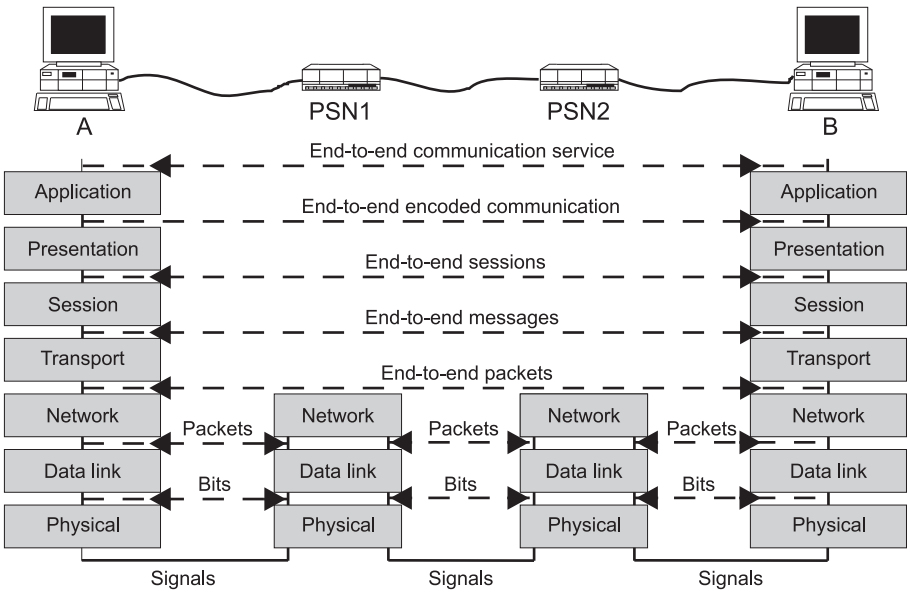
Prof. Dr.-Ing. Firoz Kaderali
Dr.-Ing. Dagmar Sommer, Dr. Thomas Demuth, Dipl.-Ing. Gerd Steinkamp, Dr.-Ing. Michael Stepping

Internet Techniques

Basics, Network Access, IP, TCP, Telnet, IRC, Email, WWW, Usenet, FTP, IPv6



Author

Prof. Dr.-Ing. Firoz Kaderali

1963 - 69	Studied theoretical electrical engineering at the Technische Hochschule Darmstadt
1969 - 74	Assistant at the Faculty of Electrical Engineering at the Technische Hochschule Darmstadt
1974	Doctorate (Promotion) at the Faculty of Electrical Engineering, Technische Hochschule Darmstadt, Subject: Network Theory
1974 - 76	Dozent für Statistische Signaltheorie at the Technischen Hochschule Darmstadt
1976 - 81	Member of the Research Center at SEL (ITT)/Stuttgart. Projektleader of Bundespost study and Fieldtrial DIGON (Digital Local Area Network)
1981 - 86	Head of the Department System Development/Large Systems at Telefonbau und Normalzeit/Frankfurt Development of ISDN-PABXs
Since 1986	Professor for Communications Systems at the FernUniversität Hagen Main interests: Communications Systems, Networks and Protocols; Network Security
1989 - 94	Head of the regional telecommunications initiative TELETECH NRW (Consultant and supervisor of over 120 Telecommunications Projects)
1990 - 96	Member of the ISDN Research Commission NRW (Projects on ISDN Applicationsdesign and Technology Assessment)
Since 1992	Director of the Research Institute of Telecommunications (FTK) in Dortmund, Hagen and Wuppertal, Joint Institute of the Universities Hagen (Electrical Engineering) and Wuppertal (Economy)
1995 - 2001	Member of management of mediaNRW, an initiative to promote the development and spreading of multimedia-applications as a whole and interactive services in enterprises, private households, and the public sector
1999 - 2003	Chairman of the research alliance Data Security Northrhein-Westphalia
2000 - 2002	Chairman of the advisory board of GITS (Gesellschaft für IT-Sicherheit) in Bochum, Germany
Since 2002	Chairman of the open source initiative Campus Source

Table of Contents

1 Basics of Communication Networks

1.1 Goal of the Chapter

This chapter contains a short introduction to the basics of communication networks as far as they are needed to understand the functionality of the Internet. We explain the digitization of information and communication networks and give a brief introduction to switching principles to enable the reader to unterstand the applied switching technique in the Internet.

After an explanation of client server architectures the different types and topologies of networks are outlined. The differences between local area networks, wide area and global networks are presented.

1.2 Introduction

Telecommunication services

Telecommunication services are playing an increasingly important role in many areas of public and business life. The history of mass communication started with the development of the telephone in the 60’ies and 70’ies of the 19th century. The telephone was invented by Reis (1861) as well as by Bell and Grey (1876).

At first, the use of telephones was confined to business life. Later, the extension to communication in private life took place. For example, in Germany, it lasted until the 70’ies of the 20th century that the use of telephone in private areas grew in a large scale. Today, most of the private households in the industrialized nations have telephones, and the trend to mobile personal telephones is developing rapidly.

Till the mid of the 1970’ies telecommunication only grew slowly. Apart from individual communication via telephone, broadcast communication, such as radio and television, had already extended to a large scale. Further, at that time private data communication networks were established.

digital techniques

In the 1980’ies the development of telecommunications proceeded rapidly. The reason for the increasing dynamic was the introduction of digital techniques (see Section 1.3).

Deregulation

Communication networks form a country’s telecommunication infrastructure and are an important economic asset. Over decades these telecommunication networks were under direct control of national goverments who ensured that telecommunication services were accessible for every citizen. The companies that were in charge of operating these telecommunication networks could act as monopolists under the supervision of the state. Although the monopoly was important for the construction of the national telecommunication infrastructure, it became clear that for the sake of diversity of services, internationalization, and economic operation, the control of the state had to be withdrawn. This process is commonly called "Deregulation" or "Liberalization" and was initiated by several countries in the 80’ies (USA 1982, Great Britain 1984, Japan 1985, Germany 1989). Since then, the liberalization of

the telecommunication markets has led to a vast increase in telecommunication services.

Today, communication services are still tied to particular networks:

- voice is transmitted over fixed telephone and mobile phone networks,
- moving pictures are transmitted over broadcast radio and cable TV (CATV) networks,
- computer services for enterprises are carried out over dedicated data networks.

These networks are operated independently from each other. Services which are present in one network are normally not present in another. This situation is currently changing, because customers demand a greater diversity of services independent of the network they are connected to. For example, mobile phone networks were created to allow voice communication independent of location. Today, besides voice services, mobile phones are also able to provide data services like message exchange and Internet services. These are services which were formerly available in computer networks only. The next generation of mobile phone devices will even be able to deal with high quality multimedia services which are now only available in broadcast and CATV networks. On the other hand, the traditional applications of the Internet are e-mail, World Wide Web, file transfer, etc. Currently, new Internet services are emerging which include Internet telephony (Voice over IP) and video on demand. It can be stated, that in the near future, services which are now provided over different networks like public switched telephone networks (PSTN) and computer networks, will be available independent of any particular network. This process is called **convergence of services**.

convergence of services

Today it is imperative for large network operators to keep the costs for provisioning, operation and maintenance of networks as low as possible. This economical pressure combined with the demand of customers for integrated services independent of time and place leads to the **convergence of communication networks** into one single heterogeneous interconnected infrastructure.

convergence of communication networks

In contrast to public telephone networks which were carefully planned and constructed by the former monopolists, the Internet grew rather uncontrolled which led to a diversity in network structures. In the last few years, however, the increase in public and commercial interest in the Internet led to similar provisioning strategies as in public networks.

The success of the Internet, especially the information services provided by the World Wide Web, resulted in a vast increase in global data traffic. A few years ago, the amount of voice traffic was considerably higher than that of data traffic. At the moment, the increase in data traffic fueled by the Internet revolution leads to a paradigm shift from voice-optimized to data-optimized network facilities and services. Due to economic reasons, this transition will take some years. In the meantime, a great part of the Internet data traffic will be transported over networks, which were originally planned to carry voice traffic.

1.3 Basic Techniques in Communication Networks

The development of communication techniques was accelerated by the introduction of digital techniques in the 1980ies. The digitization influenced different aspects of telecommunications:

- digitization of information
- digitization of transmission techniques and
- digitization of switching techniques.

These aspects will be briefly discussed in the following sections.

1.3.1 Digitization of Information

Communication networks transport data representing different kinds of information. Examples are numerical values, text, audio, speech, images and video.

To enable communication, the information to be transferred has to be converted to a certain form which allows the exchange of the information. For example, two people talking with each other generate sound waves which spread out in the air. In this way, the communication partner is reached by speech and processes the received information.

Using an analogue telephone, the way from one person to another is extended. The sound waves generated from the communication partner on one side are transformed into electrical oscillations which are transmitted over long distance telephone lines. On the side of the receiver, the electrical oscillations are converted back into sound waves.

Digitizing information is based on the fact that a finite number of characteristics of a certain feature can be encoded in an appropriate bit string.

A **bit string** is a sequence of finite length, consisting only of the symbols 0 and 1. Exemplary bit strings are 0100 or 000000 or 01101001. The possible usage of bit strings is pointed out in the following examples.

Example 1.3-1:
The 26 letters of the alphabet can be encoded with bit strings of length 5. This length comprises 2^5=32 different bit strings, which are enough to encode the alphabet. The letters could be encoded as follows: A: 00000 B: 00001 C: 00010 .....

Example 1.3-2:
The dots of an image displayed with 256 grey scales can be encoded with bit strings of length 8, as 2^8=256.

Modern digital communication networks make use of bit sequences to transmit information. But to be able to communicate among different partners via bit strings, the partners have to agree on a mutual encoding procedure and format. Without this agreement communication is impossible. Such agreements are arranged using standard formats.

A similar task in the context of digital communication is the digitization of speech.

PCM The most frequently employed principle is **pulse code modulation (PCM)**. In PCM, speech is sampled 8000 times per second. Each of the samples is represented by 8 bits. So, one out of 256 values can be assigned to each sample. Reconverting this digital information into speech comprises quantization errors which are unperceivable for the human ear. Basic use of pulse code modulation is shown in the following example.

