Exploiting database technologies for Digital Libraries

Joshua W. Hui Ajay Sood Mahadevan Subramanian Parag V. Tijare

IBM Almaden Research Center, San Jose, CA 95120
{jhui, ajay, maha, parag}@almaden.ibm.com

ABSTRACT

Management of content within a Digital Library system should address the issues of access control, integrity and recovery among other things, for both structured and semi-structured data. Most vendors either address the above issues in a custom fashion or do not address them at all. Our approach is to provide a Relational Database Management System (RDBMS) solution to the above issues. We take full advantage of the RDBMS capabilities of access control, integrity and recovery for structured data and extend the RDBMS to support the same for semi-structured data through a technology called DataLinks [7]. DataLinks extends Database technology to data residing outside of the database. Administrative tasks become simpler in a Digital Library System using the DataLinks technology. This paper describes how DataLinks feature within an RDBMS would benefit a Digital Library System in managing its content.

Keywords

RDBMS, access control, integrity, recovery and replication

1. INTRODUCTION

Digital Libraries manage wide variety of digital information like text, video, audio, images and so on. The major usage is in the management of semi-structured data which forms about eighty-five percent of the world's data [1]. The other fifteen percent is the structured data which is traditionally managed by a Relational Database Management System (RDBMS). The digital libraries, for example, can help you store, search, disseminate and protect all the scanned insurance policies in the premises of an insurance agency. A RDBMS will be more typically used by a retailer to maintain the catalog numbers and their price information.

In a typical digital library architecture, the module helping with the storage and management of the actual data could be termed as the Content Management System (CMS). One of the common architectures representing a CMS is the Triangle Architecture in Figure 1.

Typically, it consists of metadata server and resource manager(s).

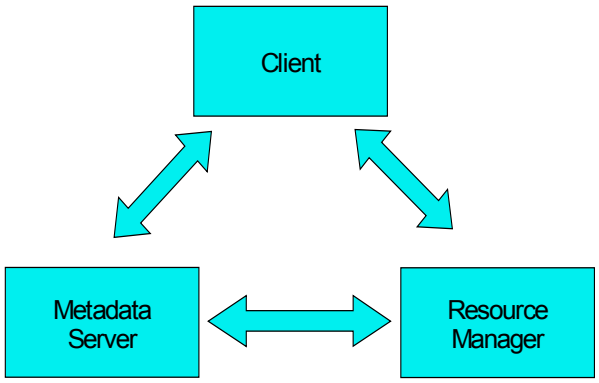

Figure 1:The Triangle Architecture

The metadata server contains all the index information for all the objects in the resource manager. This meta information helps in management issues like protection, authentication, easier search and storage management of the digital resources. Resource manager contains the actual objects like the images, documents or the video clips and serves the content to the client. The metadata server is typically an RDBMS. The resource manager in a number of scenarios is a file system with the resources in file formats.

The flow of information in Figure 1 is as follows

- Client submits a query to the metadata server. The metadata server serves the request if the client provides the necessary authentication information.
- Now there are two possibilities for retrieving the objects from the resource manager
  1. metadata server tells the resource manager to push the object to the client
  2. Client gets the object locator information from the metadata server and extracts the object from the resource manager For example in IBM's digital library product, Content Manager, option 1 is being followed.

This paper will focus on the following functions of Content Management Systems:

- Update Support - This involves updating the objects in the resource manager, like creating, updating or deleting documents,

and be able to maintain consistency with respect to the metadata server. This brings in the questions of integrity, distributed transaction management and coordinated recovery between the resource manager and the metadata server. For example, creating and deleting a document and the corresponding metadata should happen as a single unit of work to avoid problem of dangling references. Similarly, update to the resource and metadata should be synchronized. The question of synchronized backup and recovery becomes important if we want to recover the versions of resources pointed to by the snapshot of the metadata server.

- Access control - Access control refers to the level of access to the resources in the resource manager. Client could be granted access to the resources based on rules built into the metadata rather than relying entirely on the security provided by a resource manager. This issue becomes particularly relevant when a client is pulling objects from the resource manager rather than resource manager pushing objects to the client. For example, in a web environment, digital libraries could adapt so that clients are able to request for objects from the resource managers.

