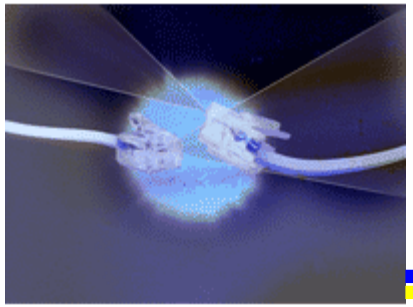
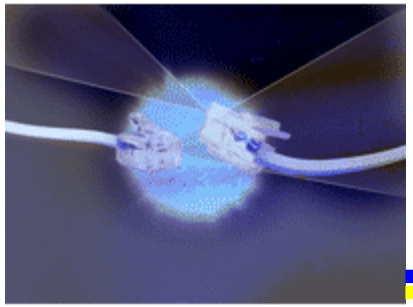


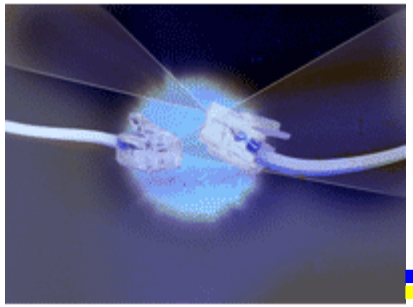
Networking Overview

- Example network application: hitting a URL from a Web browser
  - Up to 6 messages to translate URL to IP address
  - 3 messages to set up TCP/IP connection to Web server
  - 4 messages for HTTP *get* message and response, with positive acknowledgements (ACKs)
  - 4 messages to tear down TCP/IP connection



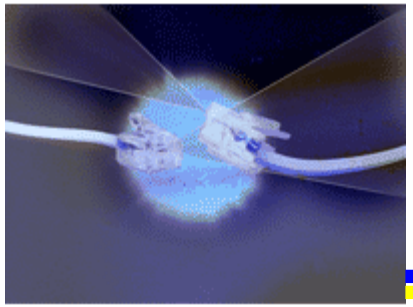
Network Requirements

- Different perspectives
  - Application developer
    - Services needed, performance guarantees
  - Network designer
    - Cost-effective design, efficient utilization, fair allocation
  - Network provider
    - Easy administration and management



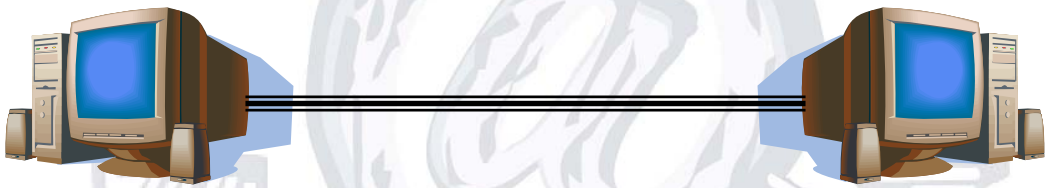
Network Connectivity

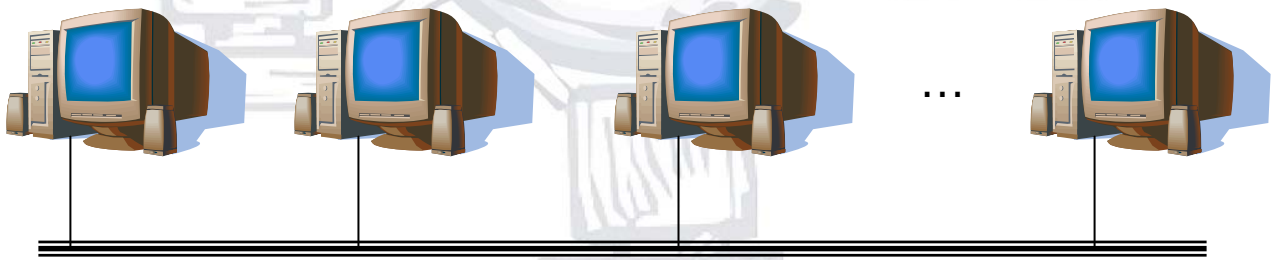
- Principle task of any kind of network is to connect things together. Scope might be limited or world wide, like the Internet
  - Scalability is typically an issue with a network
- Connectivity at different levels
  - *Link* level
  - *Node* level
  - *Network* level

