

1 – Intro to Networks

List of Slides

- 1 Intro to Networks
- 4 Why Distributed Memory?
- 5 Networking Performance
- 7 Protocols and Layers
- 8 Protocols Defined
- 9 A Simple Example
- 11 Significance of Protocol Layers
- 12 Another View of Protocols
- 13 Network Layering Revisited
- 14 A Simple Three Layer Model
- 15 Protocol Layering
- 16 Protocol Layering
- 17 Some Notation

- 18 The Protocol Stack
- 19 Benefits of Layering
- 20 Costs of Layering
- 21 Layering Design Issues
- 22 The ISO OSI model
- 23 Features of OSI/ISO
- 24 OSI Model
- 25 Network Layering
- 26 The Physical Layer
- 27 The Data Link Layer 1 of 2
- 29 The Data Link Layer 2 of 2
- 30 The Network Layer
- 31 Connection vs. Datagram Network Layers
- 32 Network Layer Examples
- 33 Transport Layer
- 35 Transport Layer Examples

- 36 TCP/IP Layering
- 37 Transport Layer Examples
- 38 The Session Layer
- 40 The Presentation Layer
- 41 The Application Layer
- 42 The OSI Layers Revisited
- 43 Summary of Layering
- 44 TCP/IP Link Layer Protocols
- 45 SLIP Deficiencies
- 46 CSLIP and PPP
- 47 PPP Frame
- 48 PPP Layout Details
- 49 PPP Transmission Details
- 50 Useful TCP/IP Diagnostic tools

2 – Why Distributed Memory?

Feature	Shared Memory	Distributed Memory
IPC	Signals, Shared Memory Semaphores, Mutex Locks Pipes, Message Passing	Message Passing
Scalability	Limited (Fan Out/Fan In)	More Scalable
Programming Difficulty	Easier	Harder
Latency	Lower	Higher

3 – Networking Performance

Because machines on a network send data across a wire, we can measure efficiency as:

$$\begin{aligned}\text{Total Latency} &= \text{Sender Overhead} + \text{Transfer Latency} \\ \text{Transfer Latency} &= \text{Time of Flight} + \frac{\text{Message Size}}{\text{Bandwidth}} \\ \text{Time of Flight} &= \frac{\text{Distance}}{\text{Signal Velocity}}\end{aligned}$$

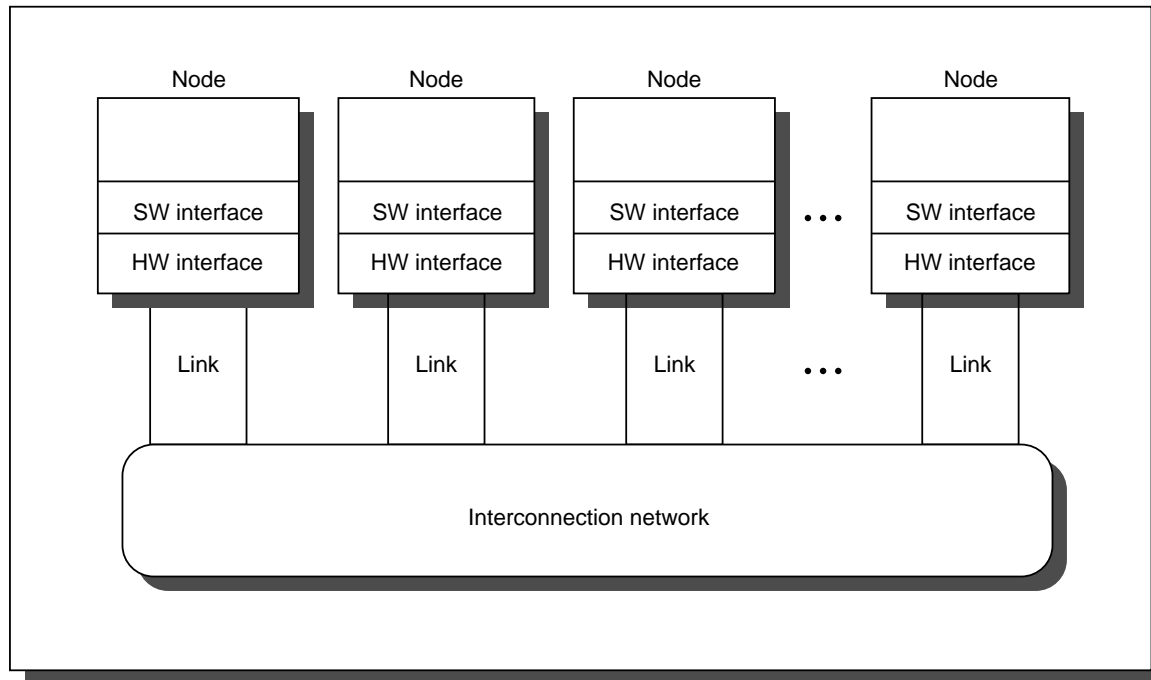


FIGURE 7.1 Drawing of the generic interconnection network.