Example 1.3-3:
The first picture (Fig. 1.3-1) shows an analogue sound wave which is to be digitized for transmission. The sound wave is sampled over time with a certain sampling rate (see Fig. 1.3-2) The result of this sampling process is a number of discrete values, each of them belonging to a certain time instant. The next step is shown in Fig. 1.3-3 where the discrete sampling values are classified according to the quantization intervals. Each of the discrete samples is assigned to a quantization interval which can be encoded with a certain bit string. Since we have chosen 8 quanitzation steps we can respresent each of them with a string of 3 bits. Fig. 1.3-4 shows the result of the quantization process. The quantized sound wave exhibits the errors due to quantization and sampling (see Fig. 1.3-5) which lead to the mentionned encoding errors (see Fig. 1.3-6). In dependence of the number of quantization steps and the applied sampling rate, the encoding errors can be reduced so far, that the human ear does not conceive them.

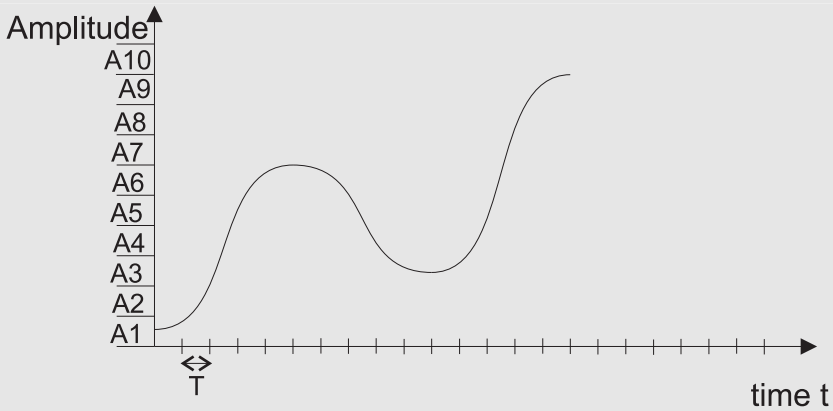


Fig. 1.3-1: The analogue sound wave

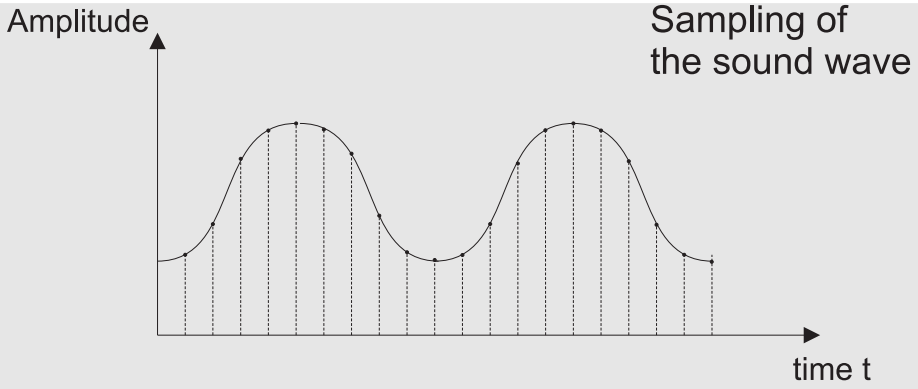


Fig. 1.3-2: Sampling of the sound wave

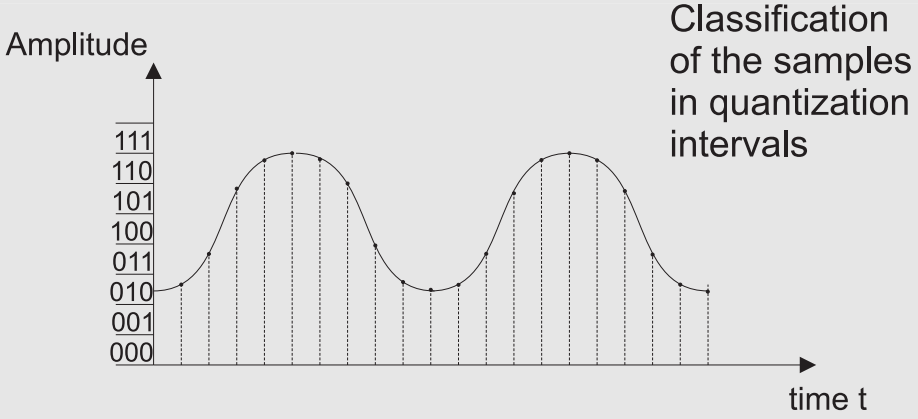


Fig. 1.3-3: Classification of samples in quantization intervals

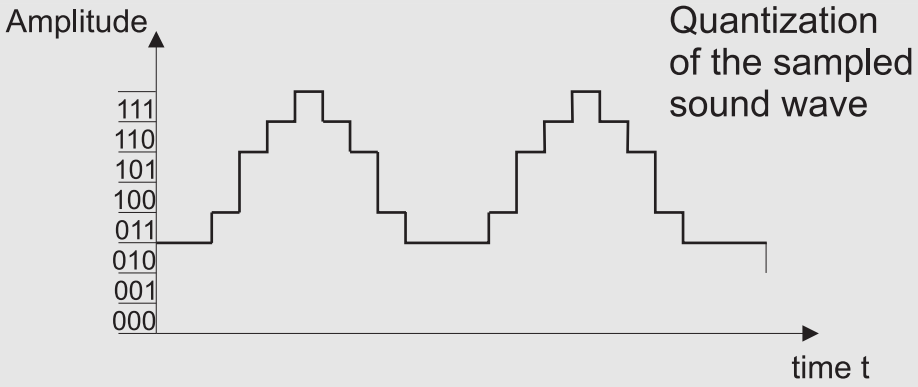


Fig. 1.3-4: Quantization of the sampled sound wave

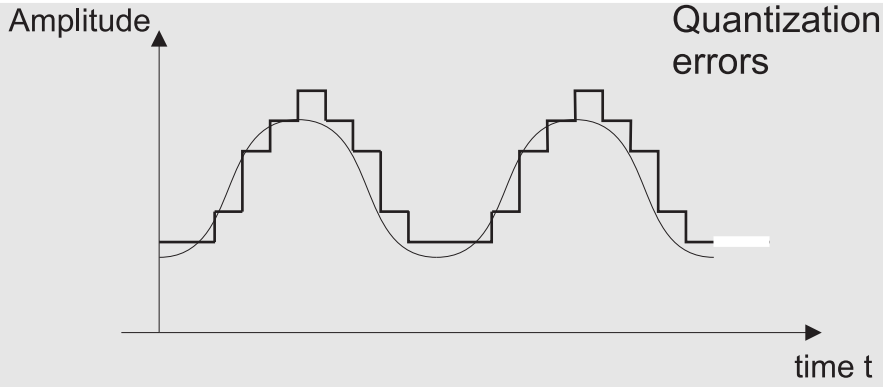


Fig. 1.3-5: Quantization errors

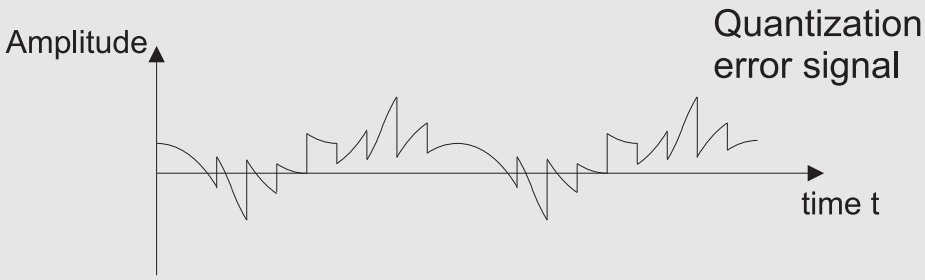
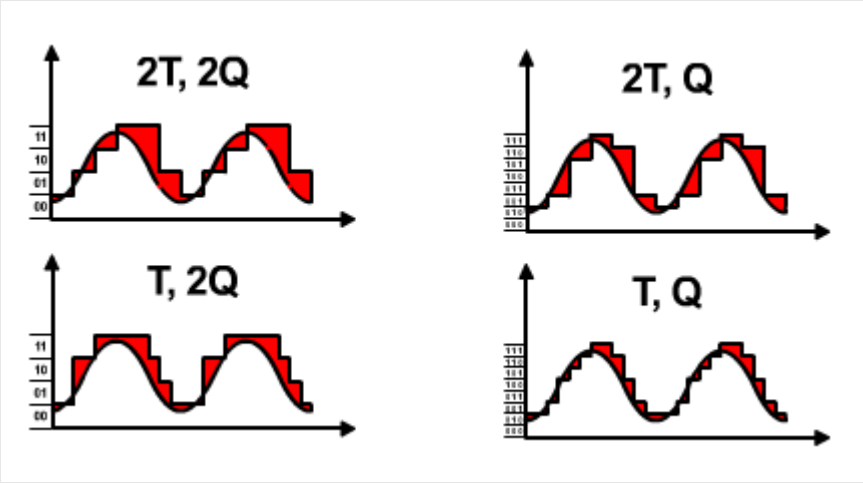


Fig. 1.3-6: Quantization error signal



Animation 1.3-7: Pulse Code Modulation

1.3.2 Digitization of Transmission and Switching Techniques

transmission technique

The notion **transmission technique** comprises technical devices and rules according to which data transmission via a physical medium, e. g. copper wires, coaxial cables and fibre optical cables, is performed. The transmission technique allows communication among two nodes connected via a communication channel. In this context, first some terms commonly used in communication techniques have to be defined:

Nodes are an abstract representation of communication and transmission equipment, e. g. switches, data end equipment.

Links are physical transmission facilities, e. g. copper or fibre optical cables, to interconnect nodes.

