

Printable View of: Module 1 - Introduction to Data Communications, Application Layer, Physical Layer

[Print](#)

File: Study Guide

Study Guide

Module 1 Study Guide and Deliverables

Readings: Online lecture material and Chapter 1 - Introduction to Networking, Chapter 2 - Application Layer, Chapter 3 - Physical Layer

Chapter 1 discussion ends March 15, 6:00 AM ET

Discussions: Chapter 2 discussion ends March 15, 6:00 AM ET

Chapter 3 discussion ends March 15, 6:00 AM ET

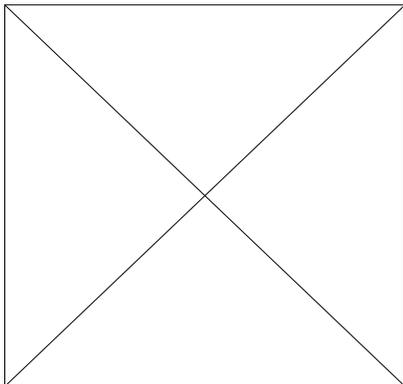
Assignments: Assignment 1 is due March 15, 6:00 AM ET

Assessments: Quiz 1 due March 16, 6:00 AM ET

File: Introduction to Networking

Chapter 1 - Introduction to Networking

Introduction



There is probably no one taking this class who does not have some experience with networking. Certainly we all use, to varying degrees, a network on a daily basis. While the may seem to have sprung on the world all at once, there was a long development both in technology and in business. Both aspects of this development must be understood to fully appreciate the impact of networking on businesses and our lives.

Objectives of this Chapter

- Understand the development of the telephone and computer industries and how this development led to and has affected networking, as well as, what it means for the future.
- Understand the two major models of networking: the beads-on-a-string model and the layered model.
- Understand the workings of the layered model.
- Understand the impact of business on standards and vice versa.

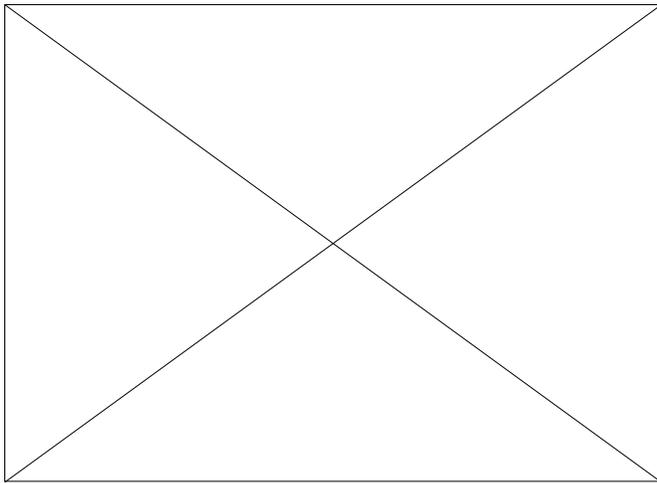
File: Terminology

Terminology

We are still in a transition with respect to terminology. There was a time when "telecommunication" referred to all voice, video, and fax communication handled by phone companies. When digital communication involved only the wires. Voice was considered to be in a physical phenomena, as opposed to an application. "Data communication" was communication between computers and terminals. Generally, data communication was point-to-point or "multi-drop."

Data communication was primarily applied to these simpler networks and their equipment, while "networking" was used to describe packet-switched networking. One could do data communication without relaying, while networking is communication that utilizes relaying. In networks, voice, video, and data are all applications. To the network, bits are bits.

Some may argue that because we want to treat different bits differently, i.e. give them different qualities of service that they do come in colors. Perhaps, but still the "colors" are not the same qualities of service (QoS) in terms of bandwidth, error rate, etc. One can always find applications that have the same QoS parameters as voice, video, or any other application. We are aware of classes of service, but not of specific applications. Trying to support applications in the network is solving the wrong problem. The network must support ranges of applications.



With the spread of the Internet, things have gotten confusing. Networking professionals-those you might meet at an IETF (Internet Engineering Task Force) meeting-believe (of routers and switches using TCP/IP. For them, the Web is just one of the applications that runs on the 'Net. To the business world-especially Wall Street-the Internet is the W equipment or provide the networks are telecom companies.

But fundamentally, from the beginning of networking we have seen the network as related to the operating system:

"Networking is Inter-Process Communication"
- Bob Metcalfe, 1972

...and only Inter-Process Communication.

File: How We Got Here - The First and Second Information Revolutions

How We Got Here

The First and Second Information Revolutions

The first information revolution was really caused by the railroads, believe it or not. From the dawn of time until the coming of the railroads, it had always taken humans the s distant points: the time that a human, horse, camel, or llama took had been the same. It took Julius Caesar and George Washington a week to go 100 miles or to send a messag those same 100 miles in two hours (via the railroad) or send a message in two seconds (via the telegraph, which came a few years later). An incredible change. Lincoln comm

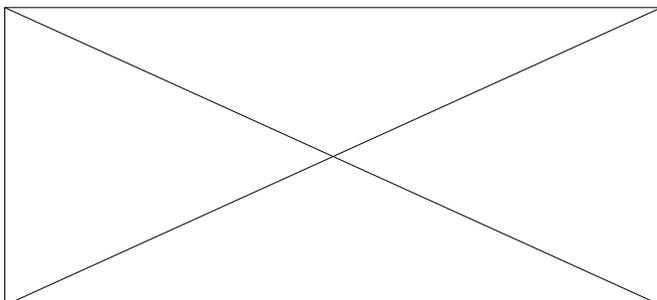
Now we are in what has been called the second information revolution, in which much more information is able to flow much faster than ever before. Still, it is doubtful that tl tremendous impact of that first one. A change of this order of magnitude after millennia is hard to compare to a similar change after only 100 years It seems we have come to c benefits before they happen. Nevertheless, there are enough changes that we didn't expect to keep it interesting.

File: The Development of the Telephony World Until 1970

The Development of the Telephony World Until 1970

After reading the account of the development of the telephone system in the textbook (page 6), we should reflect on its implications for technology. The telephone system beg century. With analog phones, it was necessary to complete an electric circuit all the way from one phone to another, regardless of whether those phones were in the same town was not unlike that from the power plant through the fuse box in your house and your light switch to your lamp.

However, the circuits only needed to be connected when the call was in progress. So a phone call would go through several switching centers where different segments were s these switches were implemented by people plugging in wires. But early in the 20th century, AT&T realized that if they didn't find a solution by mid-century they would have an operator. First, operators were replaced by mechanical switches, then by electronic switches, and then by computer-controlled switches. Toward the end of this period, maj digital, but the lines to the telephones remained analog.



It is easy to see how the architecture adopted by telephone companies for their networks consisted of boxes tied together by wires. We can call this architecture "beads-on-a-st

architecture remained the same. When communication between switches was required to coordinate control, it was done with a separate parallel network, sometimes called "o

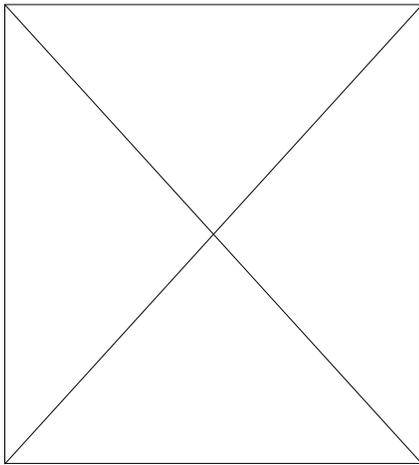
In this architecture, the major concern was with who owned which boxes (customers, phone company, other phone companies, etc.) and therefore, with the interfaces between the architecture tended to be deterministic, hierarchical, and centralized.

As data communication arose after WWII, it was done by sending data over voice wires. As the textbook points out, the corporate and technology environments of the early 20th century were developing as monopolies. The U.S. was unique in that the telephone provider was a private monopoly regulated by a government agency, the FCC. In most of the rest of the world, telephone companies were contained in a single organization in each nation, usually known as the PTT (Post, Telegraph and Telephone). Clearly this gave the PTTs unprecedented power over commercial and government entities in the U.S., AT&T was sometimes viewed with suspicion, as in the 1967 movie [The President's Analyst](#), a cold war spy-vs-spy spoof in which the villain scheming to take over the world is not any of the usual suspects but TPC, The Phone Company.

File: The Development of the Computer Industry Until 1970

The Development of the Computer Industry Until 1970

As the textbook also points out, the computer industry began after WWII (see page 9). By the late '60s, the computer industry had begun connecting terminals to mainframe computer networks, best personified by IBM's SNA (System Network Architecture). These architectures were called master-slave; today we might call them client-server.



The network architectures were asymmetric and deterministic. Given that IBM had 85% of the computer market (like a combination of Intel and Microsoft today), SNA was the dominant architecture. The lines connecting concentrators (which combined traffic to make optimal use of the lines) and terminals to the mainframe installation were purchased from the phone company. Data rates were 9600 bits per second within a building, 300 bits/sec over longer distances. But to a large degree, architectures like SNA were carefully defined to avoid infringing on the domain of telephone companies. These architectures were for the most part proprietary, except at their edges, where customers might (rarely) buy terminals from various terminal suppliers. Hence, these edges were where the most competition was required.

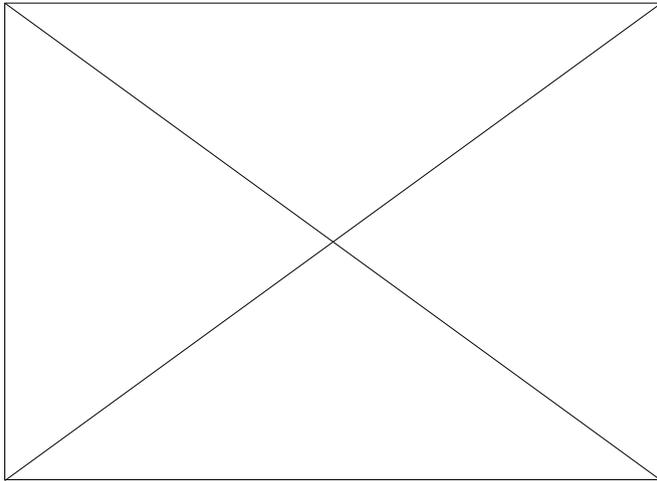
Until the 1970s, when the computer industry began to grow and the impact of the semiconductor revolution became clear, the telephone companies were willing to stay off the computer market. Telephone companies saw computers as a better way to build switches, but it became clear fairly quickly that these two technologies were coming together.