4 – Protocols and Layers

- *Layers of Abstraction* — Encapsulate complex details in the *lower layers* and interfaces with a simplified view at *higher layers*.
- *Peer Entities* — Interacting parties at the same level of abstraction.

5 – Protocols Defined

- *Protocols* — A set of rules governing communication between communicating peers.
 - Defines format of communication.
 - Specifies the set of valid messages.
 - Defines the meaning of each message (by type not necessarily content).
- A protocol is needed for any interaction between peers.

6 – A Simple Example

- Suppose we want to exchange files over a network that:
 - corrupts packets,
 - does not reorder packets and
 - does not lose packets.
- A simple (but not so great) protocol:
 - Send each file as a series of packets
 - Compute and send a *checksum* (an ECC Code)
 - Receiver sends acknowledgement (ACK) or negative acknowledgement (NACK).
 - Sender waits for ACK or NACK.
 - If the sender times out on waiting for ACK or NACK, it resends the file.
- What could improve this protocol:
 - A single corrupt bit results in entire file retransmission.
 - If the link goes down, then what?

- What if ACK or NACK gets corrupted?

7 – Significance of Protocol Layers

Protocols tell us:

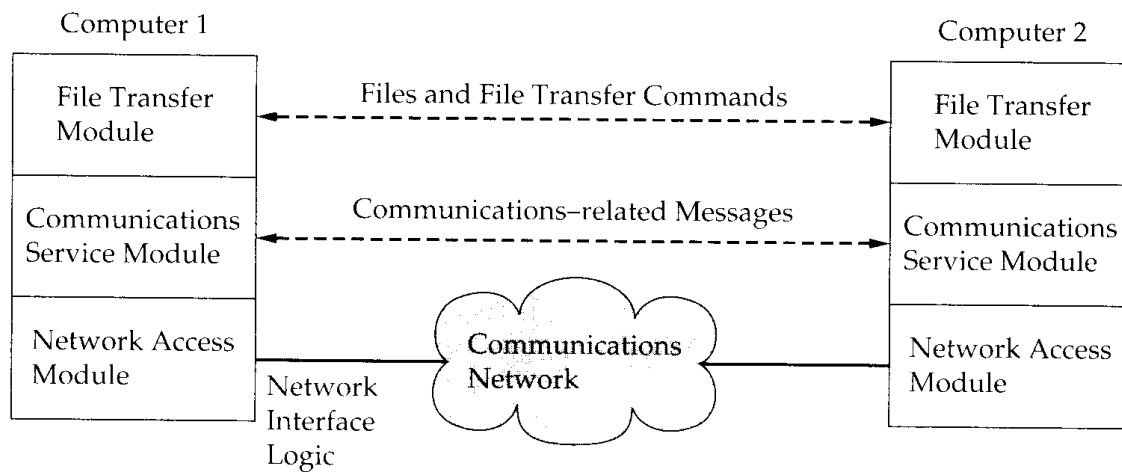
- Syntax of a message:
 - Which fields are in the message?
 - What is the format of the fields?
- Semantics of a message:
 - What does a message mean?
 - e.g. ACK/NACK
- Response to a message:
 - e.g. retransmit of file when a NACK is received

8 – Another View of Protocols

- Protocols provide a *service*
- E.g. our example provided reliable file transfer
- Peer Entities provide communicate and provide an interface to higher level entities.

9 – Network Layering Revisited

Often network is partitioned into layers, a simple example is:



10 – A Simple Three Layer Model

When multiple applications establish multiple connections on a single computer, it is necessary to give the correct data to each application.