In contrast to that, a **channel** means the physical resources of a link which are provided for a communication process between two nodes. **channel**

A**network** consists of nodes and links. The simplest form of a network is shown in Fig. 1.3-8. This network only consists of two nodes, here realized as computers, which are connected via a single link. **network**

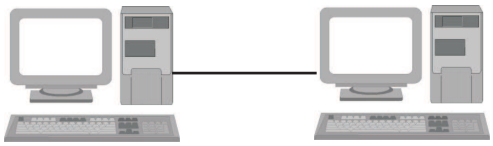


Fig. 1.3-8: Exemplary communication network

Communication in networks is controlled by a complex set of rules, the **protocols**. **protocol**

The main task of the **switching technique** is to connect devices in a communication network. Usually, devices are not directly connected but have to make use of intermediary nodes to be able to communicate with each other. **switching technique**

The reason for using intermediary nodes in communication networks which are responsible for switching is explained in the following:

Two solutions are possible to connect a number of users. In the first solution, each pair of users is connected by a dedicated link (see Fig. 1.3-9). So, each user can directly connect with every other user. But for a large number of users, this solution is not feasible due to the vast cost of the links involved.

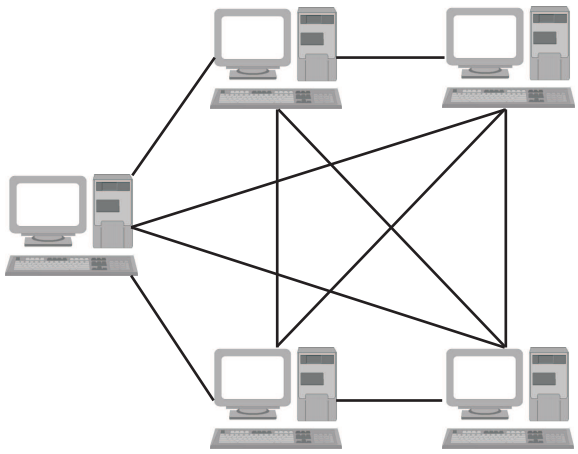


Fig. 1.3-9: Connecting users via point-to-point links

Hence, the alternative solution is that some links are shared among users (see Fig. 1.3-10). This solution achieves significant savings in the amount of required

links. None of the users communicates directly with each other. All of them are connected to intermediary nodes which are responsible for switching the messages among the different users.

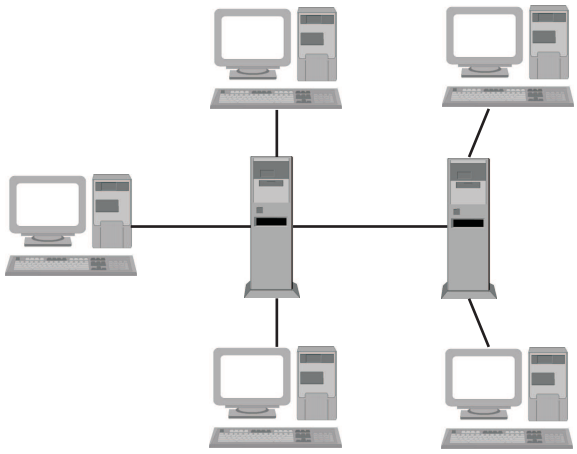


Fig. 1.3-10: Connecting users via shared links

Example 1.3-4:
The same solution is applied in telephone networks. Two communication partners of a telephone network in a city are not directly connected. They have to communicate via the local switching system.

The development of the PCM technique was driven by its cost efficiency compared to analogue systems. At the end of the 1970'ies, this led to integrated switching techniques comprising sampling, quantization, encoding, storage and arbitrary access to the transmitted signal. Thus, transmission and switching were integrated using digital techniques.

Modern networks are based on these digital techniques. For many years, the only exception has been the last mile to the end user, which was still analogue. Since the introduction of ISDN (Integrated Services Digital Network), a growing number of end users in Germany are connected digitally to commmunication networks.

ISDN

advantages of the digital techniques

The advantages of the digital techniques in telecommunications are:

- low proneness to failure
- enhanced security against unauthorized access
- reduced costs
- support of new services

Besides digitization of transmission and switching techniques, the management process has been digitized as well. In modern communication networks, a switching system is realized using one or several interconnected computers.

1.4 Switching principles

Basically, we can differentiate between circuit-switched and packet-switched communication [Kad].

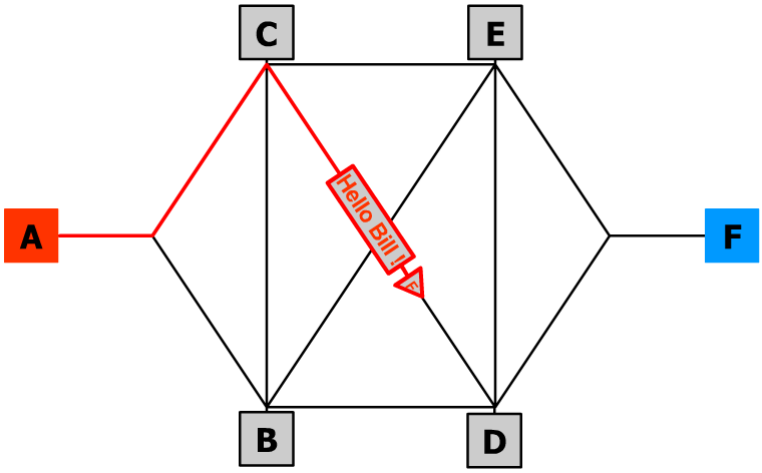
circuit switching

In **circuit switching**, the communication partners A and B exclusively use a communication link or channel across the network for the whole duration of the communication process. Nobody else can use this link or channel at the same time. The switching process can be subdivided into three phases:

1. The set-up phase in which communication partner A indicates to the concerned switching unit that he wants to communicate with partner B. The switching unit informs communication partner B about A's request.
2. The connection phase. In this phase, communication partners A and B exchange information.
3. The termination phase. Both communication partners can inform the switching unit that the communication process is finished.

message switching

In contrast to circuit switching, in **message switching** no direct path of transmission exists between the participants but the message is stored temporarily in intermediate nodes. In case of message switching from participant A to participant B, the message which has to be transmitted is provided with address and control information, temporarily stored in the switch, and possibly after passing several switches transferred to the receiver as one unit.



Animation 1.4-1: Message Switching and Packet Switching

packet switching

In **packet switching** mode, the message which is to be transmitted from participant A to B is divided into packets. Each packet is provided with destination and control information and is seperately transmitted via the network.

Packet switching can be realized in two different operational modes: connectionless and connection-oriented.

datagram

In the connectionless or **datagram** mode, each packet of the communication process is provided with destination and control information as well as a sequence number. The packets are sent to the destination independent from each other. Thus, packets can take different paths across the network and possibly overtake each other. By using a sequence number for each of the packets, the receiver can reorder the packets and reassemble them to the original message. The datagram packet switching mode is also used in the Internet.

virtual circuit

Before starting data transmission in the connection-oriented transmission mode, the path from participant A to B across the network is determined. Thus a **virtual circuit** is set up. Similar to circuit switching we can distinguish three different phases of a virtual circuit: the set-up, the connection phase and the termination. Each transmitted packet takes the same path via the network. In this way, the packets arrive at their destination in correct order and no additional sequence number is required. Packets which belong to a virtual circuit only need a reduced address information to reach their destination. Consequently, the packet overhead is reduced. During the connection phase of the virtual circuit, the links contained in its transmission path are not exclusively available for it alone but may also be used by other virtual circuits in the network as well.

The explained switching principles are illustrated in the following examples.

Example 1.4-1:
Host A wants to send a message to host F. Using circuit switching (see Fig. 1.4-1), a connection from A to F is established. During the connection phase, the link is exclusively available only for the communication partners A and F until one of them terminates the connection.

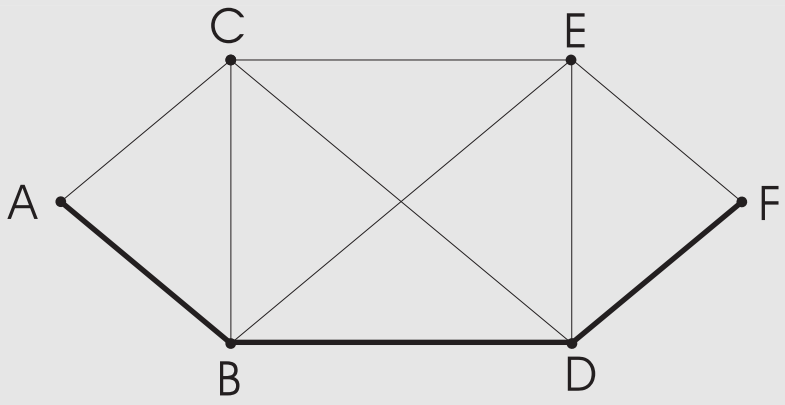
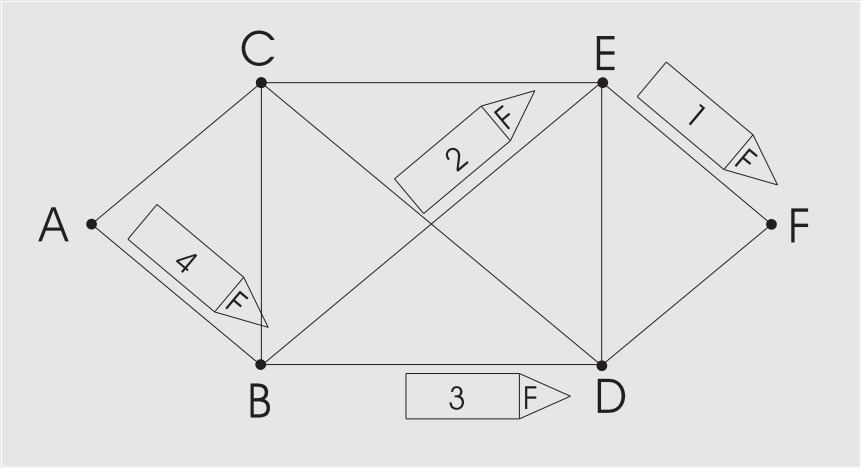


Fig. 1.4-1: Circuit switching

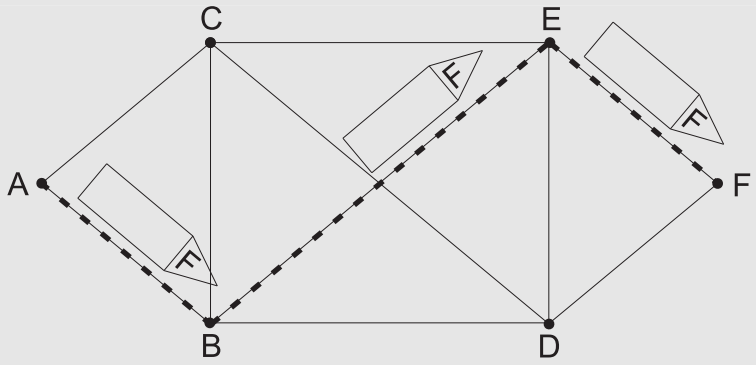
Example 1.4-2:
In a packet-switched network using the connectionless datagram mode (see Animation 1.4-2) the packets are provided with a sequence number as they may reach the destination F in incorrect order. Usually, all the packets take the path

A->B->D->F. But in case of failure of e. g. link B->D or a congestion in host D, the packets may for example take the path A->B->E->F.