Till date, many Content Management Systems have either not addressed the above requirements or have built custom solutions which handle the issues to a varying degree. For example in IBM Content Manager, synchronized backup and recovery are not supported in the current release.

To our knowledge, none of the problems have been addressed using the RDBMS. All the capabilities like recovery, integrity and access control are essentially available in a RDBMS and seems logical that RDBMS be enhanced to extend this functionality to manage the semi-structured data so that the user feels that semi-structured data is part of the RDBMS although it is actually residing outside the RDBMS. This way administrative control becomes easier and the metadata server and the resource manager need not be managed through separate solutions. By using the capabilities and advances in RDBMS technology, CMS vendors do not have to rely on closed solutions.

DataLinks is one such technology which helps extend the reach of a RDBMS to the outside data as if they were one single entity. DataLinks is an SQL standard [4] and could potentially be made available by any RDBMS vendor. This helps build a standard platform for the above mentioned requirements.

Section 2 describes the basics of the DataLinks Technology available in IBM DB2 UDB Universal DataBase (DB2 UDB). Section 3 describes how content management systems benefit from using the DataLinks technology. The subsequent section talks about future directions and we conclude in Section 5.

2. DATALINKS TECHNOLOGY IN THE IBM DB2 UNIVERSAL DATABASE

DataLinks is a technology invented at IBM Almaden Research Center. It is available in IBM DB2 Universal DataBase (DB2 UDB) and was first introduced in it's Version 5.2. It extends RDBMS to manage semi-structured or unstructured data residing outside the RDBMS. Traditional RDBMSs deal with structured data and provide integrity, recovery and access control for such data. DataLinks technology enables RDBMS to provide these capabilities for semi-structured data without needing to import it into the RDBMS. By storing references to semi-structured data in SQL tables in RDBMS, the above capabilities can be extended to the referenced data.

Semi-structured or unstructured data, such as, documents, images, video clips, e-mail messages, engineering drawings, presentations and other business formats, typically resides in file systems. Access performance, streaming requirements, proximity to client and popularity of file system API are some reasons why such data will continue to reside in file systems rather than being moved to databases [2]. However, these files are often related in some way to structured data stored in RDBMS. A typical example is photograph of a product stored in a file and its inventory information stored in the database. Since the data is managed by two distinct systems (RDBMS and file system), there is a need to synchronize updates and backup and recovery to avoid inconsistencies. For example, the file data may get updated without updating the corresponding database data, a file may be deleted without deleting the database data or the database data may be restored to an older point in time, while the file data continues to reflect the current point in time. Thus it is necessary to coordinate various functions between RDBMS and file system.

DataLinks uses distributed transaction management and recovery techniques to provide this coordination. It does not require applications to change the way they access files, that is, applications still continue to use the standard file system interface.

DataLinks consists of two main components: extensions to the DB2 UDB engine (hereafter called datalink engine) and DB2 UDB Data Links Manager (DLM). The datalink engine resides where the database is located. The Data Links Manager resides

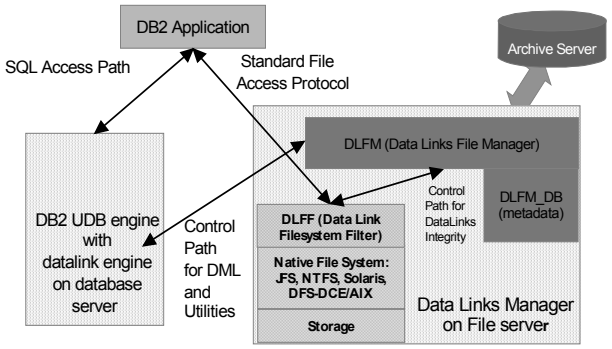

Figure 2: DataLinks Architecture

where the file system is located. Figure 2 shows the architecture of DataLinks in DB2 UDB.