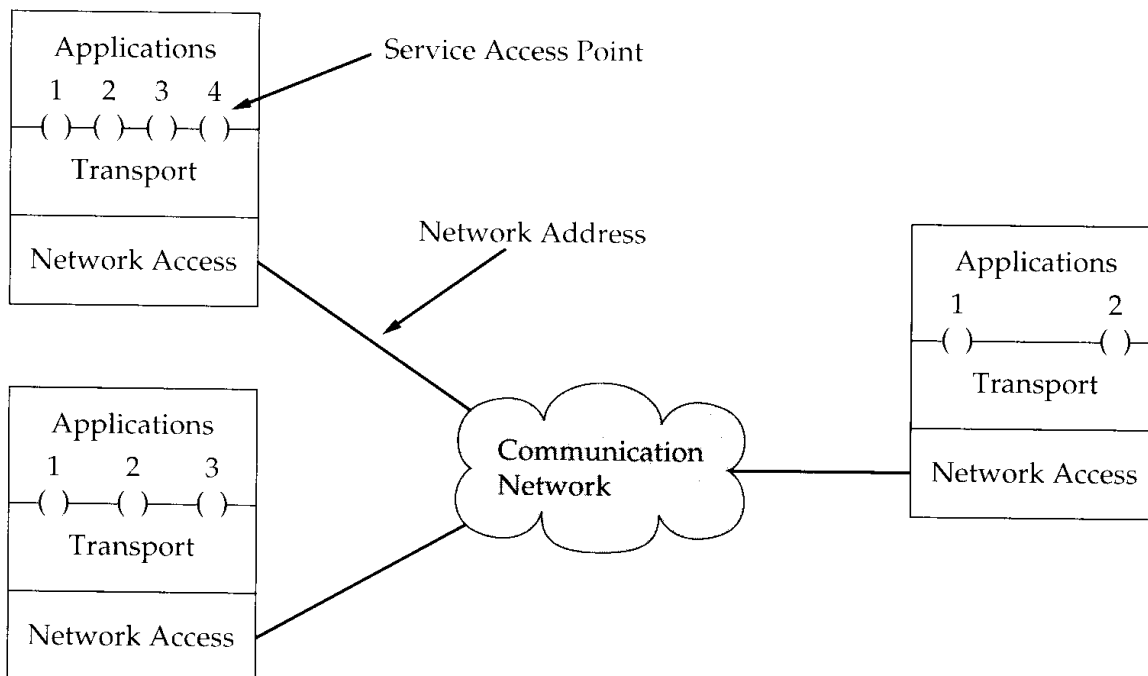
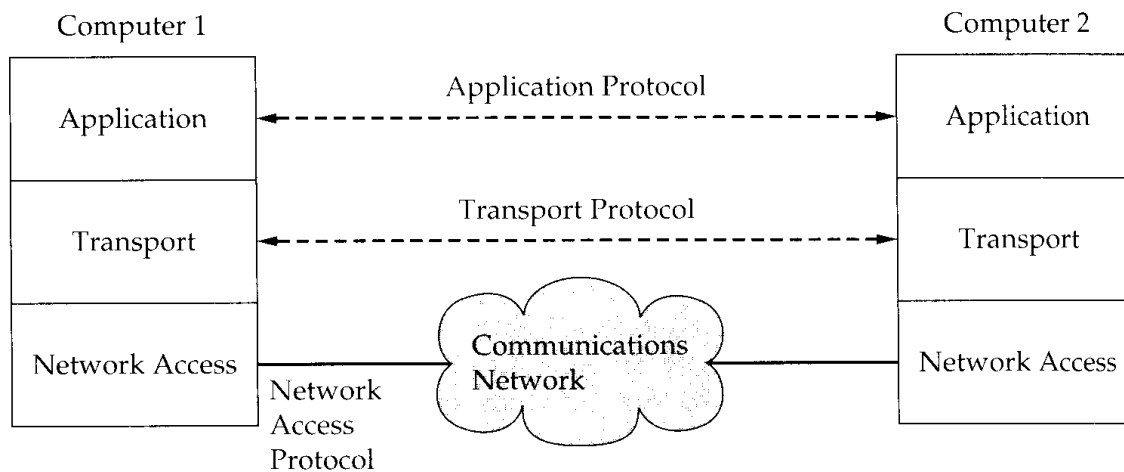


FIGURE 10-10 A Simple Three Layer Model

11 – Protocol Layering

Logical connections become formed between higher peers at level layers.



12 – Protocol Layering

- A network providing many services requires many protocols.
- Some services are independent.
- However, some services depend on each other.
- A protocol *A* might use another protocol *B* as a step in its execution.
 - E.g. Reliable file transfer protocol in our example used a packet transmission protocol as such a step.
- This sort of dependency is called *layering*.
 - Reliable file transfer protocol is layered above packet transfer.
 - Like subroutines (or inheritance in OOP).

