

# **Components of a Distributed Database**

Carolyn Mitchell

Technical Report #NSUCS-2004-005

Norfolk State University  
Department of Computer Science

**Abstract:** The purpose of the paper is to examine the underlying components within distributed database architecture. Understanding the task of a distributed database management system will lead to a successful design, which will provide scalability and accessibility. When developing a distributed database system there is a need to address the importance of security issues that may arise and possibly compromise the integrity of the system. We further propose solutions for some of the security concerns that pertain to a distributed database system.

## 1. INTRODUCTION

Today's business environment has an increasing need for distributed database and client/server applications as the desire for reliable, scalable and accessible information is steadily rising. Distributed database systems provide an improvement on communication and data processing due to its data distribution throughout different network sites. Not only is data access faster, but a single-point of failure is less likely to occur, and it provides local control of data for users. However, there is some complexity when attempting to manage and control distributed database systems.

## 2. OVERVIEW OF DATABASE SYSTEMS

For general purposes a *database* is a collection of data that is stored and maintained at one central location. A database is controlled by a *database management system*. The user interacts with the database management system in order to utilize the database and transform data into information. Furthermore, a database offers many advantages compared to a simple file system with regard to speed, accuracy, and accessibility such as: shared access, minimal redundancy, data consistency, data integrity, and controlled access [1]. All of these aspects are enforced by a database management system. Among these things let's review some of the many different types of databases.

Dr. Edgar F. Codd designed a relational model to solve pre-existing model problems while at IBM in the late 1960's. This relational model was built on mathematical principles, which he expounded upon in a book entitled "A Relational Model of Data for Large Shared Databanks [2]." A relational database is a set of tables (also called relations) that are separated into predefined categories. Each table contains *records*, which are the horizontal rows that contain one related group of data. The vertical columns are known as the *attributes*. "Data that is stored on two or more tables establishes a "link" between the tables based on one or more field values common in both tables [3]." A relational database uses a standard user and application program interface

called Structure Query Language (SQL). This program language uses statements to access and retrieve queries from the database. Relational databases are the most commonly used due to the reasonable ease of creating and accessing information as well as extending new data categories.

When dealing with intricate data or complex relationships, object databases are more commonly used. Object databases in contrast to relational databases store objects rather than data such as integers, strings, or real numbers. Each object consists of attributes, which define the characteristics of an object. Objects also contain *methods* that define the behavior of an object (also known as *procedures* and *functions*). When storing data in an object database there are two main types of methods, one technique labels each object with a unique ID. Every unique ID is defined in a subclass of its own base class, where inheritance is used to determine attributes. A second method is utilizing virtual memory mapping for object storage and management. Advantages of object databases with regard to relational databases allow more concurrency control, a decrease in paging, and easy navigation. However, there are some disadvantages of object databases compared to relational databases such as: less effective with simple data and relationships, slow access speed, and the fact that relational databases provide suitable standards oppose to those for object database systems [4].

Hierarchical databases are organized in a tree like structure where tables act as the root of the database with other tables branching out. “Relationships in such a system are thought of in terms of children and parents, such that a child may only have one parent but a parent can have multiple children.” Parents and children are connected by links called “pointers,” where a parent may have many pointers to each child. This relationship assumes that data is accessible for the user. On the other hand, hierarchical database systems are complex to use and require application developers to program routing through the linked records. In a hierarchical database all possible access points must be predetermined and followed accordingly for a successful database, otherwise access patterns not included can be extremely difficult to implement [2].

On the other hand, network databases alleviate some of the problem incorporated with hierarchical databases such as data redundancy. “The network model represents the data in the form of a network of records and sets which are related to each other, forming a network of links [5].” The relationships are represented in terms of records, record types and sets rather than hierarchy. *Records* are sets of related data values, which are equivalent to rows in a relational database model. *Record types* are a set of records, and set types are relationships of one or more record types. The network model actually resembles the hierarchical model only that it utilizes more theory for children tables to have more than one parent, allowing the network model to have a many-to-many relationship. Unfortunately, the network model is far more difficult to implement and maintain than what was needed by real end users to solve real problems [2].

