Communication across networks

Matthew Miller
Sockets

• Purpose
  – Sending/receiving data on network

• Overview
  – Interface between applications and network
  – IP Address, Port #, TCP/UDP
  – Send/receive syntax similar to pipes (Linux)
Details

• Similar functionality in Windows and Linux/Unix
• Programming Basics
  – Include library
  – Set connection information
  – Create socket
  – Establish connection
  – Send/receive data
• Key options
  – Binding ports
  – Blocking calls
  – TCP/UDP
Remote Procedure Call (RPC)

• Purpose
  – Most used for distributed systems
  – Making procedure calls on other networked machines

• Easier to implement than socket programming

• Concept:
  
  ![Diagram showing the process of a Remote Procedure Call (RPC).](image-url)

  - **Blocking state**: Shows the state where the client is waiting for a reply.
  - **Executing state**: Shows the state where the server is processing the request and waiting for the next execution.
  - **Call procedure and wait for reply**: The client initiates the call and waits for a reply.
  - **Receive request and start process execution**: The server receives the request and starts the process execution.
  - **Send reply and wait for next execution**: The server sends the reply and waits for the next execution.
  - **Resume execution**: The client resumes execution after receiving the reply.
Details

• Support built into Windows + Unix/Linux (rpc/rpc.h)
• Address space not shared
  – Data structures must be described with interface definition language for transmission
  – Interface compiler – Handles client/server interactions