File: The Rise of Networking from the Confluence of Telephony and Computers

The Rise of Networking from the Confluence of Telephony and Computers

The merging of computers and communication was hastened by the ground-breaking work of Paul Baran at the RAND Corporation, a government think tank. Baran noted that voice was continuous and could sustain some loss. Data was bursty and had to be reliable. Also, data connections tended to be quite brief (lasting only seconds, if that) while phone calls usually lasted minutes or longer. The long set-up time of phone calls was a considerable overhead for data. (Phone calls at this time were charged a fixed rate for the first 3 minutes, plus an additional minute. Data connections were generally fractions of minutes.)

The government (and some large corporations) had already built message-switching systems through which text messages of any size could be sent from one computer to another. The government came down and realized that it would be possible to dedicate computers to switching data. Furthermore, the switching could be done much more effectively if the data were sent through the network. Since the packets were being routed over different lines, if a line went down, the packet switch could be rerouted. Baran had made a significant change: he had moved computers from *alongside* the network to switching *in* the network. About a year later, Donald Davies in the UK came up with the same idea from a slightly different perspective.



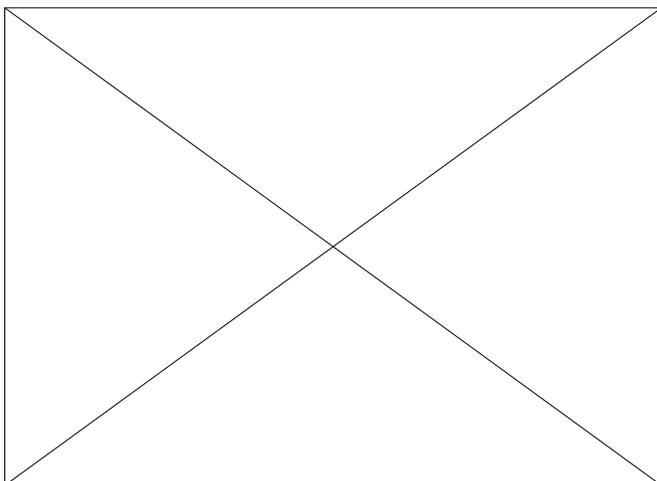
A few years later, Larry Roberts at the Advanced Research Projects Agency (ARPA) in the U.S. Dept. of Defense initiated a project to build a resource-sharing computer network that the cost of research could be reduced if there was a computer network, so that every research institution did not have to duplicate the expensive computing equipment but institutions. Not only did this project show that the ideas worked, but by some accident of history most of those early developers were not traditional telecom or data communicationists. They brought the ideas of software organization from operating systems into networking, i.e., the concept of layers. The ARPANet went live in 1969 with three nodes at UCL. It had expanded to 6, and then to 15, nodes; it has been expanding ever since. The ARPANet architecture differed significantly from the SNA terminal-host orientation. Since it was end-to-end connection-oriented. The ARPANet broke messages into packets to send them along a path.

The ARPANet was far more successful than anyone had reason to believe that it would be. Quickly, it was being put to good use. Particle physicists in Illinois were doing major online searching of news stories and instantaneous notification by email of new events was taking place; a land use management distributed database system was operating in France (too big to be a PC but...) with a plasma panel and touch screen that seldom needed a keyboard; a system similar to, but more sophisticated than, today's web was operating on the Information Center (NIC); and collaborative software development systems were being built—all before 1975, along with experiments with packet voice and wireless networking. There was great excitement about all of the new things that could be done.

But there was one more innovation to come. In the early 1970s, a packet-switching network project was begun in France at the Institute de Recherche en Informatique et Automatique. Pouzin took a somewhat different tack and made the major intellectual breakthrough that created modern networking.

Both the ARPANet and Davies' NPLNet were what we would call today connection-oriented. Packets for an application all followed the same path through the network. If they were blocked, they would recover, find a new path, and continue forwarding data without loss (hopefully). The network tried to provide a very reliable pipe to the applications.

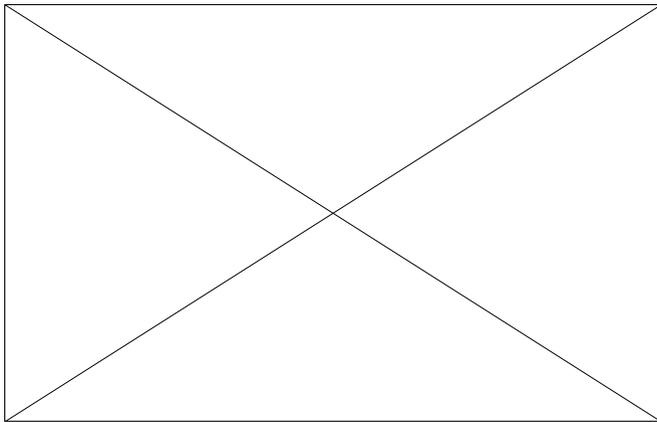
Pouzin's insight was that no matter how good the network was, the applications (or the hosts for them) would always check to ensure that all of the data arrived. He reasoned that the network didn't have to be perfect. It could provide "best-effort" service and the hosts would recover the errors. That also implied that every packet could be routed independently of the others for no reason for them to follow the same paths or for the network to go to all of the trouble to make sure that they did. As long as the "best-effort" was pretty good, the overall result would be better, and be more resilient to failure. The elements of the network were only loosely connected. The architecture was symmetrical, stochastic and connectionless. The project was a decentralized confederation of Greek islands in ancient times.



CYCLADES cemented the architecture that is embodied in the Internet today: a connectionless, best-effort, simple network with end-to-end protocol for reliability. All of the new models of communication based on a symmetrical layered architecture. There was great excitement and everyone was anxious to go out into the world to promulgate these ideas.

The telephone companies had ideas of their own. They didn't like this "best-effort" model of networking. The heart of this idea of networking, which had come out of the interconnection of the Transport Layer, the layer that ensures end-to-end reliability.

The new architecture relegated the phone companies to a commodity business of low margins below the transport layer. The transport layer effectively seals off the highly profitable services from the phone companies. The phone companies could no longer claim that these services were "in the network" and that it was their exclusive purview. They were in the hosts and open to competition.



And that wasn't the half of it. This model was symmetrical. An ironclad rule of design is that a peer architecture can always be subset to be hierarchical, but it is impossible for architecture to be peer. IBM was at a dead end and they knew it. The only strategy for them was to stonewall as long as they could.

Had IBM defined SNA to be a more general peer architecture and then created a hierarchical environment within it for the '70s market, they could easily have expanded into it. It is very likely that the Internet would never have been.

In one stroke, the research community's new big idea had (unintentionally) invalidated the architecture one monopoly (actually dozens of them—that is, all of the PTTs) and one like getting a few dozen 500-pound gorillas all mad at the same time!

The war was on. And it wasn't just a war over architecture. It was a war of connection networks vs. connectionless; the computer industry vs. telephony monopoly; Europe vs. standards committees over the next 15 years.

The research community had begun an effort in 1974 to develop an International Transport Protocol and a Virtual Terminal Protocol under the auspices of the International Federation of Network Operators. They quickly became aware of the ITU (the International Telecommunication Union) working on packet switching recommendations and tried to influence them at the last mir switching recommendations, X.25, for the interface to the network (Remember they were concerned with interfaces between boxes.) and X.28, X.29, and X.3 for remote login connection-oriented network architecture called virtual circuits. The research community had succeeded in influencing this work by introducing a connectionless service, but it

In March 1978, the computer industry through ISO attempted an end run by launching a project to standardize an entire architecture of protocols, called Open Systems Interconnection (OSI) seven-layer model based on a proposal by Charles Bachman, inventor of the entity-relation database model. Soon after, the ITU started a similar effort. Interestingly enough, AT&T and communications were merging and they wanted into the computer business. (Much of the impetus for AT&T agreeing to the break-up of the Bell System in 1984 was so that soon after the break-up, AT&T bought a computer company, NCR.) Many believed that competing efforts between ISO and ITU was a bad idea, so in late 1979 it was agreed and ITU.

Inviting the phone companies into the effort proved to be the downfall of the OSI effort. The computer industry was intent on creating a middle ground that was neither IBM's faction and no consensus on a middle ground. The phone companies did not want a layered model with a transport layer because this relegated them to a commodity business consensus between those who supported the "beads-on-a-string" connection model and those who supported the layered connectionless model led to a constant intense battle. It matches. With so much market share at stake, no one was willing to give an inch—especially since the technology was still being understood. It wasn't clear whether compromise against you. This was not an environment for developing deeper understanding and achieving a synthesis. Consequently, a bunker mentality developed in both camps which they together. Vendors were unwilling to bet on the success of either faction and so committed themselves to supporting both. In the end, the inability to resolve OSI's internal conflict because it was two architectures in one. (An implementation of just the computer-industry vision of the 7 layers, done in the early '90s, took about 70K lines of code, giving credit it won't work.") By the late '80s, there were probably more than a half-dozen organizations, all claiming to be determining the direction of OSI, all differently. OSI was self-defeating that had been there from the outset. The lesson of OSI is that being reasonable and cooperating with the legacy model will always lead to disaster.

The Internet on the other hand completely eschewed the connection approach even while using it through much of the 1980s. At the same time, it tried to distance itself from the phone company camp, an even worse fate. However, the Internet failed to complete the architecture begun in the ARPANet or to resolve its open issues, which have a widely deployed system that is an unfinished demo with 30 years of band-aids and fundamental flaws in scaling, Quality of Service, security, and addressing, etc. There as the NSF (National Science Foundation) and the IETF try to figure out where to go next. Then, in the fall of 2006, someone realized that a problem discovered in 1972 was which there is as yet no clear solution.