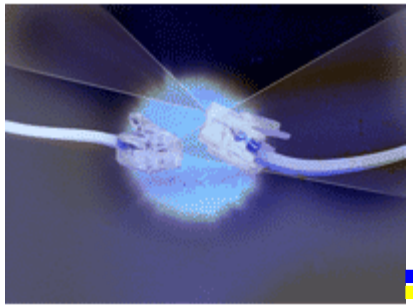


Connectivity (cont.)

- Link – physical connection between two devices
  - Point-to-point
  - Multi-access

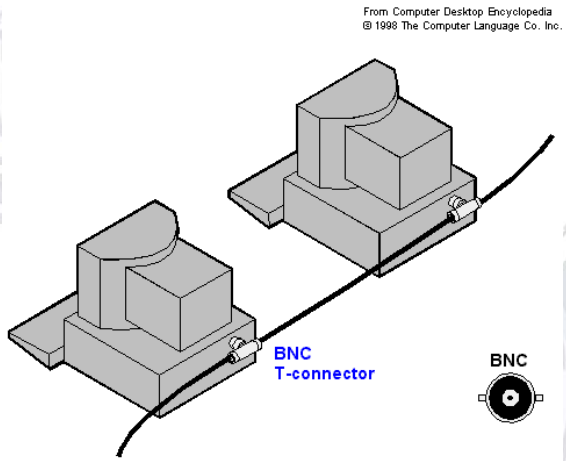






Example Multi-Access Connections

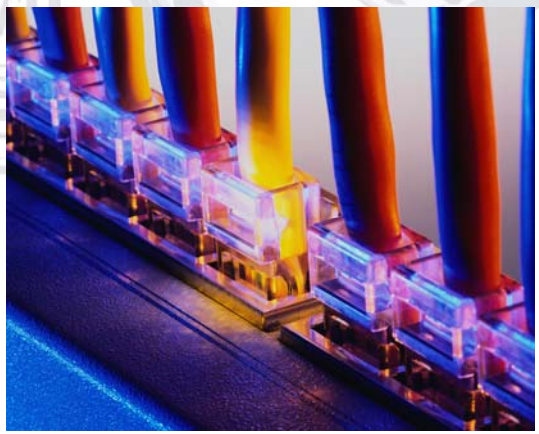
- 10Base-2

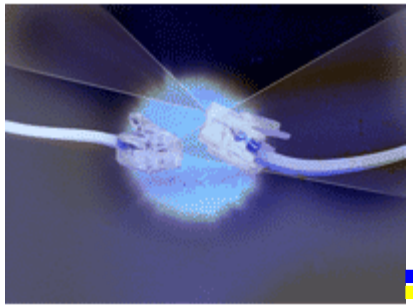


From www.techweb.com