13 – Some Notation

- Service Access Point (SAP) — Interface between an upper and lower layer.
- Protocol Data Units (PDU) — Messages (packets) exchanged between peer entities.
- Service Data Units (SDU) — Messages given to a lower layer by a higher layer.
- $PDU = \text{Optional Header/Trailer} + \text{SDU}$.
- E.g. consider mail:
 - A post office is the SAP to a customer.
 - A letter is a PDU.
 - The post office uses mail bags as an SDU to hold letters (PDUs).
 - What would be the SDU header?

14 – The Protocol Stack

- A set of protocol layers.
- Each layer uses the layer below and provides a service to a higher layer.
- Key Concepts (Just like OOP!):
 - Reuse — Many higher layers can delegate details to the same lower layer.
 - Encapsulation — changes to the implementation of one layer need not impact other layers.
 - Information Hiding — details are hidden from upper layers.

15 – Benefits of Layering

- Simplifies implementation by breaking complex problems into manageable pieces:
 - A single service can support many complex services.
 - E.g. Java relies on HTTP which relies on TCP which relies on IP (and DNS, DHCP, RIP, OSPF, BGP, PPP, and ICMP).
- Abstraction of implementation Details:
 - Separates implementation from specification
 - We are free to change implementation as long as service is maintained.
- Reuse of functionality
 - Several upper layers can use common lower layers
 - E.g. TCP on Unix.

16 – Costs of Layering

- Information hiding can be expensive:
 - E.g. Suppose a flow control protocol thinks packet loss is due to short term congestion.
 - But what if it is due to a broken link?
 - The flow control mechanism could fail because it has insufficient information!
- Can we get around it?
 - *Layering Violation* — Implementors can use loopholes (like C++ friends)
 - But then changes in one place may require global changes to maintain correctness.

17 – Layering Design Issues

- There is a trade off between abstraction and getting good performance.
- Good protocol designers:
 - permit sufficient information to be leaked for good performance,
 - but not more, or a local changes will tend to have global impact.

18 – The ISO OSI model

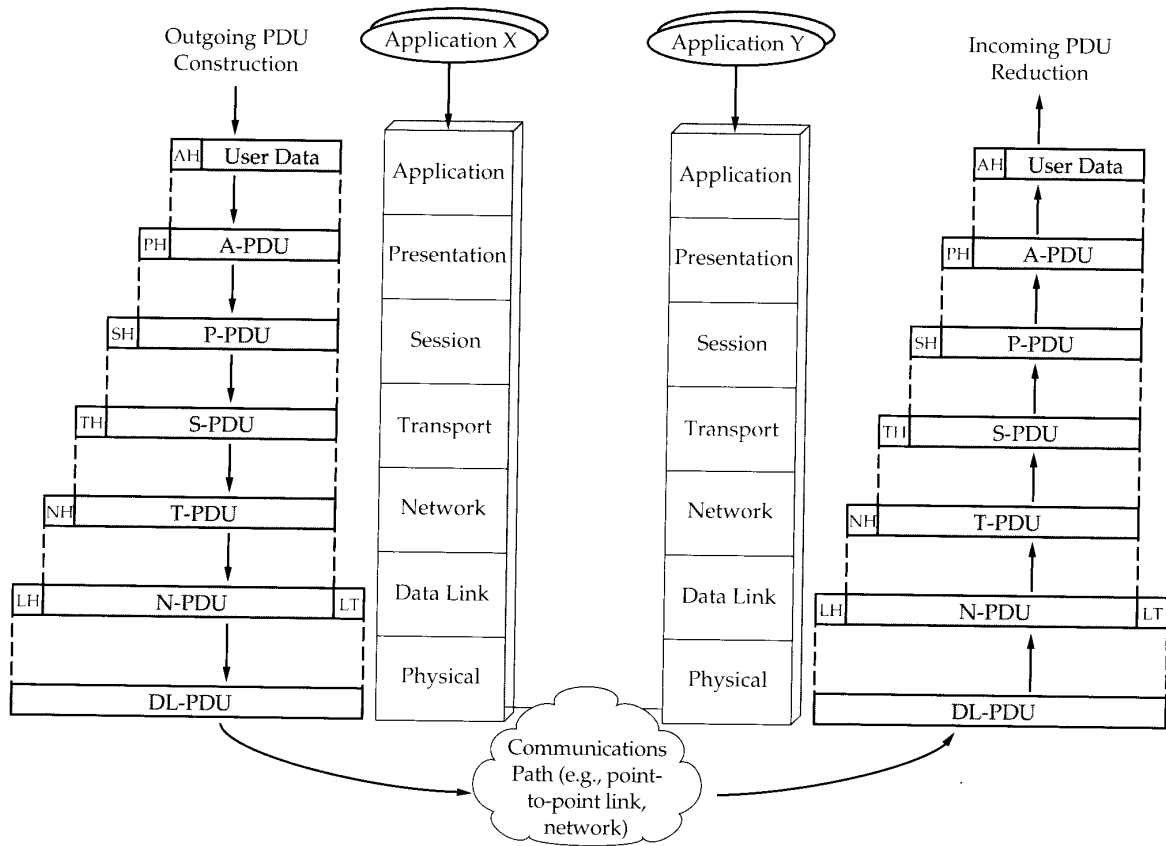
- *Open Protocols:*
 - Have publicly available specifications
 - Are designed by a committee which:
 - * Has membership open to the public
 - * Makes its transactions publicly available
- *Open Systems* implement *open protocols*.
- The International Standards Organization (ISO) developed the Open Systems Interconnect (OSI) protocol.
- As a seminal work, has influenced the entire field.