The tension between providers and users continues today. Sometimes this tension plays out in the technology like ATM (Asynchronous Transfer Mode) or MPLS (Multiprotocol Label Switching) out in the politics—as with the controversy over "net neutrality." Hopefully, this course will help you read the signs of these conflicting vested interests and at least approach the

File: The Types of Networks

The Types of Networks

The textbook's description of the types of networks is quite good. However, what the student needs to take away from this is that, with one exception, these are for the most part specific technology. The one exception is the designation of Local Area Network (LAN). LAN technologies often have distance limitations built into them. We will see how later chapters. In general, most technologies (with perhaps differing configurations) can be used in backbones, metropolitan networks, extranets, etc.

However, one does find the concepts of backbone, regional, metropolitan, and campus, etc., as well as intranet and extranet are useful tags for characterizing the size, relative

File: Layered Models

Layered Models

Networking is not a trivial task, and usually requires a number of very complex procedures. First, the sender identifies the data path and the receiver. After that, the systems involved are followed by the applications negotiating preparedness. In addition to this, a possible translation of file formats may be necessary. For all these tasks to occur, a high level of coordination

A modular approach to the design of such complex software is always beneficial. This approach breaks complex tasks into subtasks, so that each module handles a specific subtask. This approach occurs between different modules on the same system as well as between similar modules on different systems.

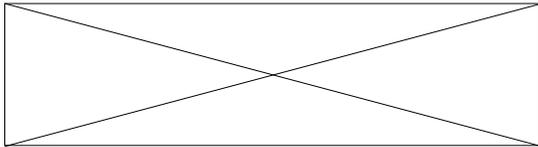
The biggest advantage of modularity is that it offers easier application development. Modules can be developed independently of other modules, and at the same time the network can be modified.

Before the advent of the ARPANet and related networks, data communication had generally consisted of point-to-point links. Most of the issues had to do with the physical medium (when data communication and networking textbooks consisted of 300 pages on the physical media and less than 50 on everything above that.) But with networks there was more than the error control on the links between hosts and switches, and between switches; about routing; and about the overall error and flow control between the hosts, not to mention the

The experience of those early developers with operating systems and software brought new concepts to the world of communications—decomposition of a problem into modular abstractions. To organize all of this, the developers utilized the concept of layers to organize networks.

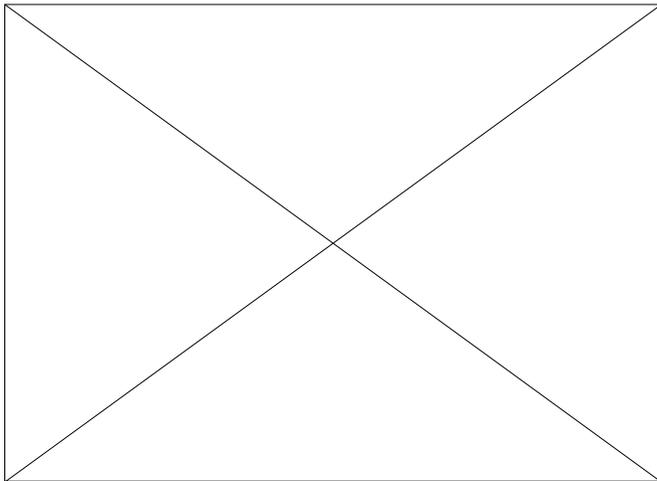
When communication takes place within one system, the mechanisms that are used can be relatively simple. But once it is necessary to go outside the system, there are many things to be done: messages can be lost; or, in some cases, messages can be duplicated, etc. Hence, it is necessary to define a protocol to coordinate the communication.

A protocol is a convention or agreement for procedures to coordinate communication between multiple parties. This shared convention for communicating information includes what is described using a finite state machine. A protocol is defined by the format of its messages; the semantics of those messages (i.e., what to do with them when they are received) (The latter is actually part of the semantics.) The primary purpose of network protocols is to move data. The messages are generally formatted with a header, which consists of information that the receiver understands, and the data, which the protocol implementation doesn't understand. A wealth of terms has evolved for these messages, depending on the layer or the protocol. They are all the same thing.



Thus, the general term for these messages is a Protocol Data Unit (PDU). In some protocols, there are PDUs that carry control information, but no data. The *header* is sometimes used to describe the control information, and the *payload* is the data. The distinction between Information and Data is intended to call attention to what the protocol understands (information) and what it doesn't (data). The

In networking, layers are not just a means of creating modularity. They have a much more important role. They also represent a locus of shared state. When protocols exchange information, they share state between the correspondents. Different groups of layers have different scope. The protocols maintain state information between different places in the network. In some cases, the lower layer protocols share state only with their neighbors. The protocol behavior on the left of the figure is completely independent of that on the right or in the middle. This, more than simple modularity, is the reason for layers in networks. As you can see, this is a major break with the beads-on-a-string model, where everything is ho



The primary rationale for different layers is the different scope of their shared state. A common misconception about layers is that they should be implemented by actually passing data through a series of systems.

File: PDUs

Tracing a PDU

Let us trace a PDU going through the network. The user's application invokes an interface primitive that passes a PDU to the Transport Layer. There a Transport header is added to the PDU, and the Transport protocol state machine is updated. The transport header identifies what stream the PDU belongs to and where it belongs in that stream; it also keeps the sender from sending to

Going down, we encounter the Network Layer protocol, which adds information about where the PDU is to be delivered, then the Data Link Layer, which is responsible for controlling access to the physical medium. Then we encounter the Physical Layer, which contains the functions necessary to interface to the electrical properties of the physical media over which we communicate with a modem, an Ethernet, WiFi, or fiber, etc. This includes the means for imposing bits on the medium, i.e., the modulation.

The Data Link Protocol is tailored to the nature of the physical medium. It will include mechanisms for detecting whether bits were corrupted in transmission and sequence numbers to ensure that bits have not been lost or have arrived out of order; it includes as well mechanisms for controlling the rate of transmission—that is, mechanisms that keep the sender from sending faster than the receiver can handle. Protocols discard messages if they arrive in error but, by one means or another, the message is re-transmitted. Media with high error rates or long delays, or those with both Error Correction, which allows a corrupted message to be “repaired” without retransmission.

The purpose of this layer is to get data to the next switch in the network with a low probability of error. As we described above, since we know the hosts at the ends will check

good enough that the host error checking is cost-effective. This is usually referred to as end-to-end error control. (For many years, the phone companies argued that hop-by-hop and that transport protocols were unnecessary. However, they were never able to build a network that substantiated that claim.)

At a router, when a data-link PDU is delivered and found to be correct, its PCI is stripped off and passed to the network layer. The primary purpose of the network layer is relay to other hosts or routers. Using the address in the network PDU header (which was data to the data link protocol), the network protocol determines which interface to send this

To do this, the PDU must be passed down to the Data Link Layer for the outgoing interface. This may use a different physical media and hence may have a different data-link come in on Ethernet and may go out on WiFi. So a different data-link PCI may be pre-pended to the PDU. A PDU could change media every time it is relayed by a router.

(An aside: Because a router has many interfaces and will be receiving data on all of them, there may be times when there are more PDUs coming in than it can handle. When it can't, it drops them and let the Transport Layer retransmit them later.)

When the PDU reaches its destination host, it will be delivered over the data-link layer attaching the host to the network router. If it is received correctly, the data link header is stripped off. The network protocol looks at the address in the PCI and notices that the message is addressed to it. So it strips off the network protocol PCI and passes it to the transport layer. The transport protocol will have also included a checksum to make sure all the bits are correct, determine the order of the PDU, etc. If all is well, transport strips its application.

Now one may wonder why error codes are needed in both the data link and transport layers. If the bits are correct at each hop along the way, won't they be okay at the end? Memory errors may occur while the PDUs are stored in the memory of the routers, and 2) the memory error characteristics are often different than the characteristics of transmission media. Errors are susceptible to bursts of errors, so one kind of error detection code is used; on the other hand memory errors tend to be single-bit errors, and they require a different kind of code. Errors from different sources: data link is dealing with those on the line, and transport is dealing with those in the routers and switches.

File: The OSI Model

The OSI Model

(N.B.: Before starting this next section, it should be noted that this discussion differs from the definitions in this textbook and several others. One of the authors of this course was a party to most of the events in networking since 1970. The material here is therefore far more accurate than that in the textbook.)

By the time OSI began in 1978, there was a general consensus around the definition of the first 4 layers, which OSI and everyone else used:

- **OSI Physical Layer** is responsible for transmission of bits and is implemented through hardware. This layer encompasses mechanical, electrical, and functional interfaces.
- **OSI Data Link Layer** is responsible for error-free, reliable transmission of data flow control and error correction. The purpose of the Data Link Layer is to provide a reliable link between adjacent nodes.
- **OSI Network Layer** is responsible for routing messages through networks, and is concerned with the type of switching used, such as circuit or packet switching. This layer is well as through packet-switching networks.
- **OSI Transport Layer provides the end-to-end reliability requested by the application. It performs flow control and detection of lost and duplicate messages.**

The structure of the upper layers had up to this point been unclear. There was a consensus that there were probably more layers, but it was unclear what they were. The proposed layers; after some investigation and some political shenanigans by the PTTs arrived at the following definitions:

- **OSI Session Layer** provides functions for managing the dialog between applications, such as major or minor synchronization, resynchronization primitives, etc.
- **OSI Presentation Layer** provides the ability to negotiate the transfer syntax between applications. This may include format and code conversion services, such as file conversion, invocation of character sequences to generate bold or italics, etc., on a printer.
- **OSI Application Layer** provides access to networks for end-users; its capabilities are determined by what items are available on this layer.

A few years into the work (by 1983), OSI had determined that, since there is no multiplexing and the scope of the shared state is the same for all three upper layers, these layers, in other words, the Session and Presentation Layers don't really exist and should be considered functions of the application protocol.

