

2001 S. Hanley, Suite 200 ●
St. Louis, MO 63144.1518 ●
Ph. 314.647.8880 ●
Fax 314.647.9434 ●
Free 800.231.4523 ●
www.intellipart.com ●

Client-Server Systems Versus File-Server Systems



White Paper

October 5, 2000

Introduction

There are two main approaches taken today by relational database management systems to support a shared central database for client workstation access. The first is a "file-server" approach in which the central server deals only with data files and is unaware of the application requesting a transfer of information. The second approach is referred to as the "client-server" model in which the server "understands" the nature of data requests and is optimized to perform those requests.

File-Server

The file-server based solution involves the use of a central networked file server (such as Windows NT) to provide logical disk drives (with shared files) for use by client machines. The file server has no knowledge of the logical requests being processed but acts as a remote disk drive using the network as an extended "bus" to transfer data to/from the client machine. All applications are run on the client creating a distributed computing environment in which the primary computing is performed entirely by the clients. Examples of these RDBMSs include FoxPro, DBase, Filepro, and Microsoft's Access.

Client-Server

In this model there is a central database server that processes logical database requests (searches, data insertion or deletion, etc.) sent by client machines. This central server understands both the nature of the request and the structure and location of the data while providing the main computational power of the RDBMS (the engine). Client machines provide the end-user with a Graphical User Interface (GUI) for applications interacting with the database. Clients use applications to create relatively short SQL commands (Structured Query Language) that are passed to the central RDBMS server; results are usually passed back as tables. Examples of client-server RDBMS include Oracle, MS SQL Server, Sybase, and Informix.

A file-server database also has inherent reliability limitations because it is maintained as a file in the file system, which can easily become damaged if either the client or server computer (or the connection

between them) fails during a transaction or other operation that writes to the database file. By isolating all database files under the control of a database server such as SQL Server, the client/server architecture can provide greater reliability and other advanced features that can't be furnished by the file-server architecture, such as the following:

Online backup

When you are using a database server, you can use an automatic scheduler to back up your database without having to exclude users from the database.

Durable transactions

Client-Server databases log transactions so that updates made within a transaction can always be recovered or rolled back to the last consistent state if either the client or the server computer fails. Although the Microsoft Jet database engine and .mdb files also provide transactions, the transactions in .mdb files aren't managed by a separate transaction log and can fail without recovery if the database file becomes damaged. A recent search on the web for "corrupt MS Access" returned over 53,000 hits.

Better reliability and data protection

If either a workstation or file server fails while an .mdb file is being written to, the database may be damaged. You can usually recover a damaged database by compacting and repairing the database, but you must have all users close the database before doing so. This rarely happens with a server database such as Microsoft SQL Server or Oracle.

Faster query processing

Using an .mdb file, regardless of where it is located, requires your solution to load the Jet database engine locally to process queries on the client. For large databases, this can involve moving a lot of data over the network. In contrast, SQL Server runs queries on the server, which is typically a much more powerful computer than the client workstations. Running queries on the server increases the load on the server more than would happen with an Access file-server solution, but it can reduce the network traffic substantially—especially if users are selecting a small subset of the data.

An Illustration

With five users on the system, the network came to a standstill. The reason was simple: in Microsoft Access, all database processing occurs on the local PC. Therefore, when the users issued complicated queries to the server, the network jammed with data being sent back to the local workstations. Often, the queries being issued from the applications required thousands of rows to be returned to the local PCs.

In the Microsoft Access server environment, this is the equivalent of calling a car dealership and asking how many convertibles they have in stock. To get the answer, the dealer drives every vehicle to your house and you count the number of convertibles. Obviously, this is not very efficient. In the Client/Server database computing environment (i.e. Microsoft SQL Server), a different approach is taken. Someone at the dealership counts the number of convertibles and passes the information back to the caller.

Conclusion

File-Server based database solutions are not appropriate for enterprise wide or departmental applications. The increased overhead placed on the network and the documented risk of file corruption mandate a Client-Server solution for these applications.