Animation 1.4-2: Datagram mode

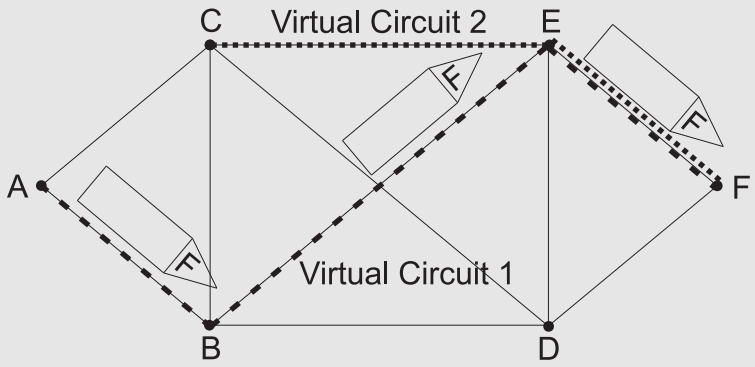
Example 1.4-3:
In a packet-switched network using connection-oriented data transmission (see Animation 1.4-3), first the path from host A to host F is determined. In this case, the path A->B->E->F is chosen. All packets belonging to this virtual circuit will take this path across the network. Therefore they also reach their destination in correct order so that no sequence number is required. If one of the concerned links fails, a new connection has to be set up.



Animation 1.4-3: Examplary virtual circuit

Example 1.4-4:
The links contained in the path of the virtual circuit are not exclusively available for it. They may also be used by e. g. another virtual circuit. In Animation 1.4-4

a second virtual circuit has been added which is established among hosts C and F, taking the path C->E->F.



Animation 1.4-4: Two exemplary virtual circuits sharing a link

route An essential task of switching is to select a path, the **route**, through a communication network to connect the communication partners and to make this path available. A number of different methods have been developed to select the path between a source and a destination. Typical routing strategies are to determine the route in advance or to dynamicly select a route depending on the current state of the network.

The outlined switching principles have shown that a connectionless packet-switched communication network can react more flexibly to failures in parts of the network than a connection-oriented network. This is one reason for the immense growth and the great success of the Internet.

A special form of connectionless packet-switching is employed in local area networks, which are introduced in Section 1.6. In local area networks, the sender transmits a message on the physical medium, e. g. a bus or a ring. All the connected devices receive this message, but only the one to which the message is addressed processes it. So, in local area networks all the connected computers share the physical medium. Therefore, the access to the medium has to be managed by certain media access procedures.

1.5 Client Server Architecture

If two or more computers are connected via a link for the purpose of data communication, in principle we have a computer network.

In the early years of electronic data processing, huge central computers were
host deployed which were also called **main frames** or **hosts**.

Since the early days of the Internet, the usage of the notion host has changed. Today, a ’host’ means a computer which is connected to the Internet. It does not have to be a central computer. In the remainder of the course, the notion ’host’ indicates a computer that is connected to the Internet.

The operation modes of central computers started with batch processing. They further developed to time-sharing. In this case, several computers, the **terminals**, are connected to the central computer via data communication links. The jobs of the individual terminals are processed piecewise in parallel by the central computer.

terminal

In the 1970'ies a trend started which can still be observed today: The technical development has led to a fast decline of hardware prices. Microprocessors with increasing speed and decreasing size are coming up constantly. This has resulted in a growing use of microcomputers allowing the user to work independently, the personal computer (PC).

Although a PC can be used independently, the need to connect computers has not decreased. Due to the growth of the Internet, it has become even more important. Networking computers allows to share periphery equipment such as printers, perform distributed computing, query databases for information retrieval, and take part in different forms of communication.

A distributed system is usually realized in form of a client server architecture. The **server** is a fast computer with a high amount of storage capacity offering its services to the **client**.

server

client

In the Internet, the access to information always takes place according to a client server model: The server provides information that is requested by the user deploying an appropriate client software. To enable server and client to work together, they have to make use of the same data transmission protocol.

Example 1.5-1:
You want to access the homepage of the university of Hagen. This homepage is located on a central computer, the WWW server of the university. You have to start your WWW browser, e. g. the Netscape Navigator or the Internet Explorer. This application is your client software. To receive the information from the homepage, your client software sends a request to the server of the university of Hagen, making use of the appropriate protocol. The server responds by sending the desired information, in this example the homepage of the university of Hagen.

1.6 Local Area Networks, Wide Area Networks

Computer network can be devided into several categories depending on their size:

- LAN, Local Area Network
- MAN, Metropolitan Area Network
- WAN, Wide Area Network
- GAN, Global Network

LAN

In a **local area network (LAN)**, personal computers, workstations, and servers are interconnected locally, e. g. spanning the area of an enterprise or a floor of offices in a large building. A LAN can extend up to about 10 km. The lines of this network belong to the organization considered and are usually managed by the organization and not by a network provider. The number of connected nodes is restricted to several 100. Usually, LANs offer transmission rates from 1 to 100 MBit/s.

A **metropolitan area network (MAN)** extends across a city or a district within a city, across the area of a bigger enterprise or a university. A MAN can extend up to 100 km. Typically, MANs provide transmission rates of about 100 MBit/s up to 1 Gbit/s. A MAN can be used to interconnect several LANs.

WAN

Wide area networks (WAN) connect computers or smaller networks within one or several countries. These connections may be realized deploying public data communication devices and links. The spatial extension of a WAN is unrestricted. Transmission rates can reach up to 2 MBit/s if using ISDN or several 100 MBit/s up to 1 GBit/s if using public broadband networks.

GAN

A **global network (GAN)** connects computers distributed all over the world. It is realized by attaching different LANs and MANs with public or private long distance links. Examples are networks of worldwide distributed enterprises or public networks such as the Internet.

Due to technical reasons, it is not always possible to attach each node to one particular network. Furthermore, it may be preferable to include certain nodes in different networks. This may be to avoid network congestion or to restrict access to resources to ensure data security. Apart from these aspects it may be desired to link comparable or technically different networks together. A special node, a **gateway**, connects technically different networks leading to a heterogeneous network. Gateways can be further distinguished according to the degree of similarity of the networks they are connecting.

gateway

bridge
router

Two similar networks, e. g. two Token Ring LANs, can be attached via a **bridge**. A bridge supports less functionality than a **router** which connects different types of networks such as ISDN with Token Ring.

Local area networks can be distinguished according to three main features:

- topology
- medium access
- transmission medium

The topology comprises the physical structure of the network. Several basic alternatives for physical structures are distinguished which may also be used in combination with each other.

To set up a **star topology**, a switch is located in the centre of the star (see Fig. 1.6-1). All the workstations are connected directly to the switch and communicate with each other across the switch.

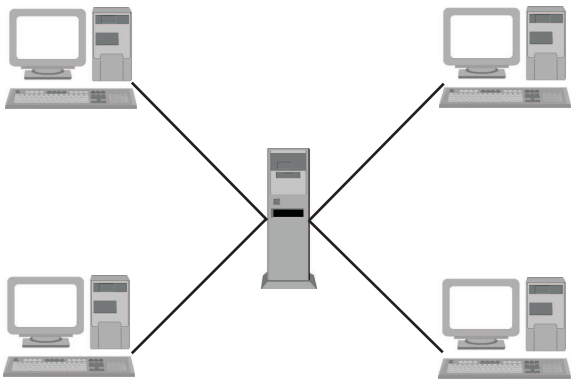


Fig. 1.6-1: Star topology

In **bus topology**, devices are attached passively to the bus (see Fig. 1.6-2). Thus, the failure of one of the attached devices does not influence the rest of the devices connected to the bus. The disadvantage of this configuration is that the bus has to be shared by all connected devices. It is a **shared medium**.

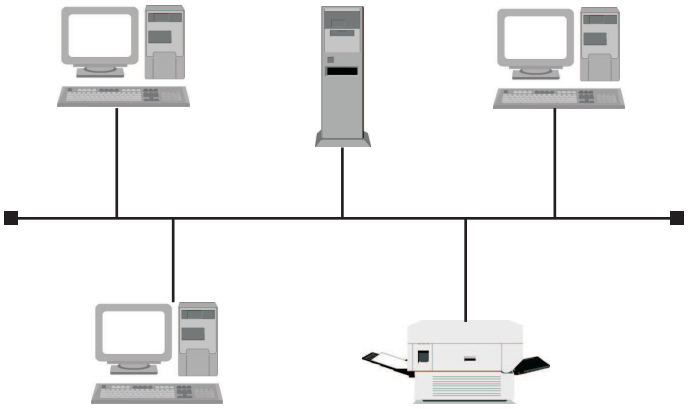


Fig. 1.6-2: Bus topology

In **tree topology**, several networks with bus topology are linked together in form of a tree (see Fig. 1.6-3). This structure is especially useful to realize a network in a building in which each floor runs its own local area network.

