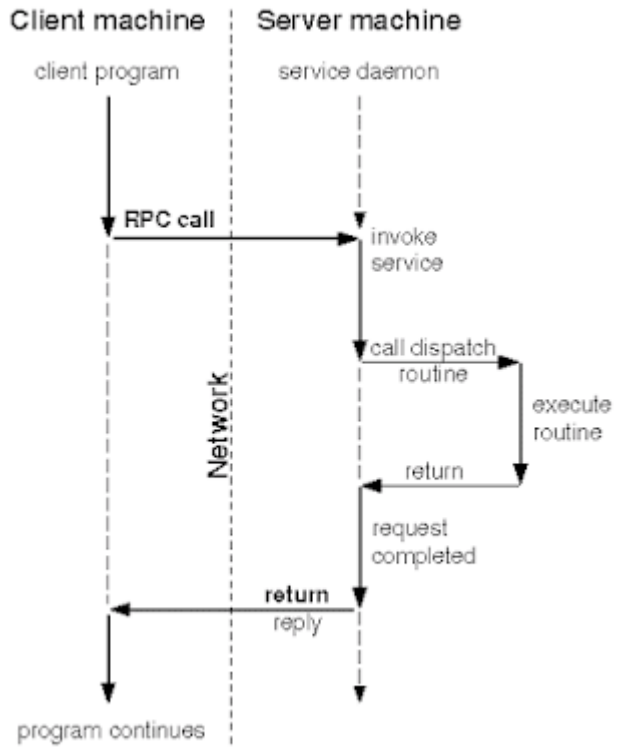
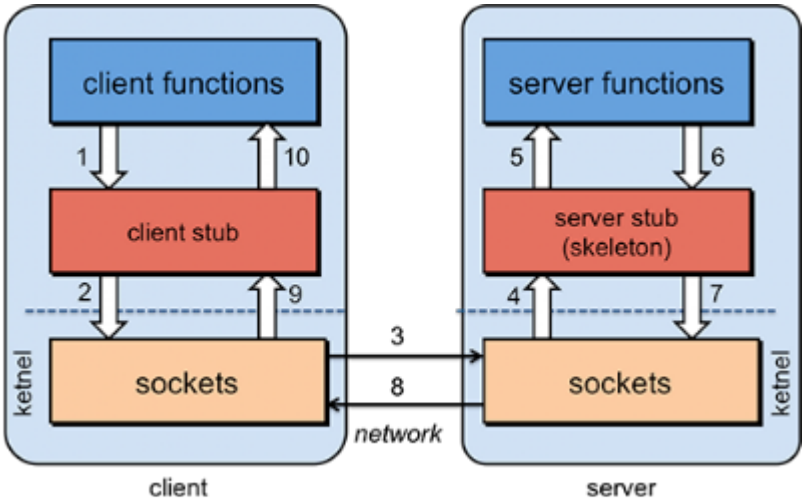
Remote Procedure Call (RPC) Implementation

- An RPC is analogous to a function call. Like a function call, when an RPC is made, the calling arguments are passed to the remote procedure and the caller waits for a response to be returned from the remote procedure.
- Figure 1 shows the flow of activity that takes place during an RPC call between two networked systems.
- The client makes a procedure call that sends a request to the server and waits. The thread is blocked from processing until either a reply is received, or it times out.
- When the request arrives, the server calls a dispatch routine that performs the requested service, and sends the reply to the client.
- After the RPC call is completed, the client program continues. RPC specifically supports network applications.
- The client calls a local procedure, called the client stub. To the client process, this appears to be the actual procedure, because it is a regular local procedure.
- It just does something different since the real procedure is on the server. The client stub packages the parameters to the remote procedure (this may involve converting them to a standard format) and builds one or more network messages.
- The packaging of arguments into a network message is called marshaling and requires serializing all the data elements into a flat array-of-bytes format.
- Network messages are sent by the client stub to the remote system (via a system call to the local kernel using sockets interfaces).
- Network messages are transferred by the kernel to the remote system via some protocol (either connectionless or connection-oriented).
- A server stub, sometimes called the skeleton, receives the messages on the server.
- The server stub calls the server function (which, to the client, is the remote procedure), passing it the arguments that it received from the client.

- When the server function is finished, it returns to the server stub with its return values.
- The server stub converts the return values, if necessary, and marshals them into one or more network messages to send to the client stub.
- Messages get sent back across the network to the client stub.
- The client stub reads the messages from the local kernel.
- The client stub then returns the results to the client function, converting them from the network representation to a local one if necessary.



Remote Procedure Calling Mechanism Implemetation



RPC Procedure

Related posts:

1. RPC messages
2. RPC mechanism
3. Advantages Disadvantages of DS
4. Distributed computing models
5. Goals of DS
6. Hardware software concepts
7. Issues in designing ds
8. Design and Implementation Issues DS
9. Structure of share memory space
10. DSM Architecture & its Types
11. File Application & Fault tolerance
12. File service architecture
13. Desirable features of good distributed file system
14. Distributed shared memory
15. Election algorithm

16. Client server communication
17. Datarepresentation and Marshalling
18. Communication between distributed objects
19. Load distributing algorithm
20. Task migration and its issues
21. Deadlock issues in deadlock detection & resolution
22. Distributed Scheduling-Issues in Load Distributing
23. Characterstics of Multimedia Data
24. Case Study of Distributed System
25. Distributed multimedia system
26. Distributed DBMS
27. Advantages of DDBMS over centralised DBMS