The datalink engine provides a new SQL data type called DATALINK. A DATALINK value is a reference (in the form of a URL) to a file residing in a file system. A variety of options can be specified in the definition of a DATALINK column. These options determine the level of control exercised by the RDBMS over the files referenced by URLs stored in the column. Using these options it is possible to specify whether referential integrity

must be guaranteed for file references; whether read and write access to the referenced files should be controlled by the database or the file system; whether coordinated recovery of the database and the referenced files is to be supported by the system.

When reference to a file is stored in a DATALINK column (e.g. using SQL INSERT), the file is said to be 'linked'. The act of committing the transaction which creates a link to a file establishes control of the RDBMS over the file. As long as the file remains linked, the system enforces RDBMS control over the file as specified by the options in the DATALINK column definition. In order to enforce these controls, some actions are needed to be done on the file server. For example, the read and write access permissions or the owner of the file may need to be changed; the DLFM metadata may need to be updated to indicate that the file is linked. The datalink engine requests the DLM to perform these actions when linking a file. All these actions must be done in the context of the RDBMS transaction which created the link to ensure consistency. Therefore, transactions which manipulate DATALINK columns involve updating data on multiple file servers in addition to updating data in the database itself - essentially they are distributed transactions. DataLinks uses a two-phase commit protocol for these transactions. The datalink engine acts as a coordinator and the DLMs acts as participants in this protocol. Likewise, deleting the reference (e.g. using SQL DELETE) will cause the RDBMS to relinquish control over that file. The file is then said to be 'unlinked'. The action of unlinking is also done in the context of a distributed transaction.

The Data Links Manager itself comprises of two sub-components: Data Links File Manager (DLFM) and Data Links Filesystem Filter (DLFF). The DLFM executes various requests from datalink engine. It maintains the metadata like which files are linked to the database and what level of control is desired for each of these. The DLFF is a control layer on the top of the native file system. It intercepts some file system operations (for example, delete, rename, open) to help enforce desired controls by the RDBMS. However, it does not interfere with read/write path so that performance of these operations is not affected. The DLFM and the DLFF together enforce the level of control specified when defining the DATALINK column. For example, if a column definition specifies that referential integrity is required for linked files, the DLFF will disallow rename or delete operations on files linked to that column; if it specifies that read/write access to linked files is to be controlled by the database, the DLFF disallows read/write access unless a valid token obtained from the RDBMS (using an SQL SELECT) is used when accessing the file. The DLFF itself is stateless, it consults the DLFM to find whether a file is linked and the level of control to be enforced.

Recovery utilities are an important part of RDBMS products. DB2 UDB's backup utility takes "snapshot" of a database. The database can be restored to that snpashot state using the restore utility. DataLinks extends the ability of these utilities so that backup and restore of referenced files is also done along with the database data in a coordinated way. If the DATALINK column definition specifies that coordinated recovery is desired, the DLM initiates archival of a file when it is linked. The archiving operation is asynchronous with respect to the transaction that creates the link. The DLM thus keeps archive copies of files. When restore utility is used to reinstate a particular snapshot of the database, the corresponding versions of the referenced files will be reinstated too. This ensures that the file data is always consistent with the database data. The datalink engine and the DLMs coordinate with each other to achieve this functionality.

Files residing on multiple file servers can be referenced from a single database. Conversely, multiple databases can reference files on one file server. Additional details on DataLinks technology can be found in [3].

DataLinks technology extends SQL language definition. These extensions have been standardized in ISO as a part of SQL99. The standard is presently in the Final Draft International Standard (FDIS) state [4]. The final International Standard (IS) is expected to be published in early 2001.

DataLinks thus enables management of file data through standard SQL interface. It enhances recovery utilities in DB2 UDB to extend their reach to file data. Thus recovery of both database data and file data is coordinated through a single, common interface.

3. BENEFITS OF DATALINKS TO CONTENT MANAGEMENT SYSTEMS

In a Content Management System, maintaining consistency between metadata server and resource manager is difficult because of two problems:

1. Updates, such as creation, deletion of objects and modification of their content, must be done in a coordinated fashion in the context of a single distributed transaction
2. Recovery operations must be coordinated

Achieving (1) requires distributed transaction management and crash recovery techniques. Achieving (2) requires complex recovery techniques.

