JITTA
JOURNAL OF INFORMATION TECHNOLOGY THEORY AND APPLICATION

OBJECT-ORIENTED DATABASE MANAGEMENT SYSTEMS: ARCHITECTURE AND APPLICATION

MAHESH S. RAISINGHANI, University of Dallas
Email: mraising@gsm.udallas.edu

GABRIEL CUSTODIO, Associates Information Services, Inc.

ABSTRACT

The advent of multimedia computing, the World Wide Web, and object-oriented application languages has caused the proliferation of complex data types that must be managed differently from traditional character or numeric data types. Relational DBMS (RDBMS) can be modified with data extenders to support these complex new data types. Object DBMS (OODBMS), however, are designed specifically for these data types, and manipulate them with far greater efficiency. OODBMS perform direct navigation, clustering, schema evolution, and other functions that RDBMS cannot match. Adoption of common interoperable standards will facilitate the move towards open systems for heterogeneous, distributed platforms.

INTRODUCTION

Consider the following scenario. You are designing a web-enabled application for a client who sells designer clothing. The client wants potential customers not just to browse the catalog on their web site, but to have an experience that emulates a visit to a designer's studio. Though there may be only 50 to 100 items in the catalog, these items must be presented real time in a slick, seamless fashion. The specifications are:

Visitors to the site may browse through thumbnail indexes of designs in the catalog.

A search engine must be available to help visitors locate clothing by type.

When an item is selected, the visitor may enter a profile to appropriately shape the item for a 3D presentation.

The design will be presented in full-color 3D. The visitor may rotate the image a full 360 degrees, and may

Raisinghani, M., and G. Custodio, "Object-Oriented Database Management Systems: Architecture and Application", *The Journal of Information Technology Theory and Application (JITTA),* 2:1, 2000, 11-18.

pitch or skew the image to view it from several different angles.

The color, shade, and material may be selectively changed, and the simulated illumination in which the design is displayed may be adjusted to several different settings.

The visitor has the option of hearing a voice commentary that describes features and aspects of the design.

Accessories that complement the design may be selected from a separate bar.

A running total of the bill, with any applicable discounts, is available upon the visitor's request.

The visitor may then elect to conclude with a purchase transaction, or be routed via e-mail or direct phone to the designer's sales assistant immediately

A session normally lasts no longer than 20 minutes, but in that time the visitor may choose from a large combination of custom views and details. Even with this mix of flexible access and complex data (image, voice, and animation) combined with some data and information gathering, the visitor does not require special technology other than a browser. This requires a database that moves dynamically and interactively with the visitor's requests. The database management system is actively engaged in the entire selling process.

OBJECT-ORIENTED DBMS

The database management system required for the application described above must provide static images, video, 3D graphics, voice, music, and traditional data to provide customized information and ambiance at the visitor's request. This web site, with all its complexity and sophistication, is envisioned for electronic commerce. Web pages that constitute a web site are essentially complex objects containing many smaller, interrelated objects. The image files, audio files, hyperlinks, and other information that comprise a web page are all objects.

The advent of multimedia computing, the World Wide Web, and object-oriented application languages such as C++ and Java has added numerous complex data types that must be managed differently from traditional alphanumeric data types. Vendors of Relational DBMS such as Informix and Oracle offer data extenders to support these data types. But the addition of data extenders does not result in pure object databases, which falls under the purview of object-oriented DBMS.

Architecture Overview

Rather than present the user with a series of predefined, static HTML pages, web-enabled (object-oriented) applications are designed to compose pages in real time from a large collection of component objects stored in several different databases. Component objects can be selected and configured at runtime, allowing the site to be customized based on individual user preferences.

The object system consists of a collection of CORBA (Component Object Request Broker Architecture) servers that process incoming web requests and return information for presentation back to the user. A web server translates these standard or custom HTTP requests into invocations on objects in the CORBA servers, then collects the returned information and compiles it into applets for transmission back to the user's interface. These applets are cached in the web server's memory to reduce the amount of inbound traffic and decrease request response time, and are known in the object community as "Servlets?