File: The Importance of Standards

The Importance of Standards

Standards are probably more important in networking than in any other part of the computing industry. Basically, without standards, it would be impossible to build equipment. This, vendors would prefer to do without standards. Without standards, customers would become locked into a single vendor and would be limited in whom they could communicate with. This, vendors opt for the next best thing: complexity. As long as standards are complex, and have lots of options or many additions, they become barriers to entering into competition by buying from a variety of vendors.

File: Types of Standards

Types of Standards

Years ago, standards were divided into two types: *Formal* standards produced by industry or government standards committees, and *de facto* standards, which emerged as don't have formal organizations include entities such as the ITU, ISO, IETF, or the WAP Forum. De facto standards generally arise from the market position of a single vendor, such as IBM. De facto standards often start as proprietary specifications.

Over the years de facto standards have gotten a bad name for being closed; this resulted in vendors finding a convenient way to circumvent the complaint. Once it was realized that ANSI or another national or international body, and that standards organizations usually feature a bottom-up structure, not a top-down one, vendors found that the best way to get their products up as formal standards by forming a consortium, forum, or task force, giving it a set of rules, and generating their own standards. There has been a proliferation of these standards since the 1980s.

The Process, or Electro-Political Engineering

Standards and standardization are of special importance in the telecom industry. The reason for this is the paramount necessity for interoperability of hardware and software at

Standardization processes can be done through national, international or professional organizations, such as ITU or IEEE.

Standards are never pure engineering. Although it may seem to be to the domain of engineers who want to be lawyers, it isn't. Standards are never neutral. Most of the arguments are technical issues, but over how the standard favors a specific vendor, and over where and how it draws distinctions or boundaries. This does not mean the arguments are not inescapable in networking; there must be agreement on how the parties communicate.

As ugly as the contentions over standards may be, the standards themselves are important to everyone. Manufacturers must understand how a standard affects their markets. As to one's market. Vendors always contend that they build what their customers ask for, while customers say they can only buy what vendors offer. Vendors will build equipment to the degree possible, they will create barriers that make it difficult for other vendors to enter that market while locking customers in.

A standard is never an implementation design or specification. It is a definition of constraints and external behavior. How that external behavior is achieved is completely up to a standard does so at their peril. Network providers must understand how standards affect them, both with respect to vendors and to providing services to their users cost-effectively.

Since standards are defining the shape of the market, they are important to users. Users have a vested interest in what standards are developed. Providers and manufacturers are not necessarily their users. Years ago the phone companies tried to argue that voice mail was the domain of the network, not the user—that is, that users should not be all machines. The current net neutrality debate is one where the sides are less clear. The current complexity of IP networks favors the vendors, who would like users to believe that

File: The Major Standards Organizations

The Major Standards Organizations

The discussion of the major standards organizations in the textbook is reasonably good, but is in error in few places.

ISO, the International Organization for Standardization, standardizes almost anything one can think of from swimming pools to nuclear reactors, highway signage, standard communication protocols. ISO is a voluntary industry standards body that produces standards, not recommendations. ISO has no relation to ITU other than the usual liaison or widespread between all standards organizations and industry-oriented entities. Even when ISO and ITU produce common standards in an area, meetings are held jointly and do not "feed" into ITU or vice versa. (For example, the number of languages in which their output is published differs between ISO and ITU.)

ITU is a UN treaty organization. It produces recommendations rather than standards. In one of those interesting quirks of legalese, if ITU produced a standard then all of the signatories follow the standard. By generating only "recommendations," countries can choose to ignore the ITU recommendations. U.S. representatives to ITU meetings must be accredited to have no role in choosing delegates to ITU meetings.

ANSI (American National Standards Institute) is the U.S. member body of ISO. U.S. industry organizations act as the sponsors and secretariats of various U.S. committees. The Banking Association is the secretariat for X9, which contributes to the ISO TC 68 on financial services.

IEEE (Institute of Electrical and Electronic Engineers) standards are still considered U.S. standards. IEEE standards are processed through the appropriate ANSI committee and then through international standards.

Most standards organizations have a basic two-tiered scheme. The lowest level consists of the committees in which the work gets done and the compromises are made. There is an approval progression of a document. The second level is considered a technical management level, intended to ensure that an organization does not contradict itself and that the

This structure can have odd quirks. For example, in ANSI one is not allowed to abstain from votes at the lowest level because the members of task groups are "experts" who, it is thus expected that they must have an opinion. However, one is permitted to abstain at the next level up because managers are not expected to be experts on what they are voting on!

All standards committees work through consensus. This means that all ballots conclude with unanimity. This does not mean that members do not vote "No." They do so frequently with a somewhat different implication: voting "No" does not mean you are against the standard. When someone votes (whether "Yes" or "No"), that person also submits comments or concerns, then it means that you approve the document, whether the comments are acted on or not. However, a "No" with comments means that your comments must be added to the fact, most "No" votes will include a list of comments that if accepted will change the vote to a "Yes."

So who gets to vote? This is where organizations differ the most. It depends partly on whether the organization is national or international. ITU has loosened its requirements to include representing governments and/or their telephone companies. In ISO, delegates represent countries supposedly representing the national industry. In ANSI, the lowest-level participation level voting is by company. This prevents a large company from stacking a meeting. ANSI even tries to assure balance by having 3 classes of members: vendors, users, and academic. A requirement for voting that requires one to have attended the last 2 out of 3 meetings to be eligible to vote. This was to ensure that a member had some idea of the issues and was voting for or against something.

The IETF (Internet Engineering Task Force) is unique in that there is no membership—anyone can respond to a call for consensus. For a group as large as the IETF, this open-ended participation is important. It is important in any organization such as IETF that all participants are responsible to someone or some (other) group regarding their votes, and that everyone is responding to. This provides a modicum of check-and-balance that helps maintain stability. One does not want a large company to hire a bunch of consultants who all attend in the position of the company paying them. This is done—and more frequently than one might expect—but the provisions we've discussed above help to minimize this practice.

File: Implications for Management

Implications for Management

The textbook concludes this chapter with the topic of implications for management (page 32). These are quite good and should be studied. But in networking, a key to good management is uncertainty. There has been some interesting work done on this topic.

Managing uncertainty is a critical and little-recognized aspect of organizational management that has an impact on networking and data processing. When done effectively, it is a key to success. High-uncertainty organizations are characterized by any or all of the following:

- Problematic preferences, i.e., preferences that are ill-defined or inconsistent
- Unclear technology
- Fluid participation

In a high-uncertainty organization, decisions tend to be made by either flight or oversight, i.e., postponing decisions until they don't matter or are made for you, or determining unforeseen consequences of other decisions. But the rules of thumb for a typical hierarchical organization are most interesting:

1. Never put high-uncertainty projects in low-uncertainty divisions.

2. For high-uncertainty projects, choose managers more liberal than yourself; for low-uncertainty projects, choose managers who are more conservative.
3. For low-uncertainty projects, hire rigid, structured people; for high-uncertainty projects, hire people who are insecure. In other words, someone who is not sure they know.
4. Intervention begets intervention. Intervention by a manager increases the probability of continued intervention.
5. To avoid this, manage at a level of abstraction commensurate with your level in the hierarchy.

Managers of low-uncertainty divisions will often appear quite competent and assured in their capabilities. They are. But this does not imply and is probably not indicative of all projects. Any of you who have experience in development, ask yourselves how often you have seen a project in trouble moved to be under the management of a very competent manager. Usually the rationale is that the project was too far gone, which may be the case, but one must also consider the effect of putting a high-uncertainty project under a low-uncertainty manager.

In the late 1980s, GM embarked on an ambitious factory-automation project. GM Research developed many new technologies and a pilot plant was converted to use robots. A very successful IT operations company—a company that was very good at low-uncertainty projects. Shortly after the acquisition, GM corporate assumed GM Research's factory-automation effort over to EDS. But GM Research had not yet "written the book" on factory automation and EDS failed. Factory Automation was still a high-uncertainty project something EDS could take on. This is not an uncommon occurrence.

Few if any companies are pure high or low uncertainty. Even low uncertainty companies will have aspects which are high uncertainty. Take a power utility for instance. Maintaining a low uncertainty task to ensure the kind of 6 nines reliability they require. However, the power grid is subject to all kinds of high-uncertainty events from choosing new technology to insulating the high-uncertainty from the low-uncertainty.

The primary purpose of science and engineering is to make high-uncertainty problems into low-uncertainty ones. Doctors, dentists, pilots, "ex-military, and graduates of Jesuit schools describing EDS and Ross Perot etc., tend to make good low-uncertainty managers. (Such statements are always plus or minus one sigma. There are always exceptions.)

[N.B.: There might be a tendency to assign some sort of value to high- vs. low-uncertainty projects or companies, etc. This is unwarranted. There is probably more money to be made in high-uncertainty ones. They are just different. What is important is to understand your talents with respect to them and to know which brings you satisfaction.]

¹According to Gary Wills, describing EDS and Ross Perot.

File: Chapter 2 - Application Layer

Chapter 2 - Application Layer

Introduction

The only reason we have networks is to support applications. Networks exist as an enabling technology that adds business value by allowing business applications to communicate. The process of communicating is complex and that this complexity is managed by software and hardware implemented in layers to provide the necessary functionality. The application architectures and standards govern how applications communicate with each other. By adherence to standards, applications can communicate successfully. Once you understand the communication needs of applications and are familiar with a few application-layer protocols you will be ready to see how the underlying application layer. You will be prepared to read and appreciate the remainder of the textbook.

