Anatomy of the Client/Server Model

In client/server architecture, clients, or programs that represent users who need services, and servers, or programs that provide services, are separate logical objects that communicate over a network to perform tasks together. A client makes a request for a service and receives a reply to that request; a server receives and processes a request, and sends back the required response.

Characteristics of Client/Server Architecture

- *Asymmetrical protocols*—there is a many-to-one relationship between clients and a server. Clients always initiate a dialog by requesting a service. Servers wait passively for requests from clients.
- *Encapsulation of services*—the server is a specialist: when given a message requesting a service, it determines how to get the job done. Servers can be upgraded without affecting clients as long as the published message interface used by both is unchanged.
- *Integrity*—the code and data for a server are centrally maintained, which results in cheaper maintenance and the protection of shared data integrity. At the same time, clients remain personal and independent.
- *Location transparency*—the server is a process that can reside on the same machine as a client or on a different machine across a network. Client/server software usually hides the location of a server from clients by redirecting service requests. A program can be a client, a server, or both.
- *Message-based exchanges*—clients and servers are loosely-coupled processes that can exchange service requests and replies using messages.
- *Modular, extensible design*—the modular design of a client/server application enables that application to be fault-tolerant. In a fault-tolerant system, failures may occur without causing a shutdown of the entire application. In a fault-tolerant client/server application, one or more servers may fail without stopping the whole system as long as the services offered on the failed servers are available on servers that are still active. Another advantage of modularity is that a client/server application can respond automatically to increasing or decreasing system loads by adding or shutting down one or more services or servers.
- *Platform independence*—the ideal client/server software is independent of hardware or operating system platforms, allowing you to mix client and server platforms. Clients and servers can be deployed on different hardware using different operating systems, optimizing the type of work each performs.
- *Reusable code*—service programs can be used on multiple servers.
- *Scalability*—client/server systems can be scaled horizontally or vertically. Horizontal scaling means adding or removing client workstations with only a slight performance impact. Vertical scaling means migrating to a larger and faster server machine or adding server machines.
- *Separation of Client/Server Functionality*—client/server is a relationship between processes running on the same or separate machines. A server process is a provider of services. A client is a consumer of services. Client/server provides a clean separation of functions.
- *Shared resources*—one server can provide services for many clients at the same time, and regulate their access to shared resources.

Differences Between 2-Tier and 3-Tier Client/Server Architectures

Every client/server application contains three functional units:

- Presentation logic or user interface (for example, ATM machines)
- Business logic (for example software that enables a customer to request an account balance)

Cookie Preferences | Ad Choices

IPR2025-00275
SportsCastr Inc. EX-2020

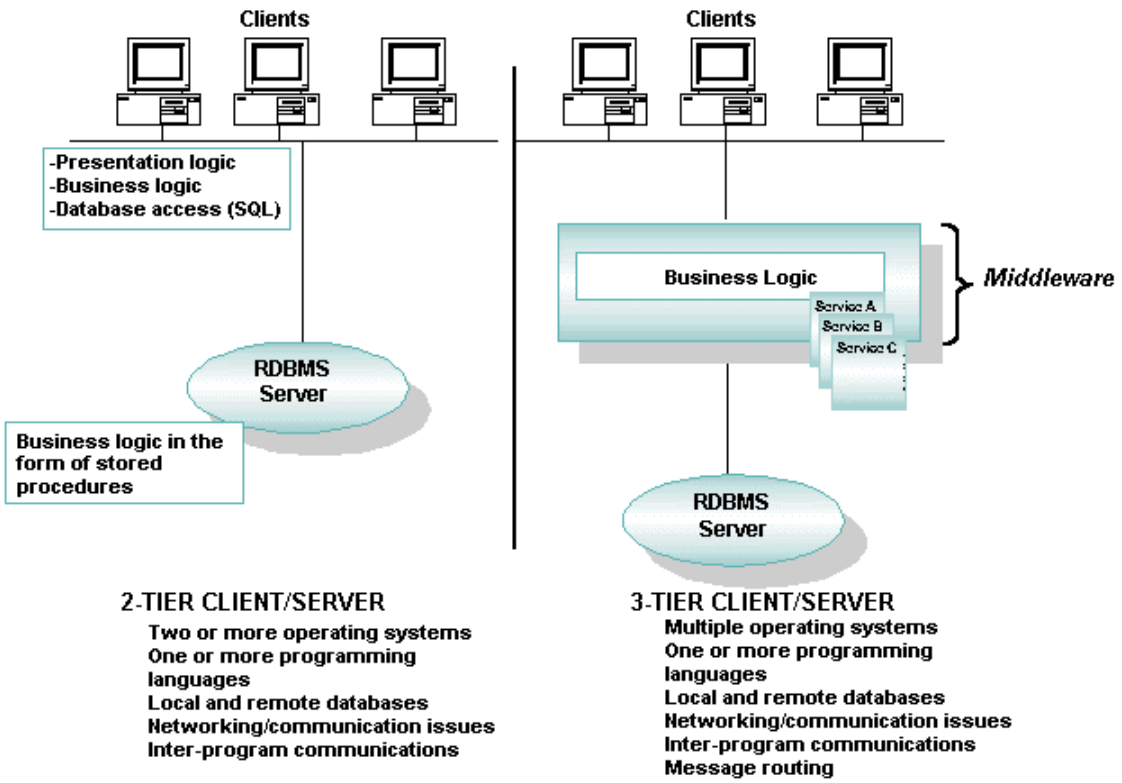
- Data (for example, records of customer accounts)

These functional units can reside on either the client or on one or more servers in your application. Which of the many possible variations you choose depends on how you split the application and which middleware you use to communicate between the tiers.

In 2-tier client/server applications, the business logic is buried inside the user interface on the client or within the database on the server in the form of stored procedures. Alternatively, the business logic can be divided between the client and server. File servers and database servers with stored procedures are examples of 2-tier architecture.

In 3-tier client/server applications, the business logic resides in the middle tier, separate from the data and user interface. In this way, processes can be managed and deployed separately from the user interface and the database. Also, 3-tier systems can integrate data from multiple sources.

2-Tier and 3-Tier Client/Server Models



Client/Server Variations to Suit Your Needs

Client/server architecture can accommodate the needs of each of the following situations:

- Small shops and laptops—the client, the middleware software, and most of the business services operate on the same machine. We recommend this approach for one-person businesses such as a dentist's office, a home office, and a business traveler who frequently works on a laptop computer.
- Small businesses and corporate departments—a LAN-based single-server application is required. Users of this type of application include small businesses, such as a medical practice with several doctors, a multi-department corporation, or a bank with several branch offices. In this type of application, multiple clients talk to a local server. Administration is simple: security is implemented at the machine level and failures are detected easily.
- Large enterprises—multiple servers that offer diverse functionality are required. Multiple servers can reside on the Internet, intranets, and corporate networks, all of which are highly scalable. Servers can be partitioned by function, resources, or databases, and can be replicated for increased fault tolerance or enhanced performance. This model provides a great amount of power and flexibility. How well you architect your application is critical to this client/server model. You may need to partition work among servers, or design servers to delegate work to other servers.

BACK TO TOP ▲ NEXT TOPIC ▶

Cookie Preferences | Ad Choices

Copyright © 2001 BEA Systems, Inc. All rights reserved.
Required browser: Netscape 4.0 or higher, or Microsoft Internet Explorer 4.0 or higher.

Cookie Preferences | Ad Choices