From http://www.windowsnetworking.com/

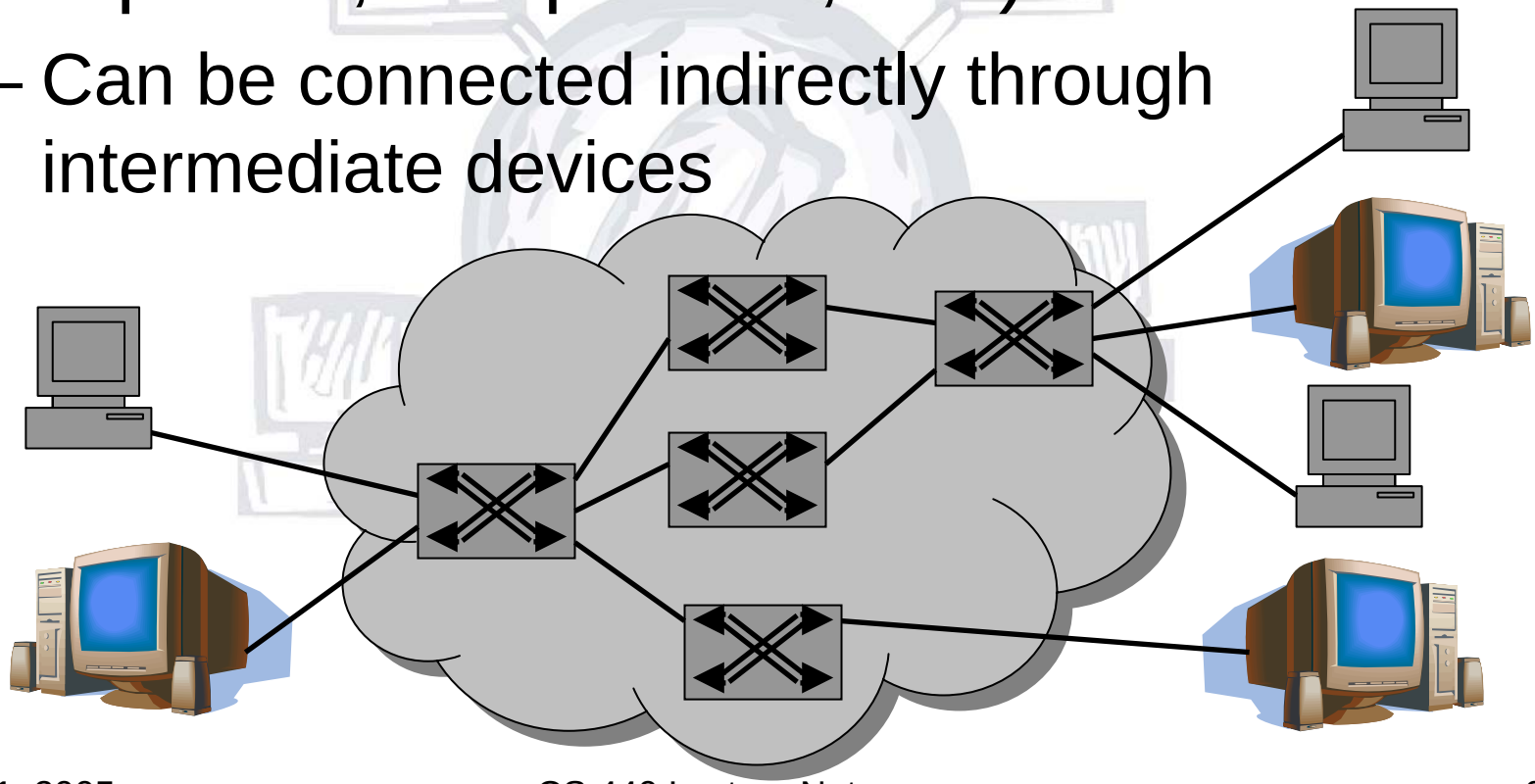
- 10Base-T

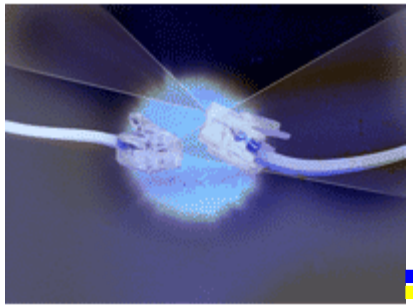




Node Connectivity

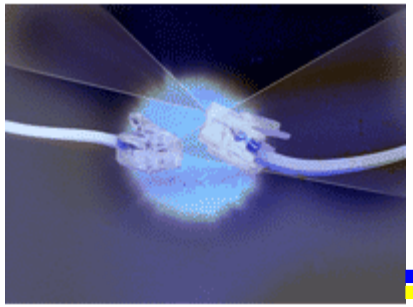
- Nodes – devices that are connected (i.e. computers, cell phones, etc.)
  - Can be connected indirectly through intermediate devices