Objectives

- Understand different application architectures
- Understand specific implementations of these architectures
- Understand basic Internet applications and their protocols

File: Application Architectures

Application Architectures

Application architectures are most easily explained by describing the distribution of four basic processing functions among two or more machines. The four basic processing functions are:

- data storage (basic operating system file management functions)
- data access (database storage and retrieval functions)
- application logic (specific application-dependent logic)
- presentation logic (graphical user interface)

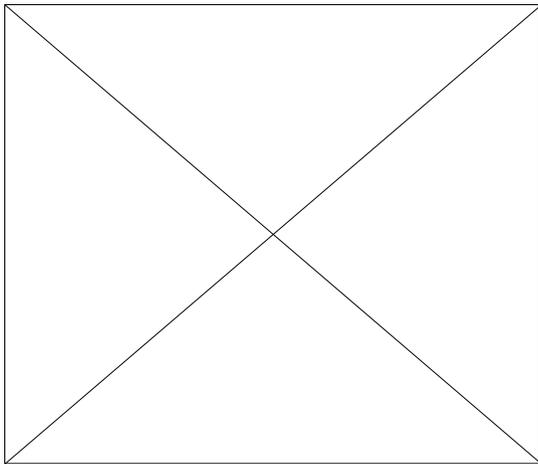
The architectures we will consider include:

- host-based
- client-based
- client-server
- publish/subscribe or notify/confirm
- peer-to-peer

File: Host-Based Architectures

Host-Based Architectures

Host-based architectures put all the processing on a single computer and provide a very simple text-based interface to a dumb terminal.



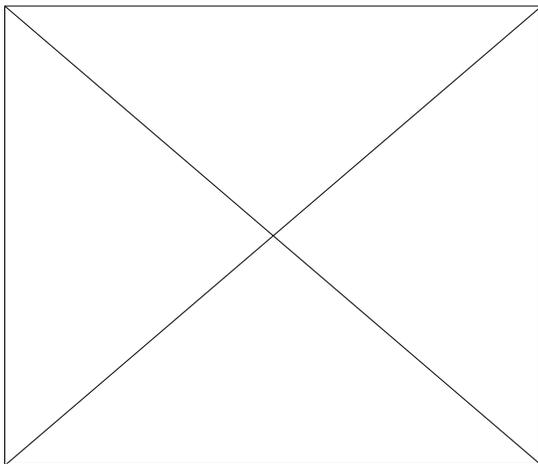
Host-based architectures were popular in the late 60s and 70s with the invention of time-sharing. Time-sharing operating systems allowed many users, each with their own terminal, to use a computer simultaneously. Of course only one user was actually executing on the CPU at a time, however the operating system switched back and forth among all users so fast (approximately 1/1000th of a second) that each user had the illusion that they had exclusive use of the computer. During this era, minicomputers were invented. Low cost minicomputers made it possible to purchase a large number of terminals. Terminals were plentiful and cost effective. The state of networking was relatively immature with simple serial communication being the norm.

Today sensor-based applications, in which sensors (very tiny hosts) collect limited amounts of information that is then communicated to a server for processing, could be considered a form of host-based architecture. Anytime you use telnet, or any other terminal emulation software, to access a server you are using a host-based architecture.

File: Client-Based Architectures

Client-Based Architectures

In the '70s and '80s three important advancements were made in the industry. First, silicon densities became great enough to make microprocessors and low-cost microcomputers. Second, local area networks, such as Ethernet-based local area networks and network operating systems such as Novell, were developed. Third, commercial off-the-shelf software (COTS) became available to meet the processing needs of many organizations. These advancements made it possible to restructure applications to take advantage of the increased processing power of microcomputers to perform the data storage function.

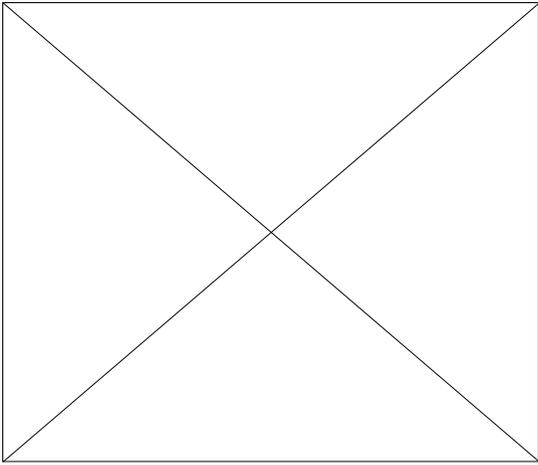


The Novell Network operating system was the de facto standard for providing file and printer services. Files were kept on the server and transferred in their entirety to clients' local drives if necessary. Due to the high cost of laser printers, the network operating system running on the server also supported sharing printers among all the connected client computers.

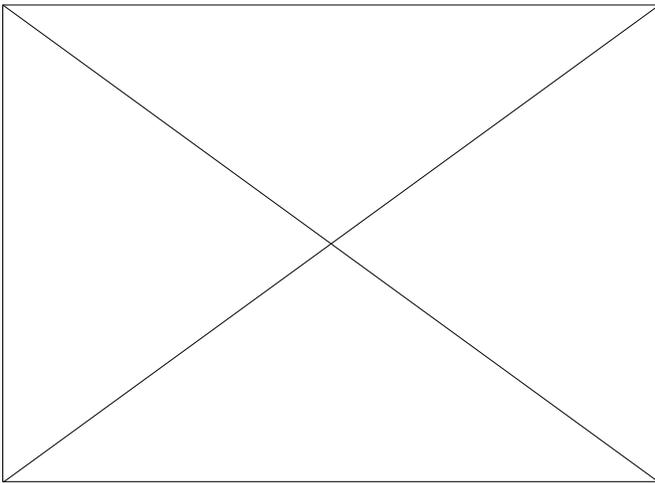
File: Client-Server Architectures

Client-Server Architectures

The client-based application architecture placed a heavy load on the network since entire files were transferred, even if only a small amount of data in the file was actually processed. Hence, client-server architectures were developed to move more of the processing, i.e., the data access logic, to the server and the network. The client made requests (a limited amount of information was transferred as part of the request) of the server and the server responded.



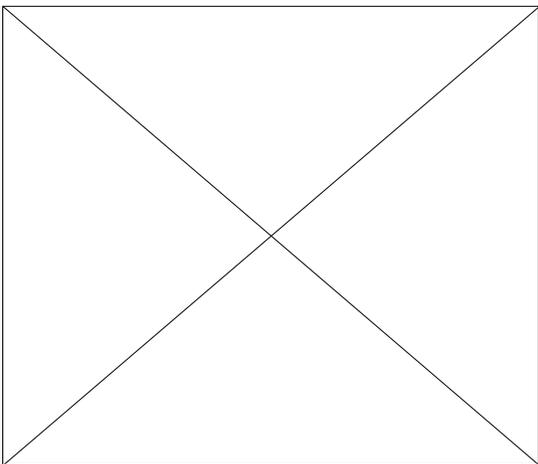
The web, with a web browser as the client and a web server as the server, is a perfect example of this architecture. One of the challenges of this architecture is to standardize it to insure successful communication. Middleware was developed to eliminate incompatibility.



File: Multi-Tiered Architectures

Multi-Tiered Architectures

There are a number of ways to distribute processing functions across machines. The client-server architecture involving two computers is referred to as a two-tier architecture. across three computers we have a three-tier architecture.



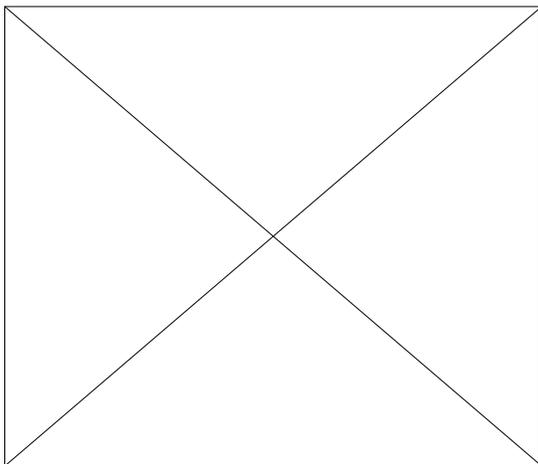
This can easily be extended across n tiers by involving n computers and typically having the application logic distributed over 2 or more computers. The four-tier architecture most modern applications is via a WEB browser. The client runs a browser. The second tier of the architecture is the web server, which then interfaces to the third tier, an appl communicates with the fourth tier, which is primarily a database server.

File: Pros and Cons of the Various Architectures**Pros and Cons of the Various Architectures**

Architecture	Pro	Con
Host-based	Economies of scale. Single point of control.	Host becoming a bottleneck. Host upgrades typically expensive and “lumpy.”
Client-Based	Less expensive to upgrade processing capacity. Rapid application development.	Data traffic must travel back and forth between server and client.
Client-Server	More efficient because of distributed processing. Allow hardware and software from different vendors to be used together.	Difficulty in getting software from different vendors to work together smoothly. May require Middleware, a third category of software.
Multi-Tiered	Better load balancing. More scalable.	Heavily loaded network. Difficult to program and test due to increased complexity.

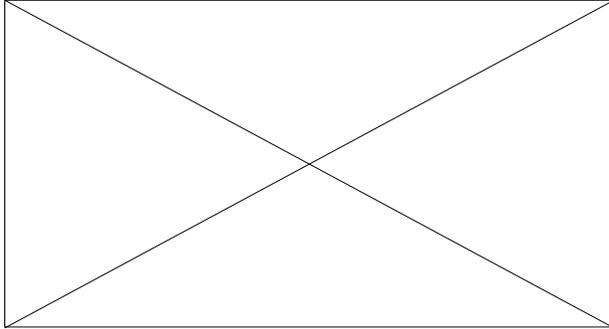
File: Thin and Thick Clients**Thin and Thick Clients**

Another classification system is based on how much application logic resides on the client machine. When there is little to no application logic on the client the client is referred to as a thin client because they are easier to manage (only the server application logic generally needs to be maintained). The best example is World Wide Web architecture, which uses a two-tier architecture. On the other hand contain most of the application logic. This is unpopular because whenever changes are made to the application logic all client machines must be updated.