19 – Features of OSI/ISO

- *Reference Model* — Formally defines what is meant by a layer, service, etc.
- *Service Architecture* — systematically describes:
 - Services (functionality) provided by each layer and service
 - The corresponding server access point (SAP).
- *Protocol Architecture*
 - A set of protocols implement the service architecture.
 - Compliant service architectures can be implemented using non compliant protocol architectures.

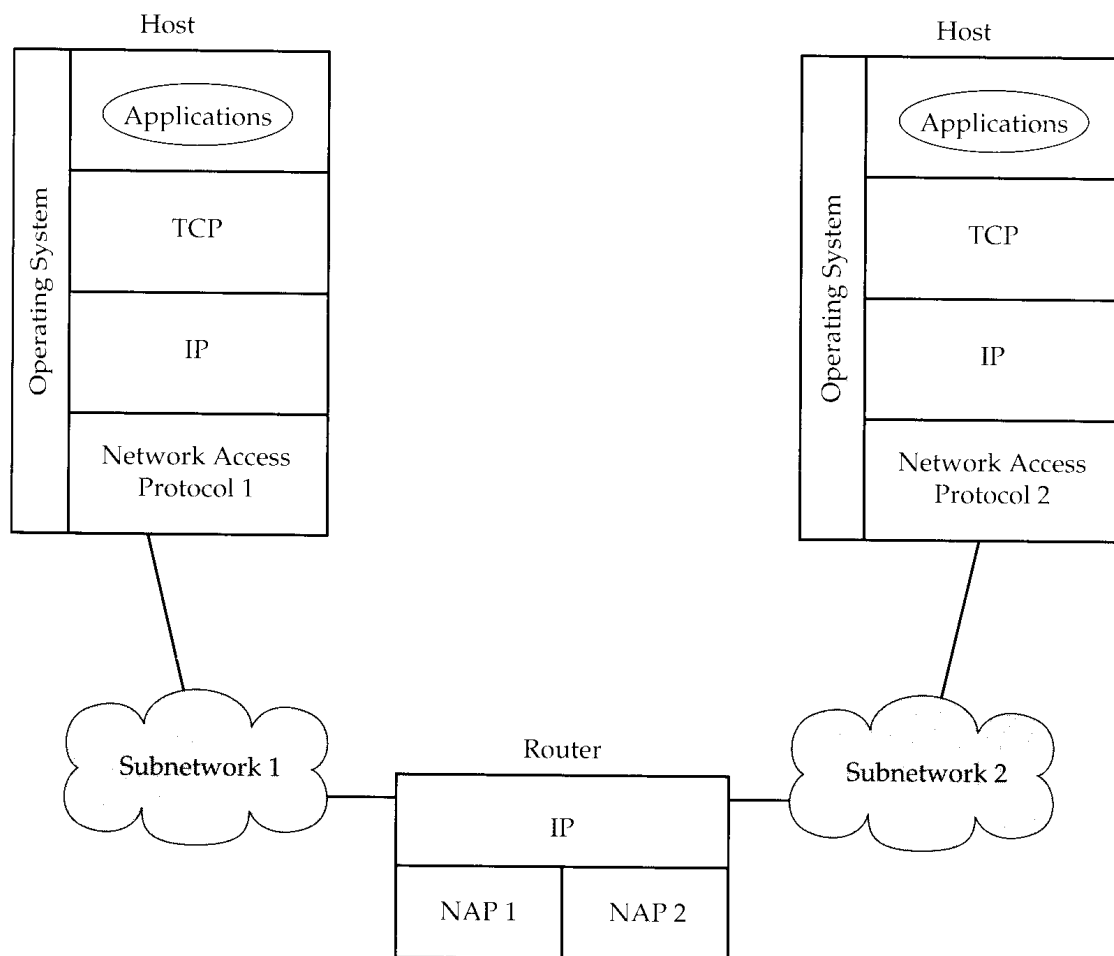
20 – OSI Model

The ISO specifies the OSI 7 layered model (e.g. X.25)



21 – Network Layering

Intermediate routing of data is sometimes used.



22 – The Physical Layer

- Moves bits between connected systems using signals.
- Standard specifies:
 - Coding scheme to represent a bit.
 - Connector shapes and sizes.
 - Bit level synchronization.
- Examples:
 - Postal Network — technology for transporting letters (e.g. trains, planes, ships).
 - Internet — technology to move bits on a wire, wireless link, satellite channel, etc.

23 – The Data Link Layer 1 of 2

- Introduces the *frame*, which is a collection of several bits belonging together.
- *Idle Markers* indicate when a link is not carrying a frame.
- *Begin* and *End* markers delimit start and termination of frame.
- On broadcast links (e.g. ethernet and radio):
 - An end system should only receive bits intended for it.
 - Requires a data link layer address.
 - Negotiates whose turn it is to transmit next.
 - Provides these functions through a *Medium Access Sublayer* (MAC).
- Some data links also retransmit corrupted packets and pace the frame transmission rate onto the link.
 - Part of the *logical link control sublayer*

- Layered over MAC sublayer.

24 – The Data Link Layer 2 of 2

- Data link protocols are the lowest layer implementable in software.
- Is strongly dependent on physical layer properties.
- Are often bundled on a *host adaptor card* (e.g. Ethernet)
- Examples:
 - Postal Network uses bags to frame letters.
 - Internet has a variety of data link layer protocols
 - * Most common is ethernet
 - * Also has FDDI, SONET, HDLC

25 – The Network Layer

- Logically concatenates a series of links together to form an abstract *end-to-end* link.