Commercially available Content Management Systems have either simply not addressed these problems or have built custom software on their own. However, with advances in database technology, it is possible to exploit RDBMS capabilities to address these problems.

DataLinks technology incorporates these capabilities in the DB2 UDB RDBMS product. It also provides the capability to implement uniform access control for metadata as well as files. Standard SQL mechanisms can thus be used to achieve access control using rules based on metadata. Architecture of DataLinks in DB2 UDB makes it particularly suitable as a basis for implementing the storage model required in Content Management Systems. The DB2 UDB engine will manage the metadata server. The Data Links Manager(s) residing on the resource manager(s) will cooperate with the DB2 UDB engine to provide integrity, recovery and access control. Figure 3 shows how DataLinks components fit in the triangle architecture. By using DataLinks in DB2 UDB, Content Management Systems can rely on DB2 UDB's robust capabilities for integrity, recovery and access control, rather than reinventing the wheel. The specific advantages achieved by using DataLinks are explained in detail in the

subsequent sections.

3.1 Integrity

Existence of metadata server and resource manager as two

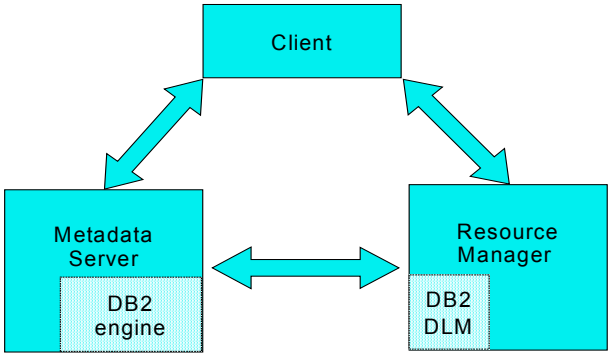

Figure 3:The Triangle Architecture with DataLinks components

independent components poses the difficult problem of maintaining integrity between these two. For example, without a distributed transaction management mechanism, files in the resource manager can be deleted or renamed, leaving dangling references from the metadata server; the data in these files can be modified making the index information in the metadata server invalid. Thus there is a need to coordinate updates to these independent entities.

DataLinks in DB2 UDB addresses these problems by providing both referential integrity and data integrity. Using SQL in DB2 UDB, the metadata information and the reference can be created in a single transaction. The action of creating file references in the metadata server also ensures the existence of these files and all this is done atomically in the context of a single transaction. Existence of the file in the resource manager is guaranteed as long as it is linked to the metadata server. The DLFF, in conjunction with the DLFM, disallows any file rename or delete operations which will violate referential integrity constraints. The deletion of the data in metadata server and deletion of the referenced file can also be done atomically, in a single transaction. DataLinks thus guarantees referential integrity between metadata server and resource manager.

Data integrity is achieved by allowing only controlled updates to linked files. An update operation of the file is not considered complete unless an SQL update is done to the reference in the metadata server. This provides a mechanism to atomically perform the two actions - (i) completing the update of the file and (ii) updating the index information in the metadata server. For example, a DATALINK column definition can specify that integrity is required for all references, read and write access is to be controlled by the database and that point-in-time recovery is required. The steps involved in updating a file linked to this column will be:

1. Use SQL SELECT to search and obtain a particular file reference from the database. This reference will have a "write token" embedded in it.
2. Use the file reference with the embedded token to open the file for writing. DataLinks can optionally block any readers on that file after the file is opened for write.
3. Use the file system API to update the file content.
4. Perform an SQL UPDATE to update the file reference with itself to declare the operation to be complete. The file is considered to be in "update-in-progress" state until this SQL UPDATE is done. Any meta information that needs to be updated in the database can also be updated in the same transaction. Thus a new version of the file and the corresponding meta information can both be created atomically in the same transaction.

Since the column definition specifies point-in-time recovery, the previous version of the file is maintained by the DLM. Therefore, changes to the content can instead be rolled back using slightly different SQL in Step 4 above.

Using DataLinks in DB2 UDB, Content Management Systems are relieved of implementing complex distributed transaction management techniques themselves. They can rely on DB2 UDB's robust transactional capabilities to achieve integrity.