**File: Criteria for Choosing an Application Architecture****Criteria for Choosing an Application Architecture**

When considering an architecture the following should be considered:

- Infrastructure Cost
 - Cost of servers, clients, and circuits
 - Mainframes: very expensive; terminals, PCs: inexpensive
- Development Cost
 - Mainly cost of software development
 - Software: expensive to develop; off-the-shelf software: inexpensive
- Scalability
 - Ability to increase (or decrease) computing capacity as the network demand changes
 - Mainframes: not scalable; PCs: highly scalable



File: Alternative architectures

Alternative Architectures

Publish/Subscribe

Clients know what to ask for, as well as when to ask for it. Hence all client-server application protocols tend to be request/response type protocols. This relationship does not r driven, based on the server knowing when to supply information to the client as well as what that information is. This architecture is useful for pushing updates to a client - for definitions, etc.

Peer-to-Peer

Client-server architectures are based on a very large number of clients and a relatively few number of servers. Clients make requests and servers satisfy those requests. Servers considerably more processing power and capacity than clients. If we blur the distinction between client and server and allow a single machine to serve both functions and then together via a network we have a peer-to-peer application. The application system then relies upon the computing power of the participants and the bandwidth of the network. formed on an ad hoc basis and varies considerably over time. Such networks are useful for sharing files - audio, data, and video. Peer-to-peer applications have also been built (VoIP); an example of this is Skype.

The challenge with peer-to-peer applications is finding what you want. There needs to be a name service, either centralized (such as the original Napster) or distributed (such as a peer-to-peer object, provides the location of that object. The interaction then takes place directly between the requester of the information and the supplier of the information.

Peer-to-peer architectures have a number of advantages over client-server architectures. First, since all participants supply computing power, the network scales very well by itself. This would be the case in a client-server network.

The robustness of a peer-to-peer network is increased by its distributed nature. Often data is replicated on many peer machines, as is the name lookup system, so that the failure of one machine does not affect the overall network. There are also a number of disadvantages to peer-to-peer networks. First, peers have a transient presence - they come and they go. Second, they tend to be difficult to locate. Third, security becomes more of an issue without centralized filtering on servers. Infected files can easily propagate through the network unless each peer has its own security mechanisms.

File: Application Layer Examples

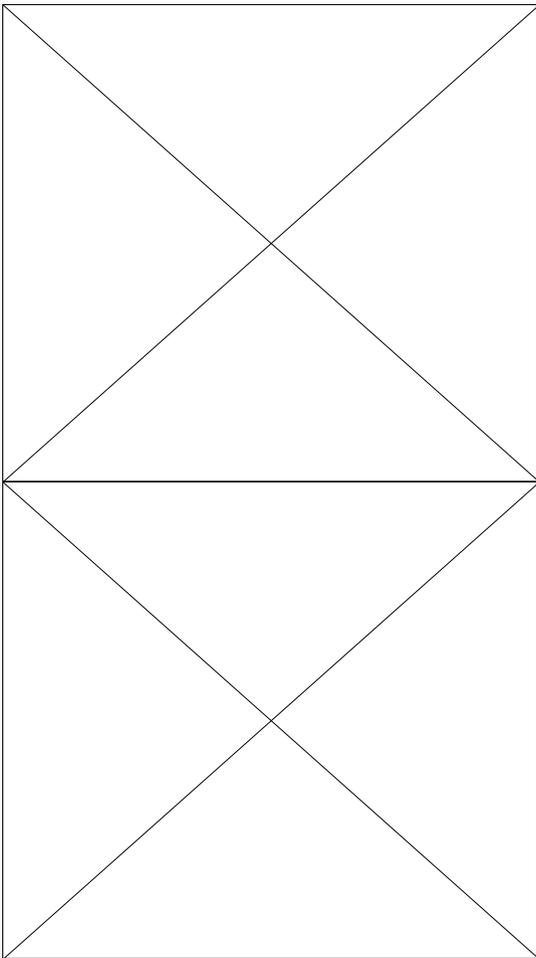
Application Layer Examples

The textbook includes descriptions of numerous application layer protocols and their relationship to the application layer architectures just described. Note the simplicity of the

File: The Web

The Web

HTTP, which is based on a two-tier client-server architecture, is a request/reply protocol - the client makes a request (a get command) and the server responds.



All communication is in the form of ASCII-encoded data. All the real work of communicating is done in the lower layers (transport, internet, datalink, and physical) and the application layer, not network issues. This is the beauty of the layered model.

File: Electronic Mail

Electronic Mail

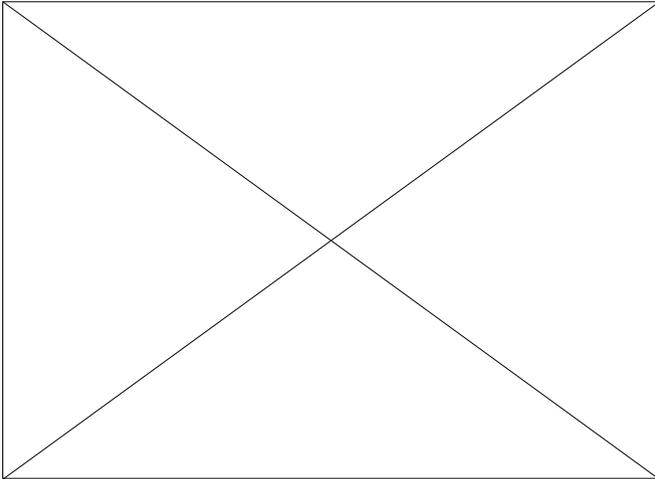
Email is one of the most heavily used applications on the Internet. While there have been numerous proprietary implementations of email, Internet standards and protocols such as SMTP, POP, and IMAP dominate the landscape. Implementation of email systems based on the architectures previously described has evolved. Host-based architectures involving pine, elm, and lmail architectures involving an email client, such as Outlook, Thunderbird, or Eudora, on a client machine communicating with a SMTP server on a mail server machine are very common. To install and configure an email client, WEB-based three-tier implementations have evolved. The three tiers include a WEB browser on the client machine, a WEB server, and a mail server.

File: Peer-to-Peer Applications

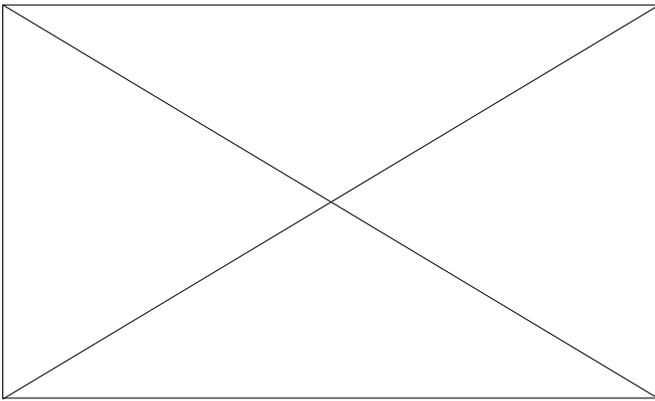
Peer-to-Peer Applications

File sharing

Gnutella is a pure peer-to-peer file sharing application. It addresses the problems of transient presence and transient IP addresses by a technique called viral networking. When a client connects to another Gnutella application on another machine, which already has a connection established to another machine. Whenever queries are generated they are passed through the network. Whenever a file is located, a direct peer-to-peer connection is established for the file transfer. In order to fully address the transient presence and IP address problem, super clients that are always on and have many files to share. While they are not officially servers, these super clients do act like servers.



Early peer-to-peer applications, such as the original Napster, did not subscribe to the pure peer-to-peer approach. They used facilitating servers—and in Napster's case an index server. In Napster, it would upload to the index server a list of file names available for sharing. When other members searched the network the index server satisfied the search from the members.

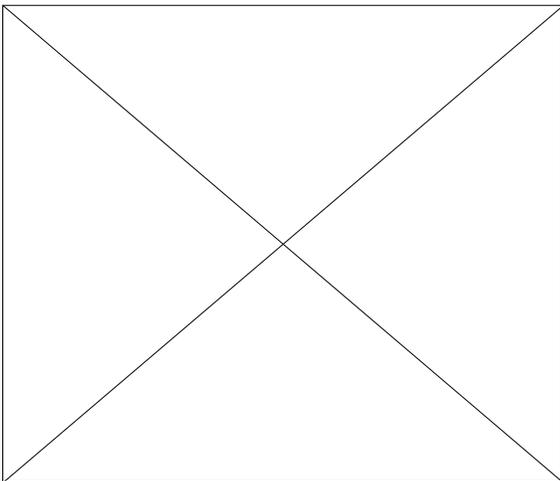


This approach eventually led to Napster's downfall, as they had legal liability for sharing copyrighted material since they maintained their index centrally.

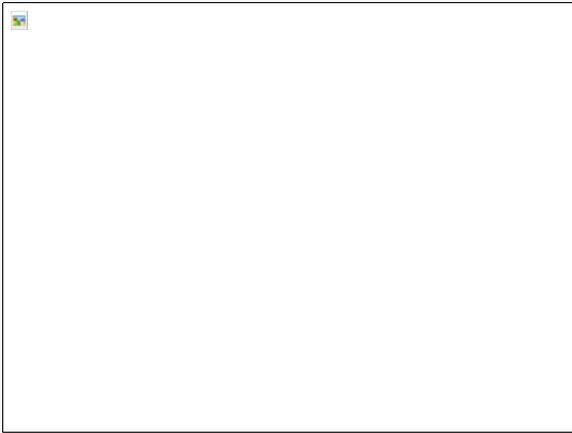
Instant Messaging

Another peer-to-peer application that is extremely popular is instant messaging - the real time exchange of messages in both directions. AOL and ICQ are two of the first wide area messaging systems use servers in three ways.

- No servers - Each party knows or learns the IP address of the other party and connects directly. This can be problematic with dynamically assigned IP addresses



- Presence servers - When an IM application is started, it sends information about its presence to a presence server. When others want to communicate, the presence server notifies them. Then two or more parties communicate directly.