- Any two end systems can communicate by computing a route between them.
- Provides network wide addressing.
- Hides data link layer details/idiosyncracies.
 - At end systems:
 - * Segments and reassembles messages
 - * Detects errors
 - At intermediate systems:
 - * participates in routing protocols to create routing tables
 - * forwards packets
 - * schedules packet transmission order
 - * (if needed) selects packets to drop

26 – Connection vs. Datagram Network Layers

- For Datagram networks, provides:
 - Routing and
 - Data forwarding
- In connection oriented networks:
 - Establish a distinct data plane and control plane.
 - Data plane responsible for forwarding and scheduling of data.
 - Control plane responsible for routing, call-establishment, and call-teardown.

27 – Network Layer Examples

- Postal Network
 - Sets up static internal routing tables
 - Forwards from source to destination
 - Many qualities of service
- Internet
 - Provided by the *internet protocol* (IP)
 - Found in all end and intermediate systems.
 - Provides abstraction of end-to-end link.
 - Segmentation and reassembly.
 - Packet forwarding, reassembly and scheduling.
 - Unique IP addresses.
 - Can use any data link layer, but provides only best-effort service.

28 – Transport Layer

- Network layer provides raw end-to-end service.
- Transport layer provides additional abstractions for end-to-end links:
 - Error Control — messages delivery recovers from:
 - * Packet Loss — retransmit (Requires a Time Out!)
 - * Corruption — detect, discard and retransmit
 - * Duplication — discard
 - Flow Control — Match transmission rate to sustainable rate on path and at destination.
 - Multiplexing — Adds a port number so multiple applications at the same end-system each get their own data.
- Some transport layers provide fewer services
 - *Lightweight* transport layers may provide

(for example) error detection but not flow control or retransmission

29 – Transport Layer Examples

- Postal Network — has no transport layer (customers do this).
- Internet — Has two popular transport protocols:
 - Transmission Control Protocol (TCP) — provides error control, flow control and multiplexing
 - User Datagram Protocol (UDP) — only provides multiplexing (UDP lacks flow control and error control).