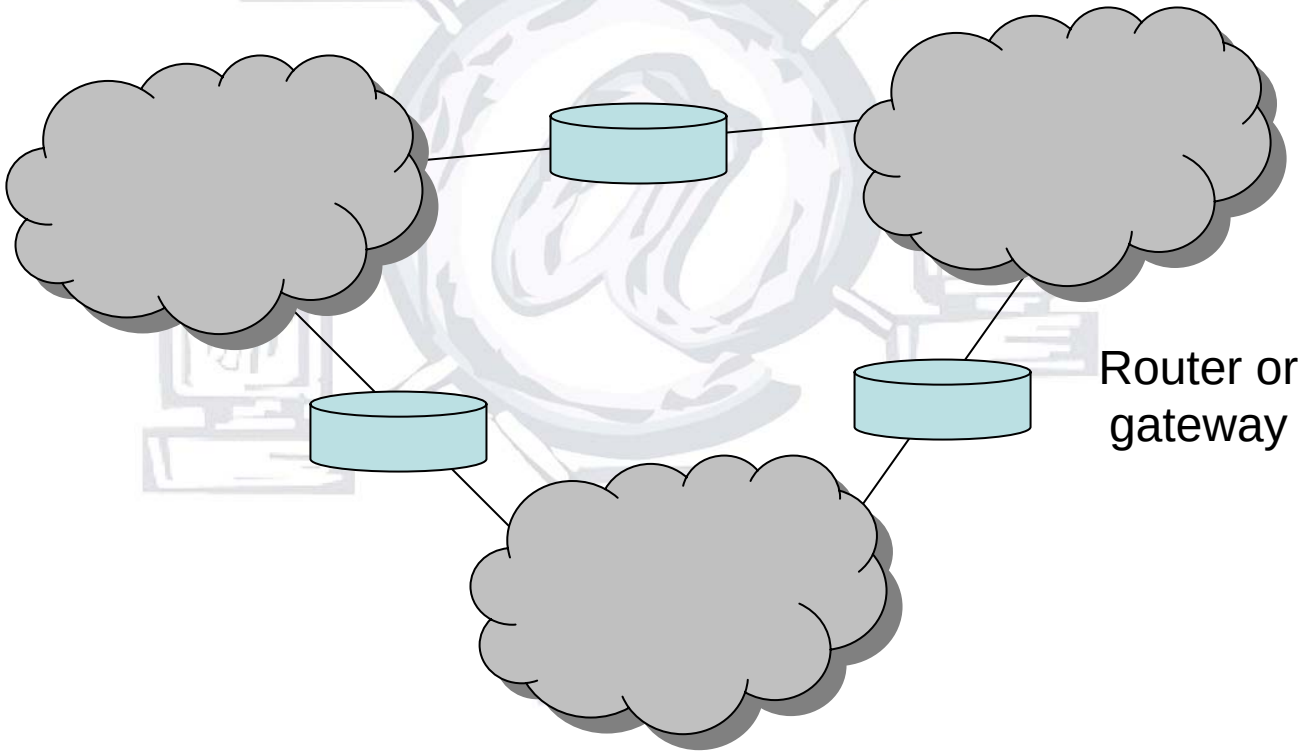
Networks

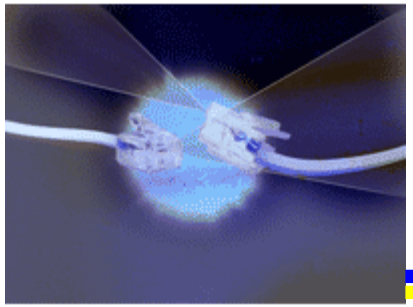
- Connection of point-to-point links – devices with more than one link can forward data between links (i.e. *switches*)
- Different types of switched networks
  - Circuit switched
    - For example, telephone network
  - Packet switched
    - Most computer networks
    - Switches are typically *store-and-forward*