Figure 1 represents a high level view of an object-oriented system architecture:

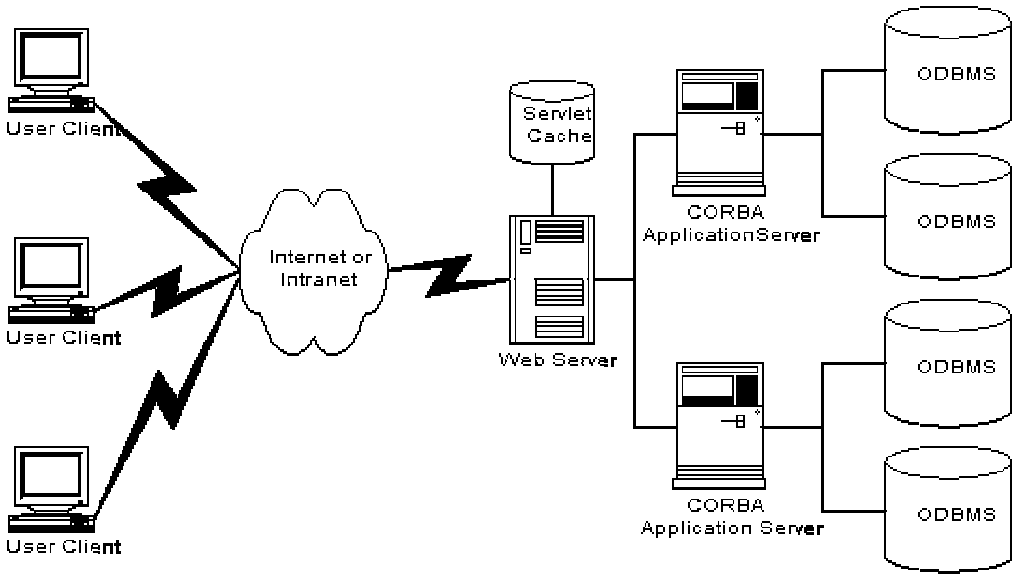

Figure 1. Object-oriented System Architecture

Application & Database Overview

The key feature of this object architecture is that every discrete element of a dynamic HTML page is modeled in the back-end database as a C++ or Java Object. A single object is used to present a site index that allows users to navigate through the site. This site index changes its appearance dynamically based on the structure of the back-end database. The key advantage of using objects to represent page elements is that these objects can encapsulate algorithms as well as data, so that intelligence and dynamic behavior can be built into site components.

Object systems are those that contain little or no fixed data, but rather interact at runtime with a comprehensive set of text objects such as write-ups and articles, in combination with non-text objects such as video and images, that correspond to particular site content. When the web server processes a request, a dynamic HTML page is constructed by assembling the set of objects that compose the requested page. These objects may contain other objects and each of these sub-components may have an internal composition as well. These objects may not necessarily reside in the same database, and may even be stored by different servers in different geographical locations.

Figure 2 illustrates how an object-based HTML page is mapped onto a set of objects in the database:

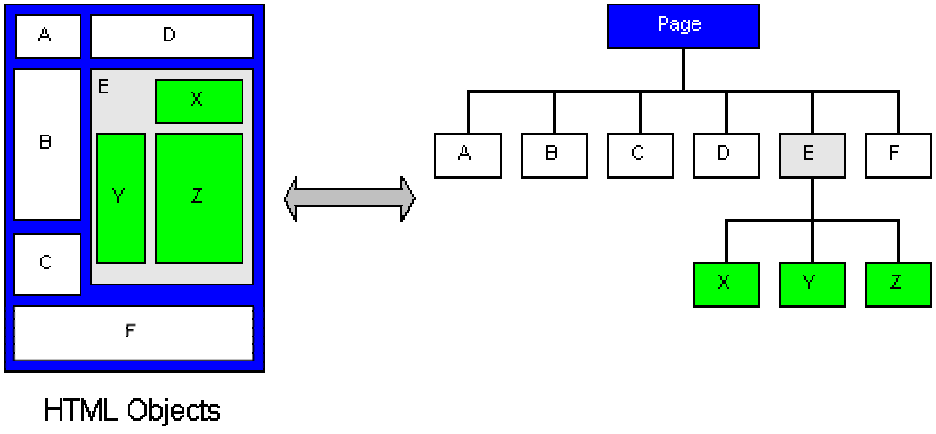

Figure 2. Object-based HTML Page Mapping

To store objects and programs based on object models, a relational system (RDBMS) will have to break up the class hierarchies and relationships into tables. Then it will have to use the processing-intensive join operation to reassemble the tables into objects for use by object-oriented applications. It is impossible for relational databases or any flat file databases to circumvent this process. This will grow more daunting as programmers and application developers build increasingly complex objects. On the other hand, due to its itemized nature, object databases can perform the following functions that are not quite possible with relational databases:

Direct Navigation Pure object databases store maps of object class hierarchies and relationships, which are directly accessed for information. Since navigation to the data is direct, it is the most efficient approach in storing and retrieving object-oriented information.