### **3. DISTRIBUTED DATABASES**

In short a *distributed database* is a collection of databases that can be stored at different computer network sites. Each database may involve different database management systems and different architectures that distribute the execution of transactions [6]. The objective of a distributed database management system (DDBMS) is to control the management of a distributed database (DDB) in such a way that it appears to the user as a centralized database [7].

Providing the appearance of a centralized database system is one of the many objectives of a distributed database system. Such an image is accomplished by using the following transparencies: Location Transparency, Performance Transparency, Copy Transparency, Transaction Transparency, Transaction Transparency, Fragment Transparency, Schema Change Transparency, and Local DBMS Transparency. These eight transparencies are believed to incorporate the desired functions of a distributed database system [8].

Other goals of a successful distributed database include free object naming. “Free object naming means that it allows different users the ability to access the same object

with different names, or different objects with the same internal name.” Thus, giving the user complete freedom in naming the objects while sharing data without naming conflicts. Concurrency control is another issue among database systems. “Concurrency control is the activity of coordinating concurrent accesses to a database in a multi-user database management system (DBMS).” There are a number of methods that provide concurrency control such as: Two phase locking, Time stamping, Multiversion timestamp, and optimistic nonlocking mechanisms. Some methods provide better concurrency control than others depending on the system [8].

#### **4. COMPONENTS OF A DISTRIBUTED DATABASE SYSTEM**

In this section we will examine the components of a distributed database system. One of the main components in a DDBMS is the *Database Manager*. “A Database Manager is software responsible for processing a segment of the distributed database. Another main component is the *User Request Interface*, which is usually a client program that acts as an interface to the *Distributed Transaction Manager*. “A *Distributed Transaction Manager* is a program that translates requests from the user and converts them into actionable requests for the database manager, which are typically distributed. A distributed database system is made of both the distributed transaction manager and the database manager. [7].” The components of a DDBMS are shown in the figure below.

## Distributed Database Architecture

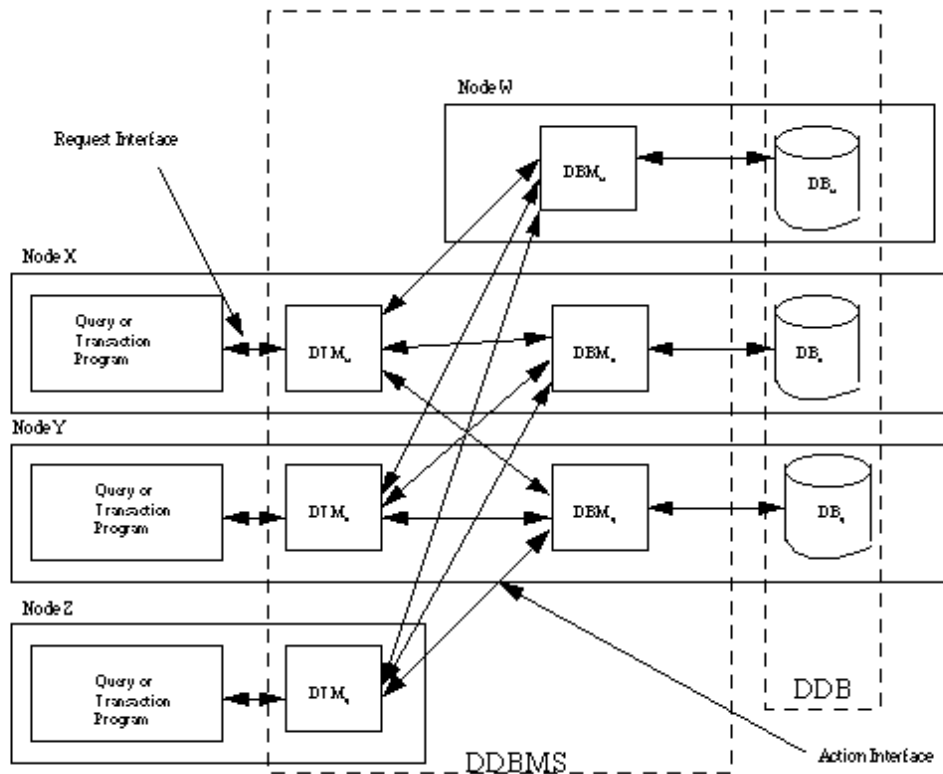


Figure 1 [6]