Network Connectivity

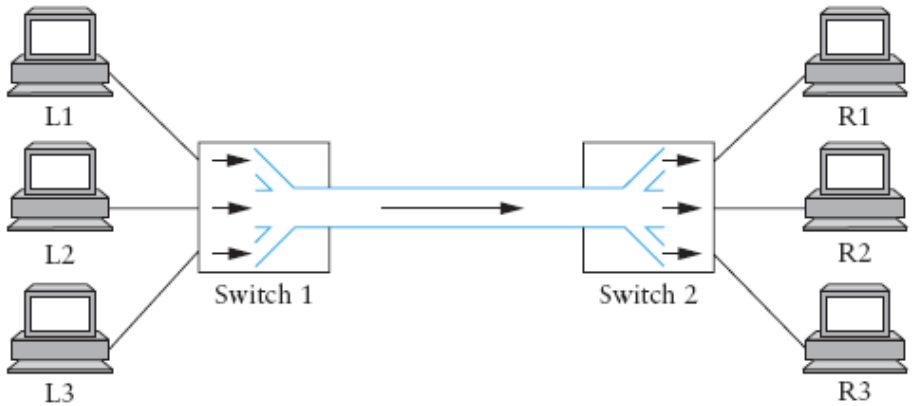
- Connect clouds together into an *interconnected network*, or *internet*

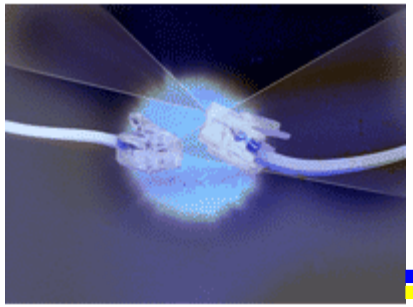




Cost-Effective Resource Sharing

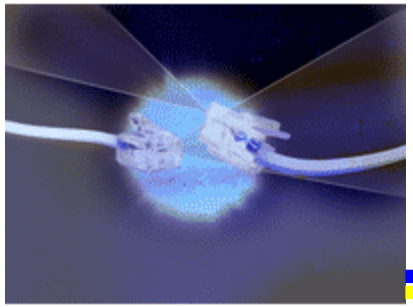
- One goal is to efficiently use connections
- Multiple nodes need to share links
  - Need to use *multiplexing*





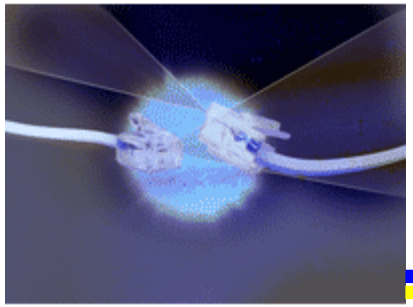
Types of Multiplexing

- Synchronous Time Division Multiplexing (STDM)
  - Break link capacity into equal sized chunks of time and assign a chunk to each host
- Frequency Division Multiplexing (FDM)
  - Same thing, but break capacity into different frequencies
- Statistical Multiplexing
  - Don’t use fixed-sized chunks; allow size to vary and dynamically allow nodes to send chunks.
  - Define max. size per block – each block is a *packet*



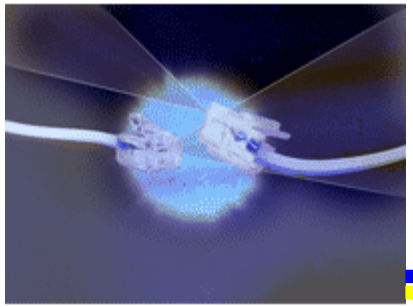
Common Services

- In addition to connectivity, networks must provide a set of services that applications can use
  - Network provides logical *channels* that connect applications on different nodes
  - Different channels might have different service requirements
    - Guaranteed delivery
    - Sequential ordering
    - Privacy



