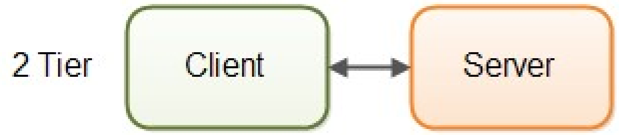
Client-Server Architecture

Client / server architecture is also called 2-tier architecture. A client is talking to a server which performs some services on behalf of the client.

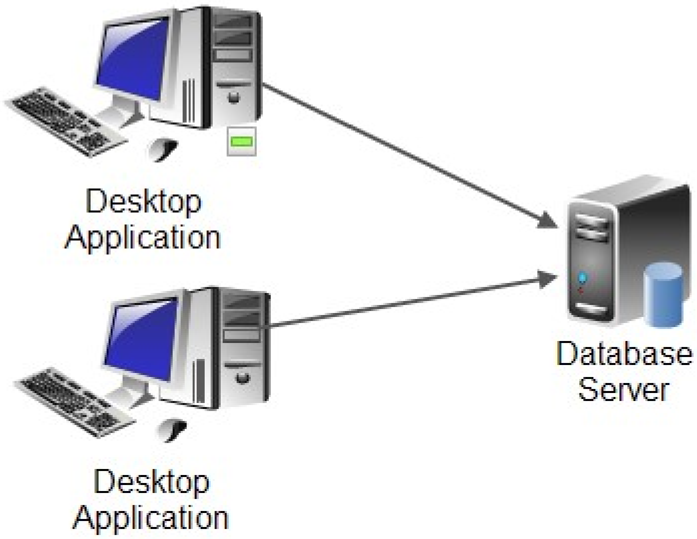


Common examples of client / server communication is:

- Desktop application to database server communication
- Browser to web server communication.
- Mobile to server communication.
- FTP client to FTP server communication.
- Email

2-Tier Architecture Disadvantages

In the early days of client / server applications, desktop application to database server communication was a normal use case. Most of the business logic was embedded inside the desktop application. Therefore, this style of client / server applications was also called "fat client applications". Fat client applications are illustrated simply here:



Embedding all the business logic of a client / server application inside the client applications has some disadvantages. First, it results in potential race conditions (parallelization problem) when two desktop applications attempt to update the database at the same time. If two applications read a record, update it, and save it again at the same time, which version of the updated record will be saved in the database?

Another problem with fat client applications is that the client application has to be installed on each client machine. Back in the day that had to be done manually, but today automatic installation systems exist that can install applications on desktop computers. That way applications (including updates) can be administered centrally.

The disadvantages of 2-tier fat client applications are what made software developers move to 3-tier and n-tier architecture.