30 – TCP/IP Layering

TCP refers to the *transmission control protocol* at the host to host (transport) layer, and IP refers to *internet protocol*. The internet uses the TCP/IP uses 4 layers:

OSI	TCP/IP Protocol Suite
Application	Process
Presentation	
Session	
Transport	Host-to-Host
Network	Internet
Data Link	Network Access
Physical	

31 – Transport Layer Examples

- Postal Network — has no transport layer (customers do this).
- Internet — Has two popular transport protocols:
 - Transmission Control Protocol (TCP) — provides error control, flow control and multiplexing
 - User Datagram Protocol (UDP) — only provides multiplexing (UDP lacks flow control and error control).

32 – The Session Layer

- Not common
- Provides:
 - Full-duplex service — for simplex networks, concatenates a pair of transport endpoint links.
 - Expedited data delivery — allows some messages to skip ahead of others in end-systems queues using a low delay transport connection.
 - Session Synchronization — lets users place markers in the data stream (for checkpointing) and roll back to a specified previous mark.
- Examples:
 - Postal Network — Implemented by a customers.
 - * Shipping and receiving clerk report to head clerk providing duplex service.
 - * Chief clerk can use courier service or

priority mail (expedited delivery).

- * Chief clerk may choose to resend a sequence of messages or not.
- Internet — Not part of the IP family of protocols.

33 – The Presentation Layer

- Impacts user data, specifying an intermediate representation, e.g.:
 - Endianness
 - Encryption
- Typically ad hoc
- Examples:
 - Postal Network — A customer translates business letters into English, although that may not be their native language.
 - Internet — no standardized presentation layer, only endianness for 2 and 4 byte integers.

34 – The Application Layer

- The set of applications using the network.
- Does not provide any services to other layers.

35 – The OSI Layers Revisited

Recall our Postal Network Example:

1. Physical Layer — The actual letter to be sent.
2. Data link layer — Letters carried by plane, train, ship.
3. Network Layer — Postal system routes letters from source to destination.
4. Transport Layer — Mail clerk resends lost letters.
5. Session Layer — Chief clerk uses priority mail, or has sequences of letters retransmitted.
6. Presentation Layer — Format of the letter sent
7. Application Layer — Person using postal system

36 – Summary of Layering

1. Decomposes complex problems into simple smaller pieces.
2. Provides the application with sophisticated services.
3. Each layer provides a clean abstract interface to the layer above.

37 – TCP/IP Link Layer Protocols

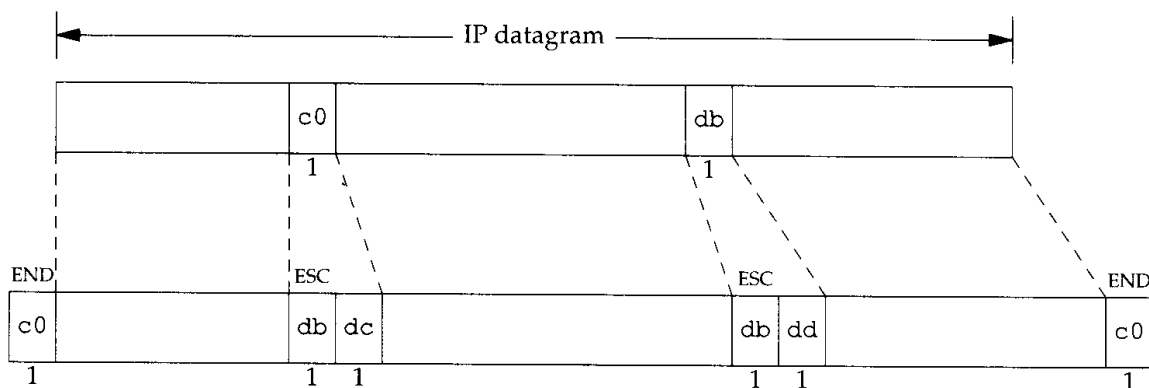
Some common TCP/IP link Layer protocols include:

1. SLIP — *Serial Line IP* — Does the following:
 - (a) The IP datagram is terminated by a special END = 0xc0 character.
 - (b) When an IP contains an END character the string 0xdb, 0xdc is transmitted instead, 0xdb is the SLIP ESC character.
 - (c) When an IP contains an SLIP ESC character the string 0xdb, 0xdd is transmitted instead.

