested and there is currently no compelling reason to support other databases.PDMS issues simple queries and requires a fast response from the database and since this is the environment in which MySQL excels over other databases, a replacement of the database system could degrade performance without offering any benefit.

4 Replica Location Service

The Replica Location Service (RLS) [5] is divided into two parts the Logical Replica Catalog (LRC) and the Replica Location Index (RLI). The LRC stores and provides a mapping between logical file names (LFNs) and their physical locations. The RLI provides a mapping between LFNs and LRCs that know about them. Both are primarily designed for speed of access in the case of millions of files. For example, the RLI stores data in a lossy format that results in false positives, i.e., it provides LRCs that may not have a particular LFN indexed.

When a physical file is registered with the service a unique LFN is associated with that file. When the file is replicated to one or more physical locations the LFN associated with the original copy is also associated with the physical locations of all of the replicas. In the case of multiple replicas, there exists one unique LFN that points to each of the physical file names (PFNs). RLS uses a database to store this information.

4.1 RLS Database Issues

RLS requires a relational database management system (RDBMS). It also depends on an Open Database Connectivity (ODBC) driver and manager. In principle, this results in flexibility for the database used. In practice, we have not had success when RLS is used on a system that RLS was not developed on.

The use of open-source RDBMSs results in other problems. The suggested database systems according to the Globus documentation are PostgreSQL or MySQL. The Globus Toolkit documentation identifies a problem with the a release of iODBC, a software component that is used to interfaces to the ODBC layer of MySQL and the latest version of the MySQL database. Such software incompatibilities necessitates experimenting with different releases of database components until a compatible set can be determined. We also encountered difficulty building the database requirements on Opteron processors for which an appropriate versioning combination, i.e., software releases that are interoperable, of open-source RDBMS, ODBC driver, and ODBC managers could not be built. For this reason using an RLS on a Linux x86 system with MySQL (which the Globus team appears to be developing with) is generally best. This is expected to change as the Globus team is rapidly improving its quality assurance process related to the Globus Toolkit. The versions of the database systems and components that we are currently developing with are as follows:

- MyODBC version 3.51.06
- MySQL version 4.0.26
- libiodbc version 3.51.2

5 Meta Catalog Service

The Metadata Catalog Service (MCS) is repository that stores descriptive information for logical data items. MCS distinguishes two types of data items – logical files and logical collections. A logical file uniquely identifies the contents of a file in which there may be several physical copies. A collection may consist of logical files, other collections or a combination of both. MCS provides users with the ability to describe files and collections through attributes. This allows users to query the service based on particular attributes, which facilitates the discovery of desired data.

PDMS uses MCS data structures and replicates its functionality. This was primarily done to take advantage of the requirements analysis done by the researchers who developed

MCS. It also allows PDMS to more effectively interoperate with tools built to use the MCS service. The MCS service itself is not used for performance reasons discussed in Section 5.1.

5.1 MCS Database Issues

The PDMS system does not use the MCS service directly. For efficiency, it only uses the MCS schema as well as MCS libraries for querying and modifying the database, directly. The use of the libraries enforces the same consistency requirements that MCS uses. The use of the schema also allows and MCS service to be presented if necessary as both the MCS service and the PDMS service can access the same database.

It is possible for PDMS to use the MCS service, however because MCS is a web service, the Grid Security Infrastructure (GSI) handshake can be very expensive. This is particularly the case from the view of the client that has already performed a GSI handshake with the PDMS server and is waiting for a response.

5.2 Use of Alternative Databases with MCS

The MCS service has been developed and tested only on MySQL. The schemas themselves are only provided in MySQL format. The code for the MCS system uses standard JDBC interface so using another database should be possible.

At the writing of this document the PDMS system is coded to use MySQL databases and this option is not configurable. Modifications would be required if it were to work with other JDBC interfaces. It would require replacing the direct instantiation of the database connection object with the instantiation of a database connection object of the new database. This instantiation occurs in only single place in the code.

One outstanding issue that needs to be explored is the replication of MCS. At the time of writing, MCS (or its database) is a monolithic server. This represents a single point of failure. The MySQL RDBMS can replicate read only "slave databases" which could introduce some redundancy. It may also be worth exploring the opportunities available with commercial databases such as Oracle parallel server. This would allow data to be replicated in a manner such that requests could be served from different database servers. This would distribute load across multiple physical hosts which will contribute to improved robustness and efficiency.

6 Reliable File Transfer Service

The Reliable File Transfer (RFT) service is a Web Services enabled component of the Globus Toolkit which can be used to manage third party GridFTP transfers. An important detail of RFT is that the service uses the credential of the user requesting the transfer, i.e., the service acts as the user. The service uses a database to keep record of the transfer requests and the progress of the transfers. It uses the GridFTP service which is the transport mechanism for the physical movement of data. RFT provides a robust manner to reliably transfer files

in the face of server shutdowns and network outages, making effective use of GridFTPs restart mechanisms. RFT information about the status of the file transfers, i.e., whether a file transfer is pending a, in progress, finished, or the transfer failed. A limitation to the service is that it does not guarantee the integrity of the transfered files anymore than the underlying protocol does. RFT also only takes one source for a transfer instead of multiple sources which would provide more fault tolerance.

6.1 RFT DataBase Issues

The default configuration of RFT assumes the availability of a PostgreSQL RDBMS, however the service could also be configured to use a MySQL RDBMS. If configured to use PostgreSQL, RFT needs PostgreSQL version 7.1 or greater. The version suggested by the Globus developers is 7.3.2. RFT could also use the MySQL database, which requires the installation of MySQL drivers. Globus developers suggest MySQL Connector/J version 3.1. Connector/J is a Java driver that converts JDBC calls to native calls used by MySQL. If the MySQL version is prior to version 4.1, rft_schema_mysql_pre4.0.sql schema should be used. RFT has not been tested to work with any commercial database but is designed to work as long as the DBMS supports JDBC.

7 Summary

This document has described some of the database issues faced in the development of PDMS. PDMS uses a number of services that rely on databases. Currently it is difficult to use anything other than same RDBMS and libraries as was used by the implementers of these services, since there are still a large number of interoperability issues that they need to address. It may be possible to move the MCS schema to a commercial database that has more support for replication.

References

[1] DB2 vs. Oracle SAP benchmark. http://www-306.ibm.com/software/data/highlights/benchmarks/20050515.html.

[2] Microsoft sql server web site. http://www.microsoft.com/sql/default.mspx.

[3] Reliable file transfer web site. http://www.globus.org/toolkit/docs/4.0/data/rft/.

[4] Web services addressing 1.0 - core. http://web4.w3.org/TR/2005/WD-ws-addr-core-20050215/.

[5] A. Chervenak. Giggle: A framework for constructing scalable replica location services. In *Proceeding of the IEEE Supercomputing 2002*, 2002.

[6] U. of Calgary Grid Research Centre. PDMS project web site.

[7] G. Singh, S. Bharathi, A. Chervenak, E. Deelman, C. Kesselman, M. Manohar, S. Patil, and L. Pearlman. A metadata catalog service for data intensive applications. In *Proceedings of Supercomputing 2003 (SC2003)*, 2003.