Clustering Object databases can perform physical object clustering to tune system performance. Objects that are used most often are stored in cached memory space, which improves application performance dramatically. Depending upon the database, the clustering can be quite fine-grained, which minimizes network traffic. Data can be sent either in a very large block in a single transmission or in a small set of objects. Clustering also enables an object database to efficiently access all of the objects that would allow multiple users and applications to each have a unique view of the underlying object model.

Data Distribution The key to this concept is a unique object identifier that isolates the logical name of an object from its physical location. This enables a single logical database to be separated and run on multiple processors. This can occur in the same machine (symmetric multiprocessing) or in a network of machines (cluster). This fine granularity of object management enables efficient data replication, which allows a network of processors to manage an application. In addition, the database is segmented into discrete components that are kept synchronized. This effectively supports the distributed form of system architecture.

Locking Locking is the capability to maintain orderly change in the state of the database by controlling read and write privileges to the database. Object databases can lock object-oriented data at the smallest object level. Users without update privileges would only be prohibited from changing a small portion of the database while they can generally work with the rest of the data while a specified user has the lock. This minimizes concurrency conflicts between users. Object databases can even lock at a logical object

level and have the system take care of locking all the related physical objects without any programming. This is called dynamic locking.

Schema Evolution Changes in the object model necessitate change in the object relationships held by the object database. Object databases have facilities to enable applications developers to change the database schema without requiring the database to be brought off-line and changed in a large batch cycle, as a relational database or any other type of database would require. Schema evolution performed online in an intelligent incremental manner can gracefully introduce system changes. In addition, many object databases keep track of different versions of the object model and match the application program with the appropriate version.

Client/Server Architecture Object systems are designed to have aspects of database processing done on clients in cooperation with activities on servers. The client functionality in an object database can be extensive, including the capability to access multiple databases and assemble the returned objects to create a unified local view. Multithreading enables a client to support multiple database tasks simultaneously. The distributed nature of many object databases is evident in their ability to maintain local object caches without tying up the central server and, most importantly, the network. Accordingly, object databases can be quite scalable.

N–Tier Architecture In this architecture, numerous systems send messages to each other. These systems normally have clients, application servers, web servers, and database servers working together as a single system to fulfill any user request. Depending on the database and application, an object database could be distributed in multiple locations. By virtue of their support for distributed computing, it will be easier for OODBMS to support many new applications as system architectures change.

Program Execution The ability to run an application in the database is one of the emerging functionality requirements for enterprise object computing. One of the primary reasons is the need to support multiple client requests for execution of a segment of centralized code, while providing critical integrity and transaction management features. Databases, by virtue of their multi-user functionality, have many of the facilities to support such application processing. Another use of database program execution is running queries in a database, which can reduce the amount of information that has to be sorted through.

Application Case Study

Despite this array of impressive features, Object DBMS has yet to reach the full acceptance of the computer systems community. While many major businesses are considering or have considered object-oriented systems, pure object databases have yet to make their full impact. This may be due to the difficulty and cost of migration from existing legacy or relational databases to purely object-oriented databases. Also, third-party vendors and seasoned DB veterans are now offering data converters to adopt existing data architectures to object-oriented applications. Oracle, Informix, and Sybase have introduced data adapters to the latest versions of their products to store complex objects and integrate with object-oriented applications. Third-party vendors such as Visigenic Software and IONA Technologies have also developed data managers based on ORBs (object request brokers) to enable integration.

Figure 3 depicts an example of a present-day object-oriented system utilizing existing legacy and relational database systems:

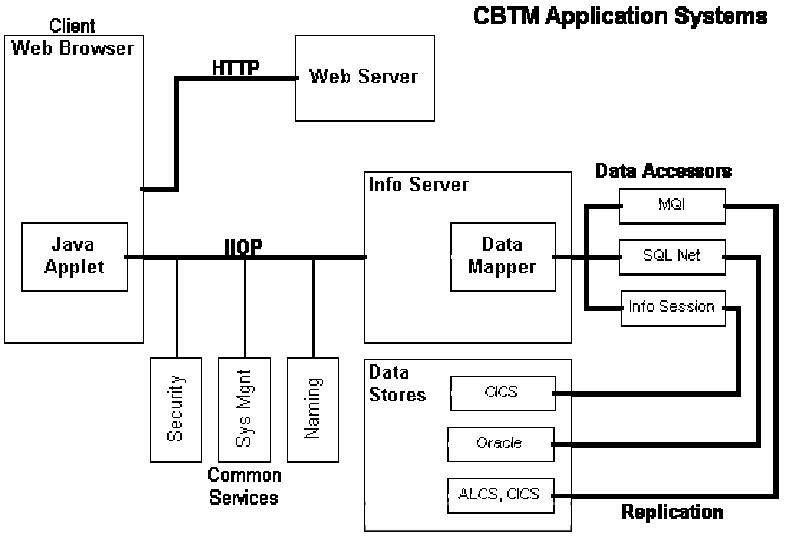

Figure 3. Object-oriented System Using RDBMS

This system is currently under development for the Consumer Branch Technology Migration (CBTM) project for the Associates First Capital Corporation (AFCC). The entire project is a joint development operation between AFCC, EDS, Sun Microsystems, and Oracle, as well as other third-party vendors.

The clients download Java applets from web servers via HTTP. The Java applets talk to applications located in information servers via an application framework, which is based on IIOP and written in Java. The applications access data via accessors provided by the data mapper located in the information server. These data accessors access data stores such as Oracle, CICS, and ALCS. All technologies selected were chosen to provide plug and play functionality for future additions to the application framework.

The physical network architecture is represented in Figure 4:

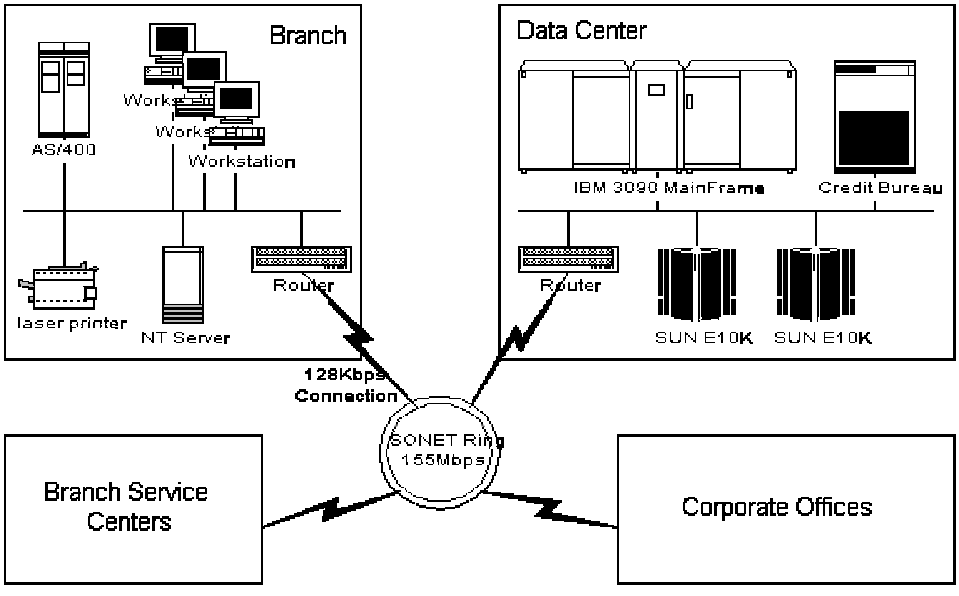

Figure 4. Physical Network Architecture of the CBTM System

OUTLOOK FOR THE FUTURE

Object databases are expected to gain a meaningful level of market credibility in the next one to two years. Currently, the object database market is being fueled by technically knowledgeable development organizations building real production systems. Confidence in the growth potential of object databases stems from the large size of the database market and the shift to object technology. Based on technological history, this trend is irreversible. It is expected that the object database market could experience 50-100% growth for a number of years. Current vendors generated over 50% growth in 1996, although from a small market base. The most promising aspect is the size of the overall database market. Even if object databases comprise only about 15% of the overall database market in three to five years, this would reflect a $1 billion market.

Use of the Object Management Groups's (OMG's), Unified Modeling Language (UML) as a standard language to analyze, model, and design business objects will promote the use of OODBMS. Also, the OMG's Meta Object Facility standard for distributed repositories provides the framework for implementing interoperable, heterogeneous, and/or multi-enterprise database/data warehouse solutions.

The lack of standards, the most widely known nemesis of object-oriented systems, has been recently addressed by the introduction of ODMG-93. ODMG-93 addresses the boundaries between the ODBMS and certain object-oriented programming languages. The programmer should perceive the binding as a single language for expressing both database and programming operations.