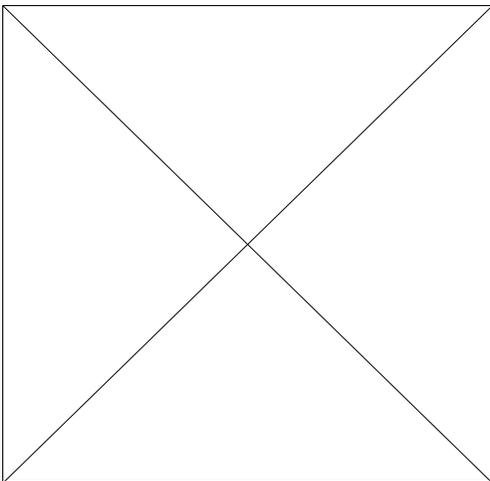


- Relay servers - In some IM systems all messages flow through a central server. This allows for some useful services (i.e., virus scanning) to be implemented.



Voice over IP (VoIP)

One of the newest applications for networking technology is to use the network to carry voice signals in packets. This has occurred as a result of the convergence of telephone significant cost savings by adopting this technology. A VoIP network basically looks like the diagram below.



http://en.wikipedia.org/wiki/Image:Voip_HowItWorks_0203v2.jpg

Cable modems, phone adapters, or a PBX take the phone's digital signal, turn it into bits, and package those bits into packets for transmission over the Internet. Since the Internet delivers real time data without delay, new protocols (such as RTP - Real Time Protocol) had to be developed to meet the needs of VoIP. Also, a signaling protocol to facilitate signaling functions is needed. Session Initiation Protocol (SIP) was developed to address these needs.

File: Chapter 3 - Physical Layer

Chapter 3 - Physical Layer

Introduction

In chapter 1 you learned about the five-layer network model. In chapter 2 you learned about the highest layer, the application layer. In this chapter you will learn about the low layer deals with the representation and movement of individual bits from a computer and/or device to another directly connected computer and/or device in the network.

The purpose of this online material is to provide some additional information and apply the concepts presented in the textbook to solving some specific problems. These problems are in this chapter.

Objectives

- Be familiar with the different types of network circuits and media
- Understand the different types of data and the different types of transmission systems
- Be familiar with analog and digital modems
- Be familiar with multiplexing
- Be able to solve problems involving circuit capacity, data rate, bandwidth, and symbol rate

File: Types of data

Types of Data

Data can be represented in two forms - analog and digital. The human voice is an example of analog data. It varies in both loudness and pitch on a continuous basis assuming that the human ear can hear frequencies from 20 Hz to 20,000 Hz. On the other hand, are inherently digital. Digital data take on a finite number of discrete values. For instance, if we consider the integers between 1 and 10 there are only 10 possible values, i.e., 1, 2, 3, 4, ... 10. The letters of the English alphabet also are digital data since there are only 26 of them with specific values. Digital data is most often displayed by binary digits (binary digits or bits). For instance, the decimal numbers 0 through 7 can be represented in binary in the following way:

Decimal Binary

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Letters of the alphabet are also represented by sequences of 0s and 1s according to various standards. A widely accepted standard is extended ASCII where each letter of the alphabet is represented by 8 bits. Following this standard an upper case "A" is represented by 0100 0001.

Analog data is represented by a continuum of an infinite number of values while digital data is represented by a finite number of discrete values.

How many bits does it take to represent information in digital form? Well that depends on how many different things you want to represent. For instance, if you want to represent the seasons of the year - fall, winter, spring, and summer? How many bits would we need? In this case we have 4 different values we want to represent. We could somewhat arbitrarily say that when the bit was 0, that was the representation for the light being off and when the bit was 1, that was the representation for the light being on. Following correspondences - 00 is fall, 01 is winter, 10 is spring, and 11 is summer. From these two examples you might infer that the more different values we need to represent the more bits we need. The relationship is:

$$\text{number of values} = 2^{\text{number of bits}}$$

So, to represent 26 letters of the alphabet, we would need 5 bits (4 bits could be used to represent 16 different values which is not enough, so we have to use 5 bits even though we need only 26).

File: Types of Signals

Types of Signals

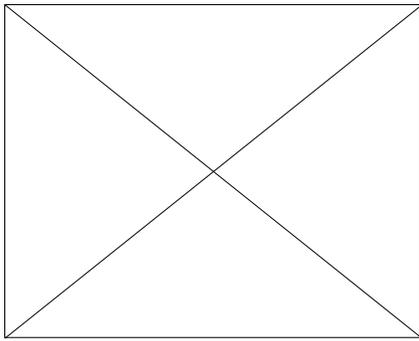
In order to transmit data we must convert the data, whether it is analog or digital, into a signal. This signal might be electricity, light, radio waves, etc. Signals also can be categorized as analog or digital.

File: Analog Signals

Analog Signals

Analog signals vary continuously over time and have an amplitude (strength), frequency, and phase.

The amplitude of a signal relates to how strong it is and is often measured in volts, for an electrical signal, or dB (decibels), for an audio signal.



From the picture above you can see that the signal starts at 0, rises to a positive maximum amplitude, returns to zero, falls to a minimum negative amplitude and then returns to zero. Analog signals repeat this cycle over and over again over time. The number of repetitions per second is known as the frequency, f , of the signal and is measured in Hz (cycles per second). It is convenient to have ways of abbreviating them. For instance 1,000 Hz is also known as 1 kHz. So 33,200 Hz is more conveniently represented as 33.2 kHz. (Move the decimal point "k" in front of the Hz unit of measure).

A million hertz is known as MHz. So 3,999,000 Hz is 3.999 MHz (move the decimal place 6 positions to the left and put "M" in front of Hz). One Gigahertz is equal to a billion hertz (move the decimal point 9 places to the left).

The time it takes for a signal to go through one cycle is known as its period, T . The frequency and the period are reciprocals of each other and are related by the formula.

$$T = 1 / f$$

So, if we had an analog signal of 1,000 Hz (1000 cycle per second), then the period, the time per cycle would be:

$$T = 1 / 1,000$$

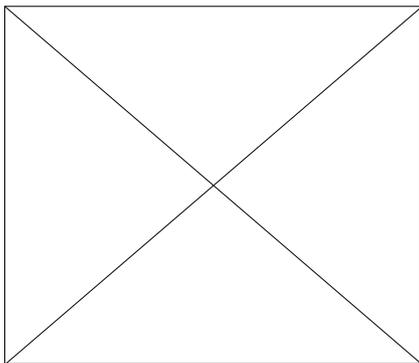
$$T = .001 \text{ seconds}$$

The wavelength of a signal is the distance it travels during the time it takes to go through one cycle. The phase of the signal, which is a number between 0 and 360, is measured in degrees.

File: Digital Signals

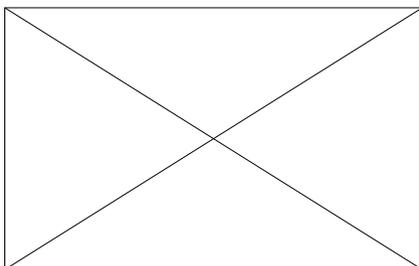
Digital Signals

Digital signals, unlike analog signals, can only take on a finite number of discrete values.



For the picture above, the signal can take on 4 different values. Digital signals are characterized by instantaneous changes in value (vertical lines in the diagram) and a finite number of values (horizontal lines in the diagram). The time during which a digital signal takes on a single value is known as the clock cycle.

A digital signal that can only take on two possible values is known as a binary signal and is shown in the diagram below.



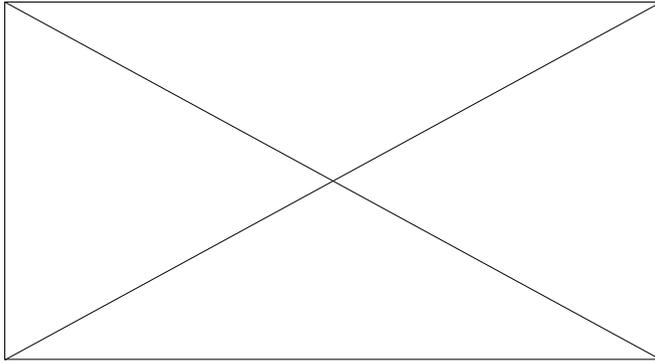
Binary signals have the same characteristics as digital signals, including a clock time - the time during which the signal is kept at a constant amplitude.

Transmission systems

As it turns out, the 2 different types of data can be sent over the 2 types of signals (transmission systems). That is, there are 4 combinations.

- digital signals / digital data
- analog signals / digital data
- digital signals / analog data
- analog signals / analog data

Examples of each are:



We don't have much control over the type of data we have, it is either digital or analog by its nature. However, we do have a choice over how it will be transmitted - either via transmission should be chosen for the following reasons:

- Produces fewer errors
 - Easier to detect and correct errors, since transmitted data is binary (1s and 0s, only two distinct values)
 - A weak square wave can easily be regenerated again in perfect form, allowing higher reliability transmission than analog
- Permits higher maximum transmission rates
 - e.g., Optical fiber designed for digital transmission
- More efficient
 - Possible to send more digital data through a given circuit, circuit can be "packed"
- More secure
 - Easier to encrypt digital bit stream
- Simpler to integrate voice, video and data
 - Easier mix and match V, V, D on the same circuit, since all signals made up of 0's and 1's

File: Metrics

Metrics

Data (bit) Rate (C)

Since we are primarily interested in transmitting digital (binary) information we need a metric that reflects the speed at which binary information can be communicated. That r measured in bits (b) per second (bps).

$$C = b / t$$

Where

C = data rate in bits per second
 b = amount of data transferred in bits
 t = time it takes to transfer b bits

Note that data is quantified in bits (b), not bytes (B). This is specific to networking and very often requires conversion (1 Byte = 8 bits) back and forth depending on whom we then:

$$C = 8 \text{ bits/byte} \times B / t$$

Associated with different circuits will be the capacity of the circuit in terms of data rate. To make large numbers more manageable the following shorthand is used:

1,000 bits/sec 1 Kbps
 1,000,000 bits/sec 1 Mbps
 1,000,000,000 bits/sec 1 Gbps

For instance, a dial-up circuit is capable of carrying roughly 56,000 bits/sec, or 56 Kbps, while a T1 line is capable of carrying 1.544 Mbps.