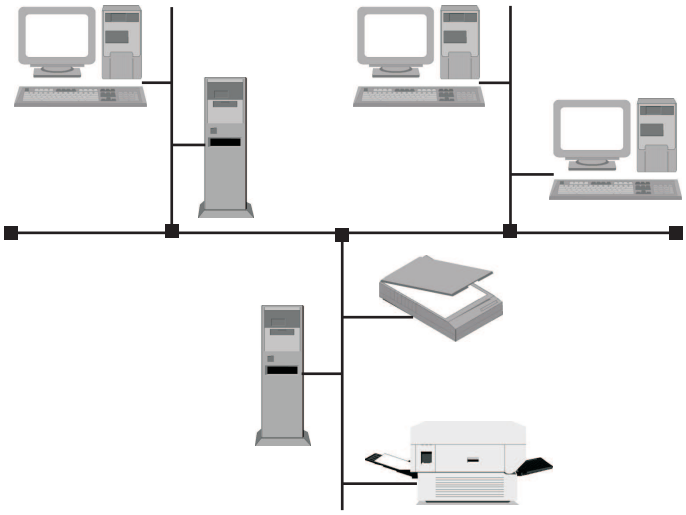


Fig. 1.6-3: Tree topology

In **ring topology**, the devices are inserted into the ring (see Fig. 1.6-4). A node in the ring removes each data packet from the ring and regenerates it to send it to the neighbouring node. The failure of a node thus leads to an interruption of the ring.

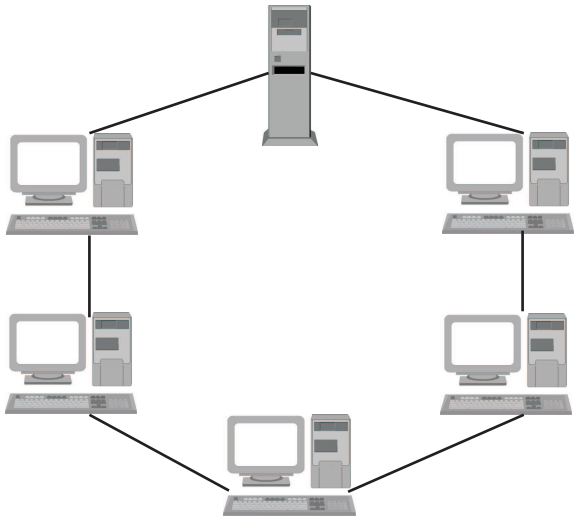


Fig. 1.6-4: Ring topology

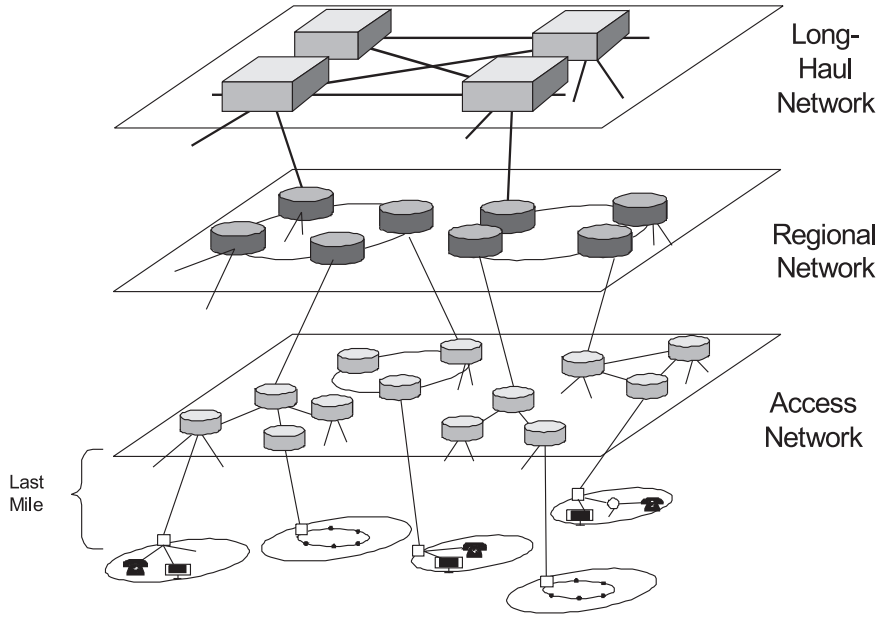
In addition to these basic structures, mixed topologies are realized as well.

The different methods of media access as they are applied in local area networks are not part of this course. These are explained in detail in the course *Communication Networks*.

Possible types of transmission media have already been mentioned. The transmission medium of a local area network may be realized using for example copper cable, coaxial cable, or fibre optical cables.

1.7 Hierarchical Communication Networks

Large communication networks like national telephone and data networks can be divided into different layers as shown in Animation 1.7-1. Such communication networks are able to provide telecommunication services to a large number of subscribers (customers) which are scattered in a wide area.



Animation 1.7-1: 3-layer hierarchical communication network

The subscribers are on the lowest level of the hierarchy, as displayed in Animation 1.7-1. The subscriber can use the complete range of terminal equipment: telephones, private branch exchanges (PBX), multimedia terminals (e.g. TV sets), single computers, LANs, etc.. The task of the upper layers is to connect a subscriber with any freely selectable communication partner. Depending on the distribution of the communication partners, the communication link will traverse one or more of the displayed network layers.

Long-haul networks

Long-haul networks span long transmission distances and connect nodes which are normally situated in a country's major cities. Typical attributes of long-haul networks are the very high data rates (from 34 Mbit/s up to several Gbit/s) and the absence of subscriber lines. Since the complete national traffic is carried over a few high capacity lines, link failure is a very serious threat. Due to the mesh structure there are always several alternatives to route traffic via other links. The nodes in long-haul networks are able to re-route traffic from a failed link within milliseconds. Such short fault times are another important trait of high capacity long-haul networks.

Regional networks

Regional networks which are also called trunk networks form an intermediate layer between long distance networks and local networks. Typically, regional networks posess a ring topology. A fully connected mesh topology would be much too expensive because of the relatively large number of nodes in this network layer. The fault tolerance is still acceptable, since a single link failure will never disconnect a single node from the rest of the network.

Access networks

Access networks cover smaller regions than regional networks, like e.g. a town, and normally posess a star or ring topology. In public switched telephone networks (PSTN) the exchange offices reside on this network layer. The nodes on this layer collect the traffic from the subscribers. Depending on the subscriber's equipment (digital telephone, analogue telephone, cable modem, LAN, etc.) special access technologies have to be employed at the access nodes. For example, for each subscriber using ADSL (see Section 2.4.3) a corresponding ADSL modem has to be available at the access node. The different access technologies subscribers can use to connect to the network are described in Section 2.4. The last part of the network which connects subscribers to the rest of the network is called the "Local Loop" or the "Last Mile".

1.8 The Internet

1.8.1 What is the Internet?

> *"It is safe to say that the Internet is not a static entity, but a complicated matrix of connections in a constant state of upgrade. Thousands upon thousands of players are making changes everyday. The Internet is not a single, monolithic network, either. It comprises an elective series of networks owned and/or operated by thousands of Internet services providers, hundreds of backbone providers, and an assortment of phone, cable, and other communication companies. These many, many separate systems interconnect at some point, thus the 'inter' in 'Internet'" [Boa].*

The basic task of the Internet is to globally transport data from one location to another. This allows communication among users, exchange of information as well as collaborative use of software and computers.

A computer having access to one of the Internet's networks may also access all the other connected networks. The connections can be realized using different technologies ranging from conventional copper wires and coaxial lines to fiber optical cables and satellite communication links.

1.8.2 The history of the Internet

Today's Internet can be traced back to the **Arpanet**. The Arpanet was funded by the Advanced Research Projects Agency (ARPA) in the U. S. Department of Defense (DoD). The Arpanet began in 1966 as an experiment to test the new packet switching technology and protocols that could be used for distributed computing. In 1969, the Arpanet consisted of four packet switching nodes, connecting a few computers and terminals.

Arpanet

Until 1972 when the first package for electronic mail was written, the two main applications in the Arpanet were **Telnet** (for remote computing) and **FTP** (for file transfer). **E-mail** became of growing importance so that only one year later three quarters of the Arpanet traffic arouse from e-mails. In 1975, the first Arpanet mailing list was created.

Telnet
FTP
E-mail

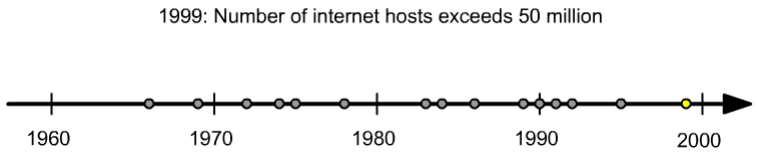
The packet switching technology of the Arpanet was so successful that ARPA also applied it to radio communication (Packet Radio) and satellite communication (SATNET). But due to the different environments of the three networks, certain parameters such as the packet size were different. To be able to integrate these three networks, in 1974, a first approach for a transmission control protocol was published. This proposal was split in 1978 and led to the **TCP** and **IP** protocols providing the foundation of the Internet. The Arpanet became just one of the connected networks. In the years 1982-1983, the Arpanet converted from its original network control protocol (NCP) to the **TCP/IP protocol suite**. In 1983, the name server was developed at the University of Wisconsin. Since then, it was no longer necessary to know the exact path to another system. This led to the introduction of the **Domain Name System (DNS)** about one year later

TCP
IP

DNS

Tab. 1.8-1: *Evolution of the Internet and the World Wide Web [ISC]*

1966	Experiments of ARPA concerning packet switching
1969	Arpanet consisted of four nodes
1972	Development of E-mail
1974	First proposal of TCP
1975	The first Arpanet mailing list was created
1978	Proposal of TCP split into TCP and IP
1983	Arpanet completely switched to TCP/IP
1984	DNS introduced
1986	Internet support extended to general research community
1989	First proposal of WWW at CERN
1990	Arpanet was shut down
1991	First prototype of WWW
1992	Mosaic published
1995	Traditional online dial-up systems begin to provide Internet access
1999	Number of Internet hosts exceeds 50 million



Animation 1.8-1: Evolution of the Internet

The Arpanet was generally restricted to ARPA contractors. But the Internet technology became too useful to remain confined to the defense community. So, in 1980-1981 the National Science Foundation (NSF) extended the support to other computer science research groups. In 1986, the Internet support was extended to all disciplines of the general research community with the NSFNET backbone. In 1990, the Arpanet was finally shut down.