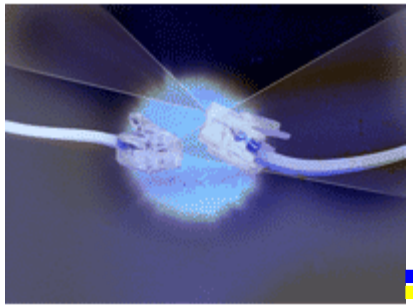
Common Patterns

- Many exchanges are *client-server*
- Different channels depending on semantics of exchange
  - Request/reply
  - Message stream
- Other exchanges and channel types; i.e. *peer-to-peer*



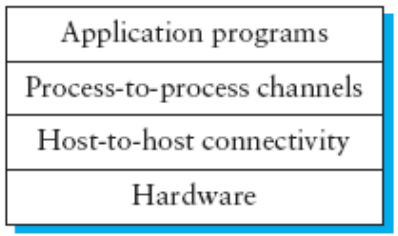
Reliability

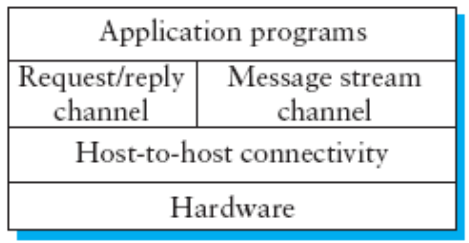
- In the real world, “stuff happens”, even to data on a network
- Different types of errors
  - Bit errors
    - Measured Bit Error Rates (BER) – 10^{-6} to 10^{-7} for copper, 10^{-12} to 10^{-14} for fiber
  - Burst errors
  - Packet errors
  - Node or link errors



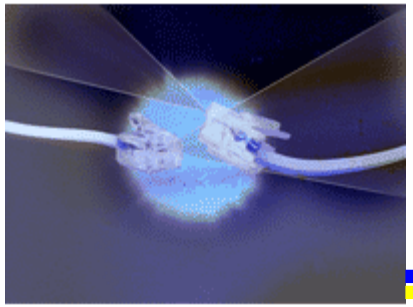
Network Architectures

- General blueprints for design and implementation of networks
- Since most networks are complex, divide into layers of abstraction



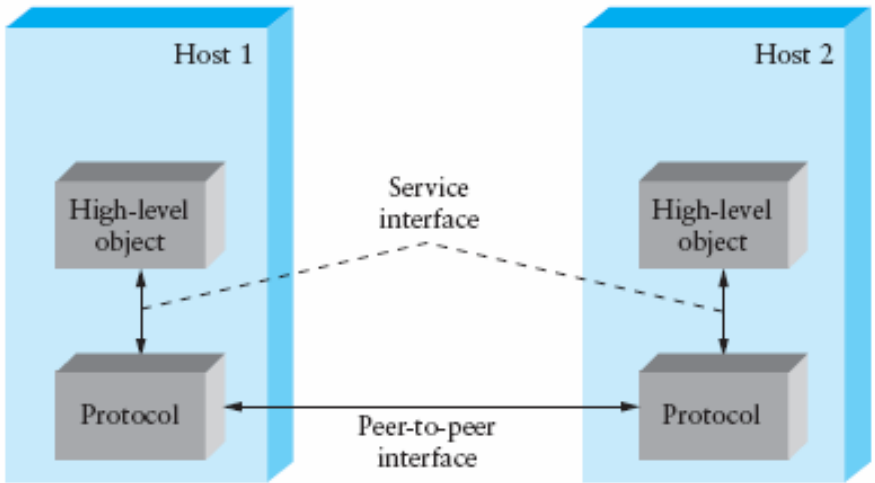


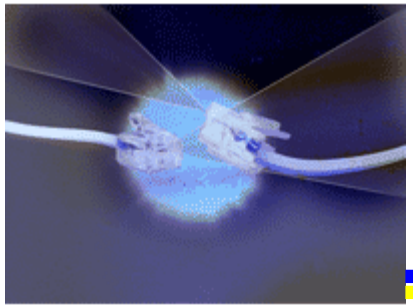
  - Individual layers might have multiple abstractions



