Security, Reliability and Availability

Client/server is the combination of three major technologies: Relational database management systems (RDBMSs), networks, and client interfaces. Clients execute specific local tasks with local resources. Servers provide shared resources and fulfill broad tasks. Communication enables definition and completion of full work processes.

Client/Server software Infrastructure

With competing paradigms-SQL databases, TP monitors, groupware, and distributed objects-the middleware that connects clients to servers has grown dauntingly complex. Client /server is the combination of three major technologies: relational DBMS, networks, and client interface (usually GUI/PC based). Each element contributes to the overall platform with very specific roles but is independent of the others in performing its functions.

Advantages of the Client/Server Environment

The client/server software **architecture** is a versatile, message-based and modular infrastructure that is intended to improve usability, flexibility, interoperability, and scalability as compared to centralized, mainframe, time sharing computing. A client is defined as a requester of services and a server is defined as the provider of services. A single machine can be both a client and a server depending on the software configuration.

Client/server is an open system. The advantages of this environment include:

1. Interoperability,
2. Data integrity,
3. Scalability,
4. Accessibility,
5. Performance,
6. Security,
7. Adaptability, and
8. Affordability.

Client/Server Architectures

The client/server software architecture is a versatile, message-based modular infrastructure intended to improve usability, flexibility, interoperability, and scalability as compared to a centralized, mainframe, time-sharing computing. A client is defined as a requester of services and a server is defined as the provider of services. A single machine can be both a client and a server depending on the software configuration.

As a result of the limitations of file-sharing architectures, the client/server architecture emerged. This approach introduced a database server to replace the file server.

Distributed Client/Server Architectures

Distributed client/server systems serve environments with mixtures of heterogeneous computers and networks, with users and objects everywhere. Distributed Object Database Management Systems (ODBMSs) provide a mechanism for users to transparently access objects anywhere, with the efficiency of native access via caching. The current generation of distributed client/server computing allows information and resources and users to be located anywhere, with transparent access among all of them.

These software architectures are based on the ORB technology, but go further than the CORBA by using shared, reusable business models (not just objects) on an enterprisewide scale.

Administrator tools have grown with the production use of ODBMS products to include interactive and programmatic interfaces. These tool capabilities include:

· Moving objects and databases

· Dumping objects and databases to text interchange format and loading back

· Detaching databases from shared environments to portable ones

· Backing up online (which, of course, must be consistent across all databases in the distributed client/server environment)

· Querying and forcing locks

· Tuning and performance and usage statistics

Client/Server in Large Enterprise Environments

In large enterprise environments, the performance of a two-tier architecture client/server usually deteriorates as the number of online users increases. This is primarily due to the connection process of the DBMS server. The data language used to implement server procedures in SQL server type DBMSs is proprietary to each vendor. Oracle, Sybase, Informix, and IBM, for example, have implemented different language extensions for these functions.