38 – SLIP Deficiencies

Some problems with SLIP include:

1. Each end needs the other's IP address.
2. SLIP lacks a packet type field, so it cannot share the line with other protocols.
3. There is no error checking data added to the packet.



39 – CSLIP and PPP

CSLIP stands for *Compressed SLIP* and replaces a 40 byte slip header with a 3 byte header.

PPP stands for *Point to Point Protocol*, corrects SLIP's deficiencies.

1. Permits 8 bit asynchronous links or bit oriented synchronous links.
2. The *Link Control Protocol* (LCP) establishes, configures and tests data links.
3. *Network Control Protocols* (NCPs) permit different protocols to share a common link.

40 – PPP Frame

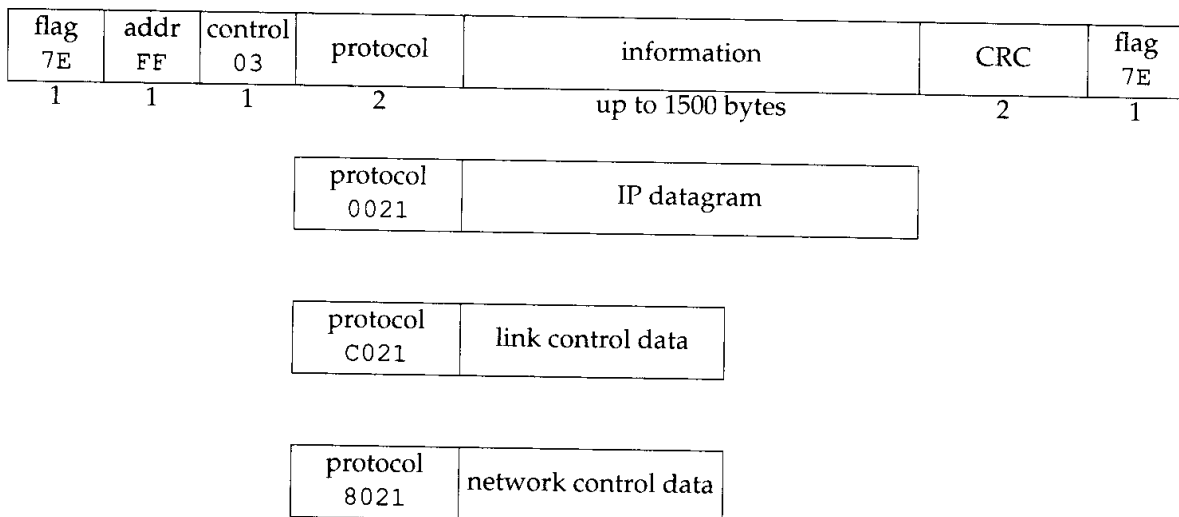


Figure 1: PPP frame layout

41 – PPP Layout Details

The fields in a PPP frame are:

1. flag — The flag is a byte = 0x7E always
2. address — The flag is a byte = 0xFF always
3. control — The control is a byte = 0x3 always
4. protocol — A two byte field indicating what protocol is being used on top of PPP, e.g for IP protocol = 0x0021.
5. information — What ever is in the protocols' frame.
6. CRC — What ever is in the protocols' frame.

Header compression (similar to CSLIP) can be used if endpoints agree to it, to reduce header data.

Many implementations don't transmit the constant address and control bytes, and encode the protocol in one byte. In this case PPP has 3 bytes more overhead than SLIP, 1 byte for the protocol and 2

42 – PPP Transmission Details

Like SLIP control characters are used in PPP:

1. `0x7e` is transmitted as `0x7d, 0x5e` (an escaped flag byte)
2. `0x7d` is transmitted as `0x7d, 0x5d` (an escaped escape byte)
3. Bytes less than `0x20` are also escaped, to avoid ASCII control characters from interfering with the line conditioning.

43 – Useful TCP/IP Diagnostic tools

- netstat — Lists the active interfaces (i.e. internet connections, and Unix domain sockets).
- ping — checks connectivity by sending a special packet requesting reply from the remote machine. (Note, ping can flood a network if used carelessly, be careful with it).
- traceroute — Gives a information about a link by returning information from every hop a special packet type takes.
- whois — Gets the internet user name from the directory. (Useful in complaining about spammers).

References