### 5. CLIENT APPLICATION

The User Request Interface is merely a client program running on one end system that requests and receives a service from a server program running on another end system. Due to the fact that the client and the server run on separate computers, by definition the client/server programs are considered distributed applications. There are two types of client/server applications. One particular client/server application is an *implementation* of a protocol standard defined in an RFC (request for comments). This type of application forces the client and server programs to abide by certain rules dictated by the RFC. Another sort of client/server application is a *proprietary* client/server application. “In this case the client and server programs do not necessarily conform to any existing RFC.” This allows the developer to have complete control over the code, allowing other developers to develop code that may not interoperate with the application.

In developing a proprietary client/server application the developer must decide whether to run the application over TCP or UDP. TCP connection oriented and ensures reliable byte-stream channel. However UDP is connectionless and forwards independent packets of data and does not guarantee delivery [9].

In designing a user request interface there is a need for web accessibility to ensure effectiveness, efficiency, and satisfying for people in different situations. Accessibility is concerned with making sites perceivable, operable, and understandable to the user. It is important that websites implement web accessibility because between fifteen and thirty percent of the general population has some form of limitation that may affect their ability to utilize technology products. Furthermore, there are law and guidelines for web accessibility such as the W3C Web Content Accessibility Guidelines 1.0 (WCAG) and the Section 508 of the Rehabilitation Act Amendments of 1998.

## **6. SECURITY ISSUES**

Many businesses are using distributed databases and with critical and sensitive amount of data being transferred across the network it is imperative that some form of security be implemented to secure the integrity and confidentiality of the system. General database security concerns must satisfy the following requirements: Physical integrity which is the protection from data loss; Logical integrity is the protection of the logical structure of the database; Elemental integrity is ensuring accurate data; Easy Availability; Access control to some degree depending on the sensitivity of the data; User authentication to ensure that a user is who they say they are. “The goal of these requirements is to guarantee that data stored in the DDBMS is protected from unauthorized modification, and inaccurate updates.” Some security threats involve: data tampering, eavesdropping and data theft, falsifying user identity, and administering too many passwords as well as others. Security can be provided for distributed databases by providing access control, user authentication, location transparency, and view transparency [1].

## **7. FUTURE WORK**

For future work I plan to examine more security issues for distributed databases. Also research more on how to implement some of the security measures that were listed above. Finally I want to track and record results as it could lead to future endeavors.

## BIBLIOGRAPHY

- [1] Charles P. Pfleeger and Shari Lawrence Pfleeger, Security in Computing, Prentice Hall Professional Technical Reference, Upper Saddle River, New Jersey, 2003.
- [2] Selena Sol, “Databases,”  
[http://wdv1.internet.com/Authorizing/DB/Intro/network\\_databases.html](http://wdv1.internet.com/Authorizing/DB/Intro/network_databases.html),  
May 24, 2004.
- [3] Tim McLellan, “What Is an Oracle Relational Database,”  
<http://www.islandnet.com/~tmc/html/articles/orareln.htm>, May 24, 2004.
- [4] “Object Oriented Databases,”  
<http://www.comptechdoc.org/independent/database/basicdb/dataobject.html>,  
May 25, 2004.
- [5] “Network Databases,”  
<http://wwwdb.web.cern.ch/wwwdb/aboutdbs/classification/network.html>,  
May 25, 2004.
- [6] Rick Noel, “Scale Up in Distributed Databases: A Key Design Goal for Distributed System,”  
[http://www.cs.rpi.edu/~noel/distr\\_scaleup/distriuted.html](http://www.cs.rpi.edu/~noel/distr_scaleup/distriuted.html), May 17, 2004.
- [7] Ovita Barretto, “Security Issues with Distributed Databases,”  
<http://students.depaul.edu/~obarratt/Security.htm>, May 20 ,2004.
- [8] “Components of a Distributed Database System,”  
<http://www.fi/~hhyotyni/latex/Final/node44.html>, May 24, 2004.
- [9] James F. Kurose and Keith W. Ross, Computer Networking: A Top-Down Approach Featuring the Internet, Pearson Education, Inc, New York, 2003.