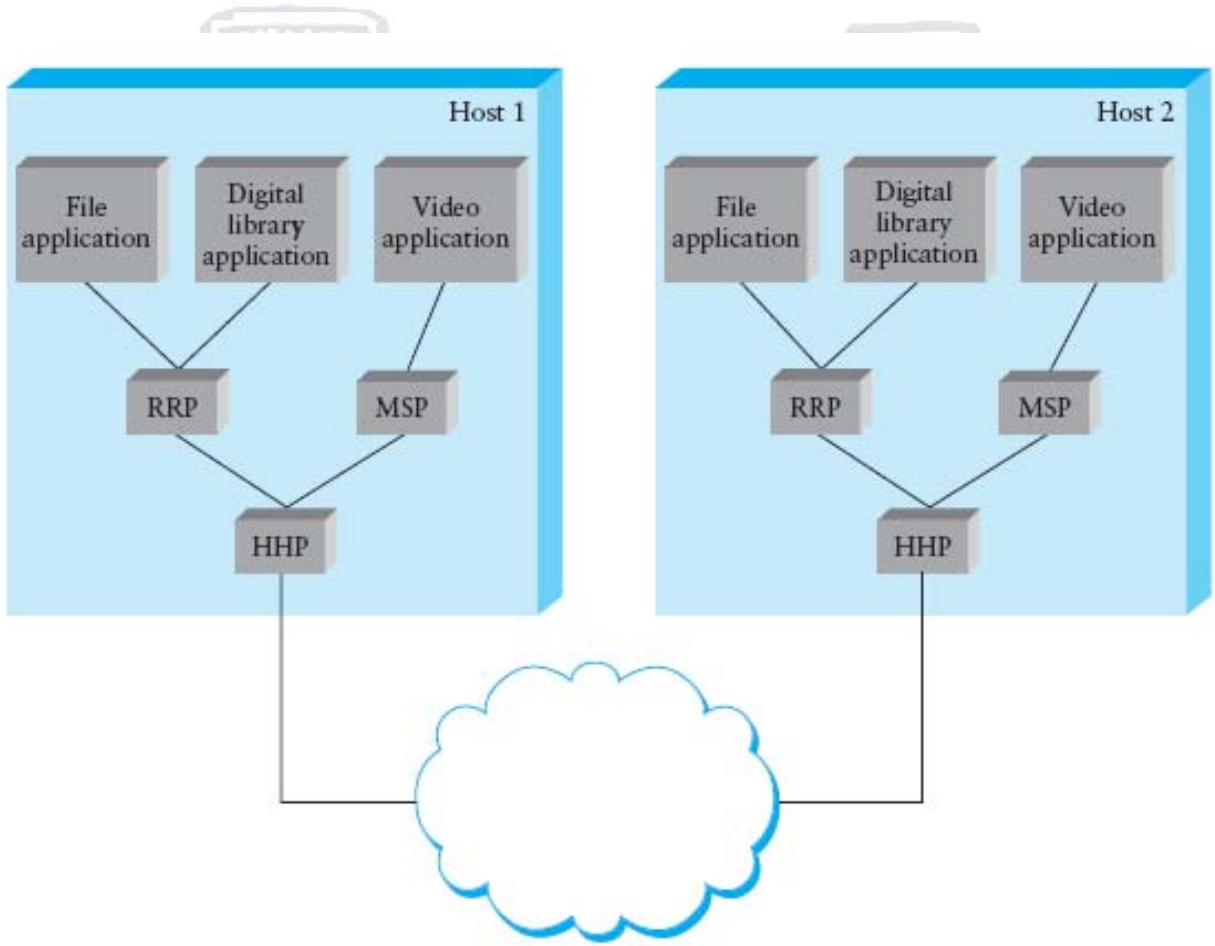
Protocols

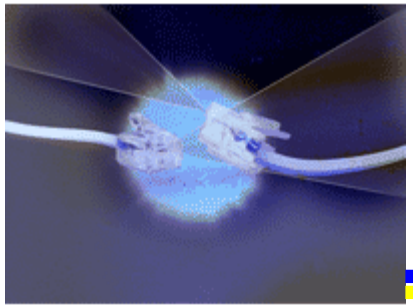
- Protocols are the abstract objects that make up each layer of the architecture
  - Each provides a *service interface* to the next higher layer (i.e. an API), and a *peer interface* to the corresponding protocol on the peer node





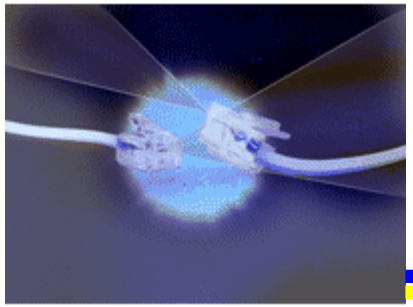
Protocol Graphs and Stacks

- 



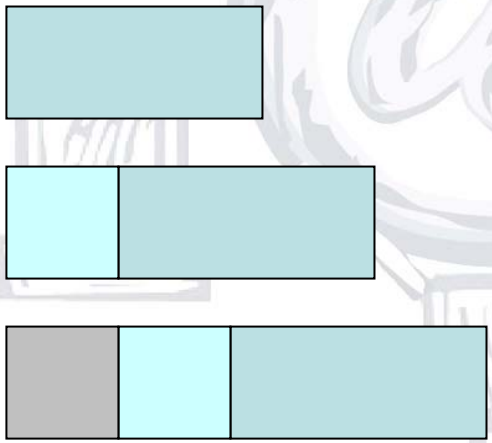
Protocols (cont.)

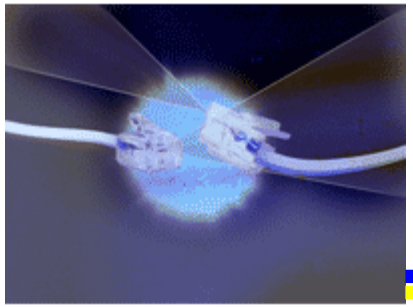
- Protocols have an abstract *specification* and one or more *implementations*
  - Each protocol module must correctly implement the spec in order to interoperate
- Standardization bodies exist to control the proliferation of protocols and to provide mechanisms for introducing, validating, and modifying protocols