CERN In 1989, at CERN (Centre Européenne pour la Recherche Nucléaire = European Laboratory for Particle Physics), a distributed hypermedia technology to facilitate the international exchange of research information using the Internet was proposed.

World Wide Web Two years later, a prototype of the **World Wide Web** was developed at CERN.

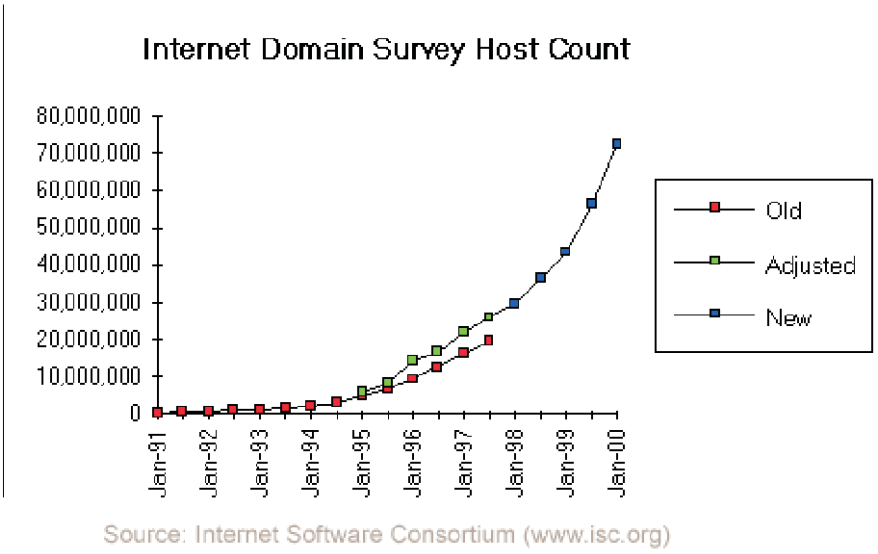


Fig. 1.8-1: Number of internet hosts [ISC]

browser In 1992, the NCSA Center at the University of Illinois developed the first graphical oriented **browser**, Mosaic, which led to an explosive growth of the World Wide Web. In 1995, the first traditional dial-up systems, e. g. Compuserve and America Online, began to provide public Internet access. In 1999, the number of hosts connected to the Internet exceeds 50 million (see Fig. 1.8-1)

1.8.3 Internetworks

Historically, the Internet is a network of independent data networks. These data networks were built by a multitude of different organizations and enterprises whose networks differed in size between LANs and WANs. Besides the differences in size there also existed a diversity in networking technologies. This diversity in technologies is a result of the varying requirements posed by different organisations. For instance, a small company connects its computers with a LAN technology. A large enterprise, on the other hand, interconnects the different sites with WAN technology. Since there exists no networking technology that fits all the needs, a technology that permits the interconnection of multiple heterogeneous networks is required. The scheme that allows the interconnection of different networking technologies is called **internetworking**.

internetworking

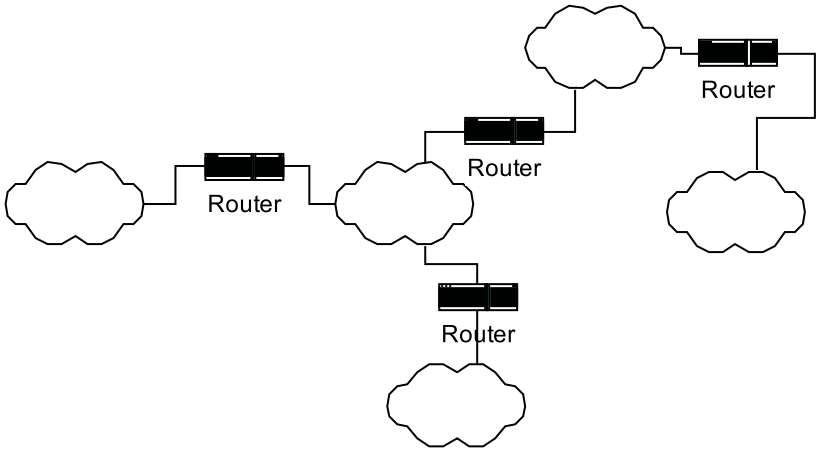
The advantage of internetworking is obvious: it allows the interconnection of different networks to enable communication between terminals connected to these networks. Internetworking can be realized even though the participating networks were not meant to exchange information with other networks in the first place. The global internetwork consisting of thousands of independent networks is called the **Internet**.

Internet

Animation 1.8-2 gives an example of an Internetwork. The binding elements between the different networks are **routers**. A router is a special kind of computer which is connected to multiple physical networks via **interfaces**.The Internet traffic consists of datagrams (see Section 1.4) which originate and end in the terminal equipment of the user and are forwarded by the routers according to routing tables. These routing tables are stored in the routers and specify the path the datagrams will take through the Internet. Users are able to connect to each network and can thus communicate with users connected to the other networks. Since the resulting network infrastructure of an internetwork can become quite large and complex, it is common to represent autonomous networks by a cloud. The cloud is a kind of abstraction which means that the interior of the network is not important in the current context.

router

interface



Animation 1.8-2: Internetwork consisting of independent physical networks which are interconnected by routers. Each network can be a WAN or LAN.

1.8.4 The Internet Architecture

Software and hardware operating on TCP/IP networks typically consist of a wide range of functions to support communication activities. To reliably exchange data between computers many seperate procedures must be carried out [Los99]:

- Package the data
- Determination of the path that the data will follow
- Transmission of the data on a physical medium
- Regulation of the data transfer rate according to the available capacity of the transmission medium and the capacity of the receiver
- Assembly of the incoming data in correct order
- Checking of incoming data for missing or duplicated pieces
- Notify the sender how much of the data have been received successfully
- Delivery of the data to the right application
- Handling of error or problem events.

The network designer is faced with an enormous task in dealing with the number and complexity of these functions.

In the 1970'ies, a number of companies developed computer networks. Each company used a different structure or architecture for its network, as there are many different ways in which network functions can be organized. Despite their differences, the various architectures used in these early networks all were organized according to layers. Introducing a layered model allows to group related functions and implement communications software in a modular manner. A group of related communication functions is called a layer of a communication model.

OSI model

In the late 1970'ies, the International Organization for Standardization (ISO) proposed an architecture model called the **Open System Interconnection (OSI)** model

[Wal91]. The OSI model is a layered architecture dividing network functions into seven conceptual layers.

1.8.4.1 The OSI model

The OSI model was an international effort to create standards for computer communication and generic application services. The OSI reference model permits the interconnection of systems of different origins which respect the standards and protocols of this model.

The OSI model is not concerned with the internal architecture of systems but with their external behaviour. Seven standardized layers correspond to two groups of functions: The transmission oriented layers and the application oriented layers (see Animation 1.8-3).

physical layer

Layer 1 is called the **physical layer** which provides physical support to transfer the data between the end points of a link. It specifies electrical, mechanical, procedural, and functional rules of exchange. This layer is the only one in the OSI model that physically interfaces with a physical layer of another end point.

data link layer

Layer 2, the **data link layer**, permits the error free exchange of data on a communication link. It may also provide link level flow control and synchronization.

network layer

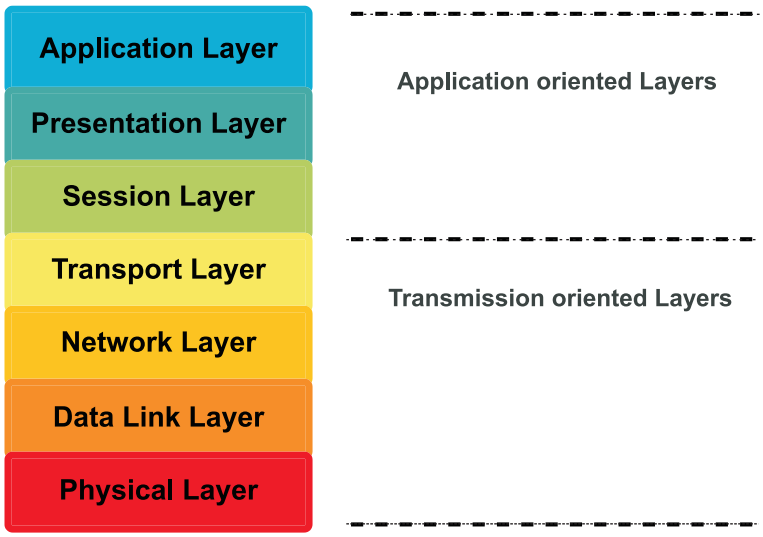
Layer 3, the **network layer,** provides services such as routing, network level flow, and congestion control. It defines the protocols capable of routing the data through one or more intermediate communication nodes.

transport layer

Layer 4 is called the **transport layer** and guarantees a constant **quality of service (QoS)** for data transfer to the higher layers regardless of the type of network actually used. QoS may be defined in terms of delay, loss rate, and priority assignment.

session layer

Layer 5, the **session layer**, defines the organization of the dialogue between distant applications. It is responsible for session establishment between end points. Furthermore, it provides support during connection recovery in case of failure and disruption of communication between processes at the end points.



Animation 1.8-3: The OSI reference model

presentation layer Layer 6, the **presentation layer**, permits systems which exchange data to interpret these independently of their syntactical representation in the system. Presentation layers of end points may exchange control information in order to negotiate a common data format or syntax.

application layer Layer 7, the **application layer,** provides an interface to the user, e. g. an application program such as e-mail or file transfer.

In the OSI model, communication between corresponding layers and adjacent layers is subject to strict rules. A lower layer is service provider to its immediate upper layer. An upper layer is user of services from the lower layer. Neighboring layers are isolated from each other and communicate only by using 'primitves'. **Service**

SDU **Data Units (SDUs)** are employed as the communicating units for the exchange of data and control information between neighboring layers. Communication bet-

SAP ween layers is accomplished by using '**Service Access Points, (SAPs)**. These are ports/addresses used for the exchange of control information and data between neighboring layers. Peer layers at the two end systems communicate (using the

PDU layers below them) through messages called **Protocol Data Units (PDUs)** in order to establish connections and exchange information. PDUs may contain control as well as user information.

