Lecture 14: Client Server Model

- Client: transmit request to server process and block.
- Server: Execute request.
- Server: Return reply to client.
L14: Peer to Peer

- Processes cooperate to solve a problem or share information.
- No designated client or server.
- Members request and provide services to each other.
L14: Group Multicast

• Single sender and multiple receivers.

• Request is sent to a group of servers and the one that can provide the service responds.
  – No specific IP addresses, but a special group address

• Fault tolerant
L14: Messages

• Two components: Data (Payload) and Header.
• Data is transmitted in a fixed size grouping called a datagram or packet to accommodate different sized messages.
• Padding is used to fill out packets that do not have enough data.
L14: Header Info

• Contains information to get the data from one place to another.
• Header info is added at each level of the protocol stack.
• May indicate size of message, sender, receiver...
• Generally includes information about the message type or structure (data/command).
L14: Transfer Schemes

• Connection Oriented establishes connection before exchanging data.

• Connectionless: each packet carries full address information.

• Datagram: message sent, no acknowledge.

• Acknowledged: message sent and ack.

• Request-reply: sender transmits request and server returns answer (client server).
L14: Pipes

• One-way communication.
• When processes are on separate modules, the pipes are identified by the path to their location – named pipes.
• Restricted to one domain and a single file system.
L14: Sockets

• Processes communicate by sending messages between socket pairs.
• Each socket is identified by a port number (local endpoint address) and a global address (host address on networks, e.g. IP address).
• Communication can be bidirectional or unidirectional.
L14: Remote Tasks

• Client and Server assume direct connection with each other.

Problems:

• How processors express data?
• Data formats
• Addressing schemes
• Message may be executed, partially executed or not executed at all.
L14: Remote Tasks

![Diagram of remote tasks with client, server, kernel, and physical network connected through logical communication.]
L14: Remote Procedure Call (RPC)

• Combines the local procedure call model with the client server model.
• Goal is to have tasks interact with local and remote procedures seamlessly.
• Try to make a remote call look local.
• Create stubs that put parameters into messages
  – Stubs act as representatives of the remote procedure code.
• Pointers become meaningless.
• E.g.: Network File System (NFS), Message Passing Interface (MPI)
• Call is intercepted by the client stub.

• Stub builds message containing info - parameters, return values, data - all put in a format compatible with the network and with the remote system.
Assignment

• Finish reading chapter 16