Encapsulation

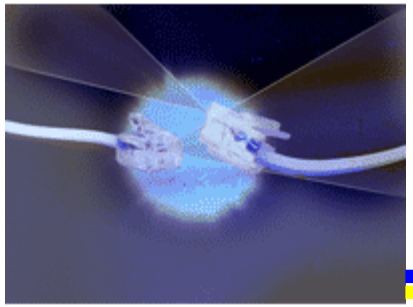
- Each successive protocol in a stack should treat the data handed to it as a “black box” and add on the necessary data to perform its function





ISO Standard Architecture

- International Standards Organization's Open System Interconnection architecture (ISO-OSI)
- Reference model – not widely used in practice
- ISO also publishes the *X.* protocol specifications that fit into this model



OSI Model

-

Application

Presentation

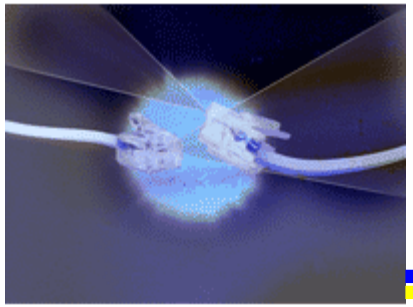
Session

Transport

Network

Data Link

Physical



Internet Architecture

- ARPA model

Application	FTP, HTTP, ...
Transport	TCP / UDP
Network	IP
Physical	Ethernet, X.25, FDDI, …