This specification recognizes two types of objects: transient and persistent. *Transient* objects are stored in and managed by the runtime system, be it Smalltalk, Java, or C++. Transient objects are what programmers have been using for years; they terminate when the program does. *Persistent* objects are managed by the OODBMS. As long as a root object in the OODBMS ultimately references an object, it is retained. From the programmer's perspective, however, persistence is transparent.

An object definition language (ODL) defines the OODBMS schema. Similarly, an object manipulation language (OML) is used to manipulate objects, and it doesn't distinguish between persistent and transient. The object query language (OQL) follows SQL-92 but offers object-oriented extensions and integration with any of the programming languages that have ODMG-93 binding. Relationships between objects in this specification are treated just as in its relational counterpart: one to one, one to many, or many to many.

ODMG-93 seems to promise that up-front work in database design will reap tremendous downstream benefits. Database design tools that follow this model generate code for the target application development environment, rather than code for the target database environment. Because the schema definition will be done in the application language, the developer need never be exposed to the underlying OODBMS.

CONCLUSION

Businesses increasingly require high performance access to complex data, and object databases provide superior performance and scalability compared to relational database alternatives. The World Wide Web has highlighted the reality that many businesses have complex structured data and unstructured text, email, reports, graphics, images, and audio and video resources that need to be seamlessly integrated. OODBMS can pave the way for organizations to reuse vital information, enable greater accessibility to corporate information resources, and create new applications and knowledge management opportunities.

It is unlikely that object databases will become the de facto standard overnight. Data adapters that integrate current data architectures to object-oriented systems extend the functionality of relational databases and thus delay the necessity for replacements. Adoption of common interoperable standards will facilitate the move towards open systems for heterogeneous, distributed platforms.

REFERENCES

Interview with Wayne Proctor, Systems Architect CBTM Project, AFCC

Interview with Paul Rogers, Senior Systems Designer, Sun Microsystems ?Enterprise Systems Planning

Interview with Jim McGuinness, Senior Systems Developer, Electronic Data Systems ?CBTM Project, AFCC

?u>Object DBMS: Now or Never? DBMS July 1997

Bloom, Paul I. ?u>Object Databases versus Universal Servers: Reality and Myth? ODI, White Papers

"Java Data Management ?Quantum Objects Implementations? ODI, White Papers

"Reevaluating Distributed Objects? DBMS January 1997

AUTHORS

Mahesh S. Raisinghani is a faculty member and co-director of the E-Commerce MBA Program at the University of Dallas?Graduate School of Management. He also serves as the director of research at the University of Dallas?Center for Applied Information Technology. His primary areas of expertise are Electronic Commerce Technology and Management, Strategic Utilization and Management of Information Systems, and the Organizational Impacts of Emerging Technologies. He is the chair of the Electronic Commerce track and a world representative for the International Resources Management Association, an active member of the Association of Information Systems, Decision Sciences Institute, and the International Association of Computer Information Systems. He has had numerous listings including the *Who's Who in Information Systems, Who's Who in the World* and *Who's Who Among Students in American Universities and Colleges*. Dr. Raisinghani conducts seminars in e-commerce and global information systems for executives.

Professor Raisinghani's previous publications have appeared in the Journal of Information Systems Management, International Journal of Information Management, International Journal of Materials and Product Technology, Journal of Electronic Commerce, Journal of Information Technology Theory and Application, Industrial Management and Data Systems, Electronic Commerce World, American Business Review, Minority Business News USA and Arthur Anderson's KnowledgeSpace. His chapters have been published in Annals of Cases in Information Technology Management, Managing Web-Enabled Technologies Application and Management: A Global Perspective, and Health Care Information Systems: Challenges of the Next Millennium; and has proceedings published in several regional, national and international information systems conferences in Australia, Canada, Greece, Israel, Mexico, Puerto Rico, South America and the U.S. His seven years of professional experience in information systems has taken him to Canada, China, India, Japan, Mexico, Singapore, Thailand, and U.K.

Gabriel T. Custodio. Is currently the Director of Computing Environment for the Business Technology Migration project of the Associates Information Services, Inc. His main assignment in this position is to design and build the infrastructure in which new systems are effectively developed and implemented. He received an MBA in Information Systems with emphasis in Application Development from the University of Dallas in April of 1999, and is currently pursuing a Masters of Management degree in Electronic Commerce from the same University. His diverse work experience ranges from Marine Transportation and Shipping Management, Oil Exploration, Retail Operations, Real Estate, and Marketing Management.