3.2 Backup and Recovery

In response to any damage or corruption, the recovery of both metadata server and resource manager should be synchronized. Recovery of the metadata server data should automatically lead to the recovery of the referenced files at the various resource managers. This involves coordination of backup of metadata server database with the backup of the reference files at the resource managers. This also involves rebuilding of these data should they be damaged or corrupted in some way.

Current implementations of Content Management Systems either do not support coordinated backup and recovery of the metadata and data at the various resource managers or they support it in their own proprietary way without the feature being provided through a RDBMS. Our solution is to exploit coordinated backup and recovery feature built within an RDBMS so as to take full advantages of *single point of administration*.

When files on a resource manager are linked, they are scheduled for copying asynchronously to the transaction that links (inserts) them. The files are copied to an archive server (disk, IBM TSM etc).

Backup of the metadata database is coordinated with its resource managers to ensure that by the time a metadata server backup (online or off-line) is successfully completed, the corresponding files at all the resource managers have also been backed up.

Rebuilding of data is done through restoring from a previous database backup and reapplying changes as indicated in the database log. This also ensures that the files at the active resource managers are also restored to the proper versions as referred to by the metadata. Thus the metadata is synchronized with the files at resource managers.

3.3 Access Control

Different Content Management Systems have different mechanisms to provide access control to files residing in the resource manager. One example, as given in Section 1, is IBM Content Manager, in which all data requests are sent to a central metadata server where the access control is performed. The request is then forwarded to the resource manager and finally the data is delivered to the clients on a given port. This provides a centralized solution to simplify administration. However, this model may not be suitable in other situations, where users want to control when and how the data is delivered. Each access to the resource manager requires sending a request to the metadata server. Sometimes, due to the network condition, users may want to avoid multiple trips to the metadata server for any repeat access to files in the resource manager.

Another approach is to allow direct access to the resource manager. This model provides the flexibility of when and how users access files in the resource manager. However, this usually requires building another layer on top of the native system to check against the access model in the metadata server because the access model provided by the native system cannot be directly mapped to the one in the metadata server. For example, the access model used in RDBMS can be defined through the use of SQL View. The definition of SQL View can contain arbitrary predicate. This provides more flexibility. On the other hand, the access model used in the file system is typically based on an Access Control List (ACL) associated with a file. It is less flexible because the access model is changed only when the ACL is altered.

With DataLinks technology, the access model that is defined in the RDBMS can be extended and applied to files residing in the resource manager. This is achieved through the token mechanism. Unless a user has authority to fetch the DATALINK value from the table, no token will be generated. The token becomes a ticket to carry the authority for accessing a file that is referenced by a DATALINK value. Access to the file (either read or write) is granted only when a valid token is provided. The token is generated at runtime when querying a DATALINK column from an SQL table. It is embedded inside the returned name of the DATALINK value.

Suppose the file name without a token is

```
HTTP://XYZ.COM/image/image001.gif
```

the DATALINK value returned from the table will be (including the token)

```
HTTP://XYZ.COM/image/8QXzDf2fDSQHsU;image001
.gif
```

Instead of providing the filename in the file open call, a user supplies the whole name including the token. The DLFF (described in section 2) will intercept the open call and validate the token. If the token is valid, the DLFF will let the open call proceed.

Therefore, the access model defined in the RDBMS for controlling access to the metadata can be used to control access to files in the resource manager as well. No extra access control mechanism is required in the resource manager. Files can be accessed through any native file system interface or protocol. New features provided by the native system can be easily exploited by the applications. Moreover, assuming the communication channel is secure, by passing the token, the authorized user can designate another user to access the file. This user does not need to be registered to the system. It is useful in the Web environment where connection is ad hoc and session-less.

Apart from using the RDBMS control via the token mechanism, while defining a DATALINK column, other levels of access control are also available, such as retaining the access model defined in the file system. In this case, a plain filename will be returned in a query against the DATALINK column. The access model will follow what is defined for the file in the file system.

3.4 Replication

Access performance may be a concern when dealing with metadata and files in the content management system. This can happen when files are big and the number of users is large. By incrementally replicating both metadata and files in the content management system to multiple instances, load balancing is achieved.