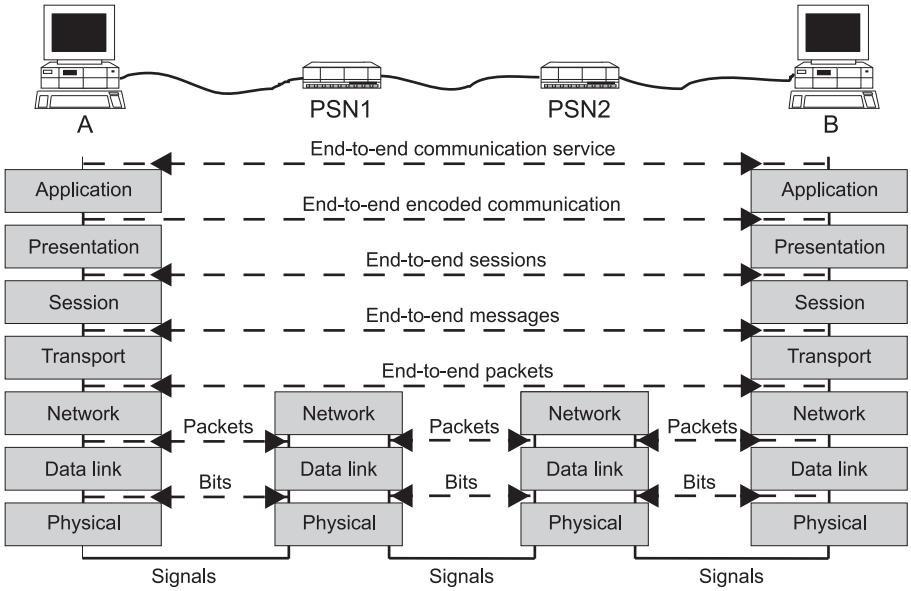


Fig. 1.8-4: OSI model in an exemplary network

To explain the functions of the OSI model more concrete, we want to examine what happens in a network as shown in Fig. 1.8-4 when computer A is used as a remote terminal for computer B. The two computers are connected via two **packet-switching nodes (PSNs)**. In the considered application, the key strokes typed on A's keyboard are sent to B. Messages from B are displayed on A's monitor. Successive keystrokes from A are placed into different packets. Long messages generated by B are divided into smaller packets for transmission to A. Each packet is provided with its destination address so that the PSNs know where to send it. Thus, information from A is put into packets with destination address B. The packets are sent to PSN1 which forwards them to PSN2. PSN2 sends the packets with destination address B to host B. Computer B extracts the pieces of information that it receives from the packets to reassemble the message in its original form. The same procedure is performed from B to A.

PSN

To perform this remote terminal application, a possible scenario takes place as follows: Computer A calls computer B and informs it that it wants to be connected as a remote terminal. B decides whether it accepts the connection and informs A. Assuming that B accepts the connection, A and B exchange information. This reliable form of information transmission requires a number of network operations. In the following, for each of the OSI model layers, these operations are explained in a simplified manner.

The application layer provides a set of commands to the user who may use it for the exchange of messages with a remote computer. The designer of this application software assumes that the application layers of the two hosts are able to exchange messages reliably using the service provided by the lower layers.

The interpretation of the bit strings generated by A's keyboard as well as the correct display of information on A's terminal which is sent by B usually requires some format conversion. The conversion is needed to accomodate the different data representations used by A and B. Such conversion is performed by the presentation layer.

The presentation layer is given the messages by the application layer in A, and converts them into a standard format for their transmission to the presentation layer of host B.

When the connection is first set up, computers A and B agree about some rules for the dialogue. For example they may agree that the communication will take place in full-duplex mode, which means that both computers may transmit simultaneously. Such negotiation is performed by the session layer.

The transport layer controls the delivery of the messages between the end nodes. This layer may divide messages into smaller packets. When the application requires it, the transport layer uses acknowledgements to verify that the packets are received by the destination. In our example, the messages sent by B are divided into numbered packets in B's transport layer. The packets are resequenced in A's transport layer.

Notice that the application, presentation, session, and transport layers perform services between the host computers. They are not present in the packet-switching nodes. The intermediate nodes, e. g. switches, may implement only a part of the model. Usually they only use the three lower layers which encompass the transmission and routing functions, including **error recovery**, **flow control**, **congestion control** and **recovery**, and **synchronization**. The intermediate points are not required to examine or manipulate the information which is exchanged between the end points.

transmission and routing functions

The packets must find their way through the network. This is one of the tasks of the network layer. This layer keeps track of how congested various parts of the network are and it selects the path followed by the packets. In our example, the network layer of host A decides to forward the packets with destination B to the PSN1. Due to the destination address B of the packets, the network layer of PSN1 decides to forward the packets to PSN2 which finally transmits the packets to the network layer of host B.

Each packet is transmitted on a link as a sequence of bits. The data link layer has to check whether the packets were transmitted properly over the link. The data link layer initiates the retransmission of packets that arrived incorretly. In our example, the data link layer of host A has to deal with the correct transmission of the packets to PSN1. The data link layer of PSN1 has to care for the correct transmission of the packets to the data link layer of PSN2, and so forth.

To transmit a packet, each bit is converted into an electrical or optical signal by the physical layer. The signals are sent over the physical link and received at the other end where they are converted back into bits. Successive bits are reassembled into a packet by the receiver. The packet is then passed to the data link layer of the receiver.

1.8.4.2 TCP/IP Layers

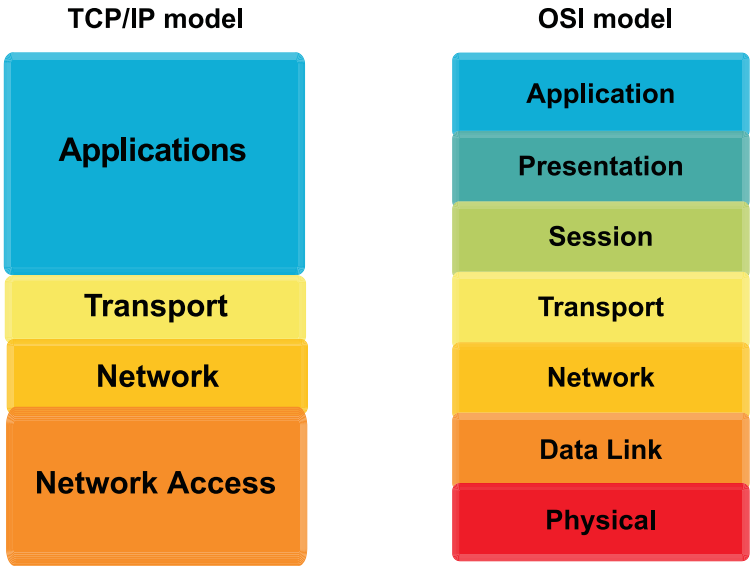
TCP/IP model

There is no official **TCP/IP model** as there is in the case of OSI. But based on the

protocol standards that have been developed, the communication tasks for TCP/IP can be organized into four relatively independent layers:

- Application layer
- Transport layer
- Network layer
- Network access layer

In Animation 1.8-5 the TCP/IP and the OSI layers are shown in comparison to each other.



Animation 1.8-5: TCP/IP model in comparison to the OSI layers

network access

The **network access** layer is concerned with the exchange of data between an end system and the network to which it is attached [Sta]. It is responsible for the access to and transmission of data across a network for two end systems which are attached to the same network. At the data link layer, data is organized into units called frames. Each frame has a header that includes address and control information and a trailer that is used for error detection (see Fig. 1.8-6).

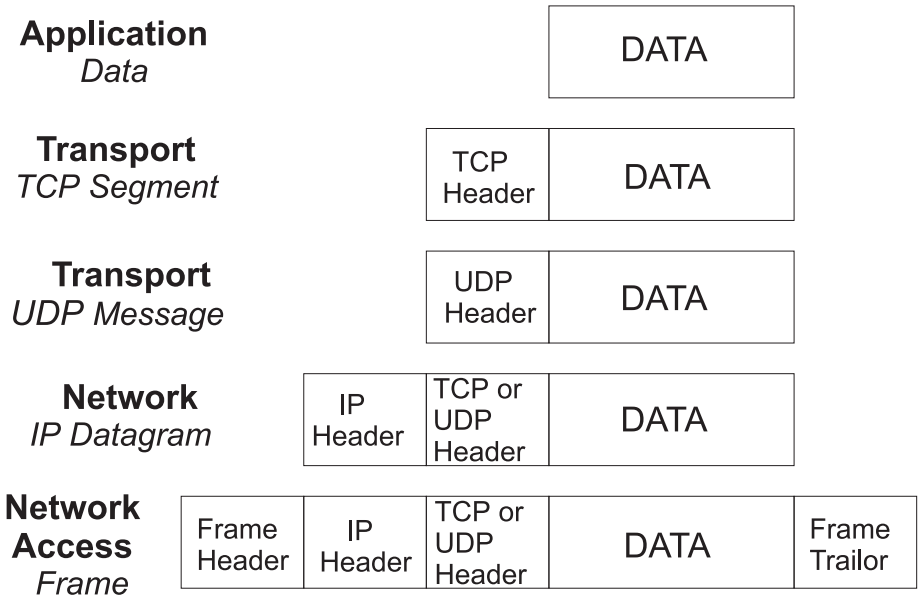
Fig. 1.8-6: Frame format

At the **network layer,**data is routed accross the internetwork. The Internet Protocol (IP) operates at this layer to route packets across networks independent of the network medium. Data may traverse a single link or may be relayed across serveral links in an internetwork. Data is carried in units called datagramms (see Fig. 1.8-7). The IP layer is called connectionless because every datagram is routed independently and IP does not guarantee reliable or in-sequence delivery of datagrams.

Fig. 1.8-7: IP datagram

The **transport layer** manages the flow of data between two internetwork hosts. It relies on two transport protocols, the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). TCP provides reliable data connection services to applications. It contains mechanisms which guarantee that data is delivered error-free, without omissions and in sequence. UDP is classified as a connectionless protocol. It is much simpler and does not offer reliability guarantees, flow control, nor error-recovery measures. It is sometimes used in place of TCP in situations where the full services of TCP are not needed.