The expression for data rate can also be used to calculate one of the values, given the other two. For instance, if we wanted to know how long it would take to transmit 1,000,000 bytes using the data rate relationship as follows:

$$C = b / t$$

$$t = b / C$$

$$t = B * 8 \text{ bits/byte} / C$$

$$t = 1,000,000 \text{ bytes} \times 8 \text{ bits/byte} / 56,000 \text{ bits/sec}$$

$$t = 142.86 \text{ seconds or } 2 \text{ minutes } 22.86 \text{ seconds}$$

(Note: You may find it useful to always write the units of the quantities being used and then simplify those units algebraically to make sure your result is in the proper units and factors.)

Sometimes you will be given baud rather than data rate. This is discussed on pages 104 and 105 in the text. Data rate, or bit rate, is the number of bits transmitted per second and a symbol rate, on the other hand is the number of signal value (state) changes that occur each second. For binary signals, where each signal value carries one bit, baud is equal to

So, for binary signals:

$$C = \text{baud} = b / t$$

However, for digital signals other than binary, one signal value can carry more than one bit.

So, for digital signals:

$$C = \text{baud} \times \text{number of bits per symbol}$$

Application of this concept allowed dial-up modems to transmit at faster data rates. Original modems used binary signals and were restricted to about 14 Kbps. More sophisticated modems took on 16 different values which allowed each value to carry 4 bits. Hence, the signal still only changed 14,000 times per second but now its value corresponded to 4 bits of information, 56,000 bits per second.

In case you are wondering what the relationship is between the number of values a digital signal can take on and the number of bits it represents, there is a very simple relationship that we saw earlier applies here as well.

$$\text{number of values} = 2^{\text{number of bits per value}}$$

So for a binary signal where the number of values is 2:

$$2 = 2^{\text{number of bits per value}}$$

$$\text{number of bits per value} = 1$$

And, for a digital signal that can take on 32 different values:

$$32 = 2^{\text{number of bits per value}}$$

$$\text{number of bits per value} = 5$$

Bandwidth

Digital signals are characterized by their data rate. Analog signals tend to be characterized by their bandwidth. So what is bandwidth?

Earlier it was mentioned that analog signals have a frequency associated with them, i.e., 4 MHz or 2.4 GHz. Once we add information to an analog signal (the process of adding information to an analog signal no longer occupies a single frequency, it occupies a range of frequencies. The difference between the highest and lowest frequency is the bandwidth (B) of the signal. The bandwidth of the human voice is from approximately 400 Hz to 14,000 Hz, so the bandwidth of the human voice is 13,600 Hz or 13.6 kHz. AM radio signals have a bandwidth of 200 kHz, and standard TV signals 6 MHz. Why the difference? The more information the signal is carrying the larger the bandwidth must be. FM radio stations sound much better than AM radio stations because they are carrying more information than AM radio signals. But how does all of this relate to carrying bits of information? There are various modulation schemes discussed in the text for doing just this. The greater the bandwidth, the larger the data rate the signal will support. This is not the strength of the signal relative to the strength of interfering signals (noise) will also affect the data rate. You might guess that interfering signals (noise) that are strong relative to the signal would decrease the data rate. Claude Shannon, an engineer and mathematician, determined that the relationship between data rate and bandwidth is as follows:

$$C = B \times \log_2 (1 + S / N)$$

where:

C = data rate (maximum achievable)
 B = bandwidth
 S = signal strength
 N = noise strength

This looks quite complicated, but it really isn't. Forget all the log stuff for the moment. Notice that if the bandwidth goes up (increases), keeping everything else constant, the data rate will double. Without getting into the details of logs, if you increase the signal strength the data rate goes up and if you increase the noise the data rate goes down.

For a specific application of this equation consider a telephone circuit. The bandwidth is approximately 3.4 kHz. The S/N, known as the signal to noise ratio, is approximately

So:

$$C = 3,400 * \log_2 (1+1000)$$

Log₂ (1+1000) is approximately 10

$$C \approx 3,4000 * 10$$

$$C \approx 34 \text{ Kbps}$$

Our text (see page 105) describes the similar relationship from which the same conclusions can be drawn, in the absence of noise. The author states that the symbol rate (baud) is very little noise.

$$\text{Baud} = B$$

If the circuit is noisy the symbol rate may fall as low as 50 percent of the bandwidth. This relationship can be used to calculate the data rate for a specific form of modulation.

Remember that

$$C = \text{baud} \times \text{number of bits per symbol}$$

So

$$C = B \times \text{number of bits per symbol}$$

This relationship can be applied to determine the data rate in a **specific instance when the modulation technique is known**. For instance, if QAM is being used (4 bits per symbol) or 13,600 bps. Shannon's equation places an upper bound on the maximum achievable data rate given a different modulation scheme.

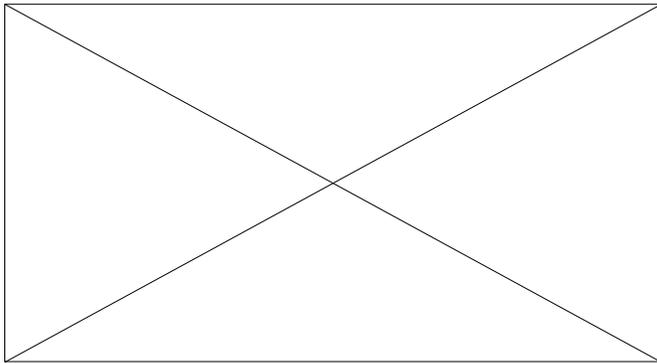
These relationships suggest that it is best to have a very large bandwidth so we can send lots of data very quickly, i.e., so we have a high data rate. However, the bandwidth of the media and devices being used. Generally, the higher the bandwidth the greater the cost. So there are limits—physical and economic.

File: Conversion of an Analog Signal to a Digital Signal

Conversion of an Analog Signal to a Digital Signal

Conversion of an analog signal to a digital signal happens every time you use the telephone. Your voice is converted into an analog signal for transmission from your telephone that you use a hardwired telephone). Once your signal arrives at the local office it is converted into digital form by a device called a CODEC. This conversion is done by a technique (PAM) which involves 4 steps:

- Take samples of the continuously varying analog signal across time
- Measure the amplitude of each signal sample
- Encode the amplitude measurement of the signal as binary data that is representative of the sample
- Send the discrete, digital data stream of 0's and 1's that approximates the original analog signal



In the picture above the analog signal amplitude is measured 13 times. Each measurement is represented by a number between 1 and 8. Since there are 8 possible values of the amplitude, each measurement is represented by 3 bits.

This stream of 3-bit binary numbers creates a rough (digitized) approximation of the original signal. A better representation of the signal can be created by taking more samples more accurately. In either case more bits will be generated. The main questions, then, are how often must the signal be sampled and how accurately do we need to measure the

In 1927 Harry Nyquist, an engineer at Bell Laboratories, discovered that to accurately represent an analog signal, the signal must be sampled at a rate equal to twice the bandwidth whose bandwidth is 4 KHz it must be sampled at 8,000 samples per second, if the bandwidth is 20 KHz it must be sampled at 40,000 samples per second. Telephone circuits take the information content of the human voice falls in this range. Limiting the bandwidth to 4 KHz then sets the upper limit on the sample rate to 8,000. Music on the other hand takes the full range of frequencies in the signal, so a bandwidth of 20 KHz and a sample rate of 40,000 is common. Given that the measurement of the signal's amplitude is not exact, quantizing error. 7 or 8 bits is generally the sample size for voice, while 16 bits is used for music.

Applying this information to determine the data rate generated by converting an analog signal to a digital signal is straightforward.

$$C = N \times s$$

$$N = 2 \times B$$

$$C = 2 \times B \times s$$

Where:

C = data rate
 N = number of samples taken per second
 s = sample size in bits

So, for a telephone circuit:

$$C = 2 \times 4,000 \times 8$$

$$C = 64,000 \text{ bps} = 64 \text{ Kbps}$$

To convert an analog music signal to digital:

$$C = 2 \times 20,000 \times 16$$

$$C = 640 \text{ Kbps}$$

Applying an earlier concept of the significance of data rate, if you wanted to know how much storage would be necessary to save an hour's worth of music in digital form:

$$C = b / t$$

$$b = C \times t$$

$$b = 640,000 \text{ bits/sec} \times 3600 \text{ sec/hour}$$

$$= 2,304,000,000 \text{ bits}$$

$$b \text{ in bytes} = 2,304,000,000 \text{ bits} / 8 \text{ bits/byte}$$

$$= 288,000,000 \text{ bytes}$$

File: Physical Layer Devices and Scope

Physical Layer Devices and Scope

Physical layer standards determine such things as: representation of bits in signals, data rate, distance the signal can travel and still be received properly, type of media the signal travels on (e.g., copper wire, fiber optic, etc.). Hardware is designed to adhere to specific physical layer standards. For instance, your personal computer probably has a network interface card (NIC) in it (network) that has a jack into which a cable can be plugged. This interface must follow a specific physical-layer standard. The other end of the cable that plugs into your pc must also follow the same physical-layer standard. Some devices, such as repeaters and hubs, work at the physical layer. That means that all ports on these devices adhere to the same physical layer standard. If you had a network just comprised of computers and repeaters/hubs, then the scope (that part of the network that works at the bit level) would be the entire network. Repeater and hubs are not very common anymore—switches have replaced hubs. Switches work at the data link layer, which you

Discussion Category: Module 1 Discussions

null

Assignment: Assignment 1

Instructions:

To begin, download the assignment details by clicking on the link below:

[Assignment 1](#)

Submit the completed document using the Attachments tool on this page.

Need Help?

- For clarification about the assignment details, contact your facilitator or instructor via email or the discussion board.
- For help uploading files, view the [Technical Support](#) page in the Syllabus.

Attachments:

Due Date:

March 15, 2010 3:00 AM

Collaboration Type

Individual

Grading Criteria:

out of 100

Assessment: Quiz 1

[Print](#)