Replication not only helps in the load balancing situation, but also in other areas, such as providing a hot standby system and enabling data sharing among different physical locations and business partners. Nevertheless, most of the replication technology is available only to the metadata residing in the RDBMS. If users want to replicate both data in the metadata server and files in the resource manager, an external mechanism is required to be implemented on the top of whatever provided by the RDBMS. However, with DataLinks Replication technology, the replication of the metadata and the external referenced files can be handled by the RDBMS in a transactional fashion. The technology is built into the DB2 data replication product, called DB2 DataPropagator.

DB2 DataPropagator uses the database log to capture changes to any registered SQL table or view in the source RDBMS and then externalize the changes in a staging table. When the replication cycle is triggered (either by an event or a timer), the staging table will be examined to locate the committed changes and then propagate them to the subscribed table in the target RDBMS. For DATALINK value, before the metadata is propagated to the table, the referenced file gets copied first from the source file system to the target file system. The new file reference is used when propagating the metadata to the target table. As a result, a DATALINK reference is established to the file which is just copied to the target RDBMS.

By using DataLinks, no external mechanism is needed to handle the replication of external files in the resource manager separately. Both data in the metadata server and files in the resource manager can be replicated to the target system in a synchronized and consistent way.

4. FUTURE WORK

There are many other useful features for the Digital Library which can be considered for the future release of the DataLinks

Technology. One of them is to support other nontraditional object storage systems. DataLinks currently supports several popular file systems, such as AIX JFS, NFS, Solaris UFS, DCE/DFS and NTFS. The support can be extended to other storage systems, such as Hierarchical Storage Management (HSM) system and video streaming server. Moreover, the interface to the DataLinks File Manager can be externalized to allow other vendors to integrate their object storage servers with DataLinks. In such case, the DATALINK value no longer necessarily refers to a physical location of a file, but rather provides a logical reference to an object. Another enhancement is to provide object creation support through SQL. With the existing DataLinks technology, users can do modification and deletion of the files within the SQL scope. However, a file still needs to be created before it can be linked to a DATALINK column. A function can be added to allow creation of the file during the linking process if the file does not exist. Therefore, the whole life cycle of files can be totally captured under the control of the DataLinks.

5. CONCLUSIONS

Access control, integrity and recovery are essential functions for managing content in Digital Libraries. Due to the lack of these functions for semi-structured and unstructured data in RDBMS, Digital Library products had to build custom solutions.

This paper described how DataLinks technology provides each of these functions inside RDBMS. We believe that Digital Libraries can greatly benefit from synergistic integration with these advanced RDBMS functions. This helps avoid building and maintaining custom solutions for above mentioned features. Moreover, by relying on standard RDBMS functions, they can also leverage other DataLinks features like replication.

6. ACKNOWLEDGMENTS

The authors would like to thank Hui-I Hsiao and Inderpal Narang for their valuable comments during the preparation of this paper.
The authors would also like to thank all the people who have contributed to DataLinks technology over the last few years.

7. REFERENCES

[1] DB2 Magazine, "Controlling the Digital Deluge: IBM Content Manager", Blaine Lucyk, summer 2000.
[2] IBM, “DataLinks: Managing External Data With DB2 Universal Database”, White paper prepared by Judith R. Davis for IBM Corporation, February 1999.
[3] H. Hsiao and I. Narang, “DLFM: A Transactional Resource Manager”, Proc. ACM SIGMOD Conf., Dallas, Texas, May 14-19, 2000.
[4] ISO/IEC 9075-9:2000, “Information technology - Database Languages - SQL - Part 9: Management of External Data (SQL/MED)”
[5] D. M. Choy, C. Dwork, et al, "A Digital Library System for Periodicals Distribution", Forum on Advances in Digital Libraries - ADL'96
[6] M. Papiani, J. Weson, A. Dunlop, and D. Nicole, “A distributed Scientific Archive Using the Web, XML and SQL/MED”, ACM SIGMOD Record, Vol. 28, No. 3, September 1999.
[7] IBM, http://www.almaden.ibm.com/cs/datalinks