In the **application layer**, as comparable to the one in the OSI model, the user interacts with the network application. Data is received as commands from the user and as data from the network application on the other end of the connection. Operations such as e-mail and file transfer are provided in this layer.



Animation 1.8-8: Packaging data for transmission

In Animation 1.8-8 it is outlined how application data is packaged for transmission. An application generates data which shall be transmitted via a communication network. A number of headers are added to the information before it is placed onto a medium in a frame. At the receiver, the incoming frame is decomposed layer by layer. Each header is processed and finally the data is delivered to the destination application [Fei99].

1.9 Summary

After a brief introducition of the history to the development of communication networks and services, the basic techniques in communication networks were explained. The digitization of information, transmission, and switching techniques was outlined and the advantages of digitization were pointed out. Further, the concept of client server architectures were introduced which are also deployed in the Internet.

Regarding network architectures the characteristics of LANs, MANs, WANs and GANs were explained and the differences among these networks types were shown. In this context network topologies and hierachies were outlined as well.

After the description of the history of the Internet its architecture was presented and compared to the architecture of the OSI model.

2 The Network Access Layer

2.1 Goal of the chapter

This chapter gives an overview of some basic technical aspects of the Internet. To understand the problems which arise in data communication the physical properties of transmission media are examined. The first section of this chapter gives a short introduction to the basics of data transmission. A main objective of this section is to show, how the properties of a physical medium, like bandwidth and noise limit the possible data rate. Furthermore, a first introduction to digital modulation is given. The basics provided in this section are essential to understanding data transmission with modems.

Two important roles for the operation of the Internet are are the carrier and the Internet service provider (ISP). In the last few years the Internet evolved from a restricted communication network for military and sciencists to an open telecommunication platform. The last section of this chapter shows the different access technologies that private users can employ to connect to the Internet.

2.2 Introduction to Data Transmission

Computers are digital devices which exchange information by transmitting and receiving bits through the connecting physical transmission medium. Physically, communication systems use electromagnetic waves in form of electric current, radio waves or light to transfer information. This chapter will give an overview of the two basic schemes which can be applied to transmit bits over physical transmission media. These are the direct transmission with bits encoded as voltages which is referred to as **baseband transmission**, and **modulation** which is also called **bandpass transmission**.

Fig. 2.2-1 shows a simple model of a communication system [Pro94]. It can be seen that the communication system consists of a transmitter or modulator linked to the information source, the transmission medium which is also often called channel, and a receiver or demodulator at the destination point.

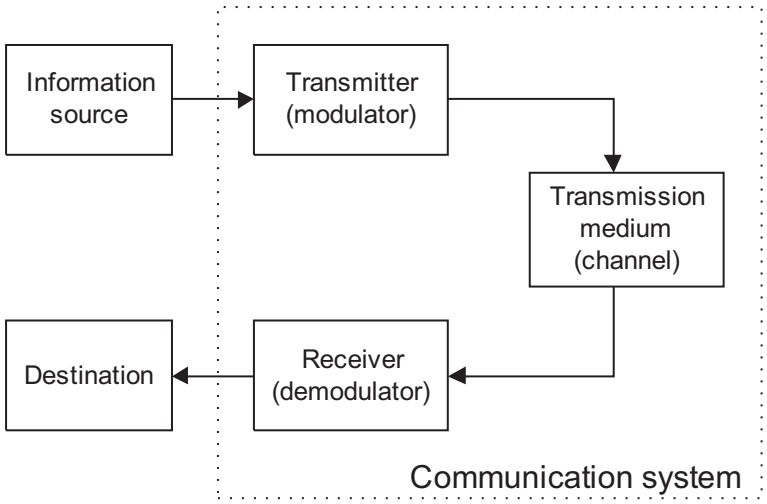


Fig. 2.2-1: Communication system

2.2.1 Short Distance Direct Transmission

A communication process is called **asynchronous** if the communication devices do not need to coordinate before sending data. In practice this means, that the sender can wait an arbitrary time before sending data and the receiver must be ready at any time to accept the data. Asynchronous communication is useful for devices like computer keyboards which can be operated any time by the user. If a key of a keyboard is touched data flows from the keyboard to the computer. As soon as the key is released the data flow stops.

Computers use small electric currents to communicate with other computers or peripheral equipment. A very simple scheme uses voltages to encode bits. The arbitrary scheme as shown in Fig. 2.2-2 uses a negative voltage to encode a 0 and a positive voltage to encode a 1.

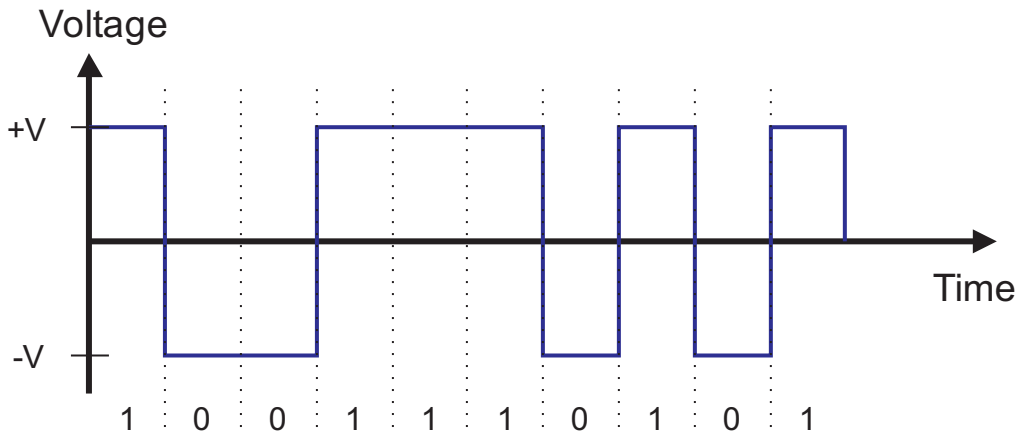


Fig. 2.2-2: Bits are encoded as positive and negative voltages

In this example sender and receiver have to agree on a set of parameters, e. g. the amplitude of the electrical signal (V) and the maximum data rate.

In order to allow the interconnection of devices from different vendors, the specifications of communication systems are standardized. Organisations like the International Telecommunications Union (ITU) and the Institute for Electrical and Electronic Engineers (IEEE) publish specifications for communication systems in documents known as **standards**.

For the **serial** asynchronous communication as described above the standard RS-232-C (commonly abbriviated as RS-232) was specified by Electronic Industries Association (EIA). RS-232-C was also standardized by ITU as V.24 (functions) and V.28 (electrical specifications). RS-232-C is the most widely accepted standard for transmission of characters over short copper cables (max. length of 15 m) between a computer and peripheral devices like keyboard, printer and modem.

The most popular standard for character encoding is the **American Standard Code for Information Interchange (ASCII)**. Standard ASCII uses 7 bits for the encoding of characters which means that 128 different characters can be encoded (codes 0-127). It forms a sort of lowest common denominator for what a character set must support. Since standard ASCII is only adequate for writing American english it can be extended by assigning additional characters needed for other languages . The International Standards Organization (ISO) has defined a number of different character sets based on ASCII that add additional characters needed for other languages. Hence **Extended-ASCII** also codes country specific characters as well as graphic symbols and consists of 8 bits (codes 128-255). The most prominent such character set is ISO 8859-1, commonly called Latin-1. Latin-1 includes enough additional characters to write essentially all Western European languages.

Characters can be catagorized into

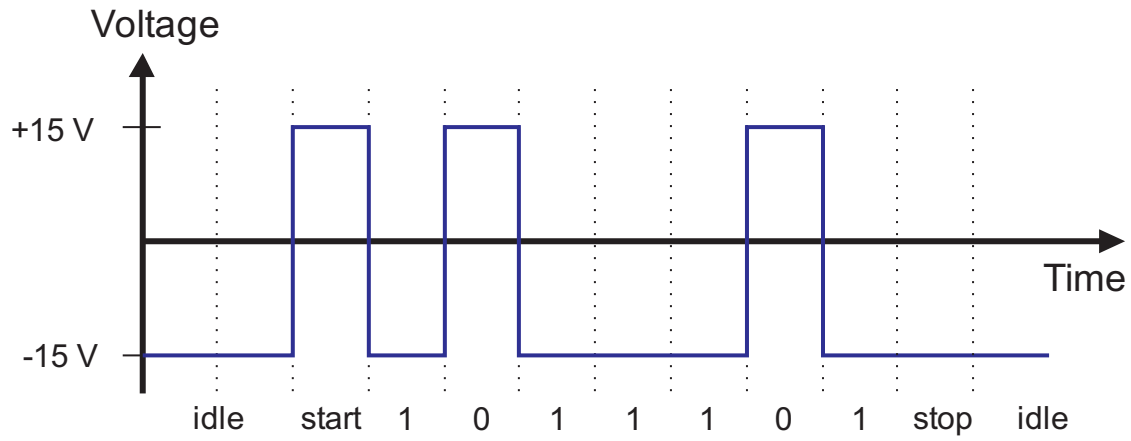
- **printable characters** like alphabetic, numeric and punctuation characters,
- **control characters** - also known as **non-printable characters**, like BS (backspace), LF (line feed), CR (carriage return), SP (space), DEL (delete), FF (form feed).

Standard ASCII is listed in the Appendix.

RS-232 normally uses ASCII characters with a length of 7 bits (8 bits are also possible). The transmission is called serial because the bits are transferred in sequence (as displayed in Fig. 2.2-2). Furthermore, sender and receiver do not coordinate before transmission of a character, which makes the communication process asynchronous. However, once the sending of a character has started, all bits are transmitted one after another without delay. In Animation 2.2-3 the RS-232 encoding of a character is shown. It can be seen that the RS-232 transmission differs in some points from the arbitrary scheme presented in Fig. 2.2-2. Animation 2.2-3 shows

that RS-232 uses negative 15 V[1] to represent a 1 and positive 15 V to represent a 0. Since the idle state is also represented by -15 V and must be distinguished from a 1 an extra 0 bit which is called **start bit** has to be inserted prior to transmitting a character.

The idle period after the transmission of a character can last arbitrarily long but is restricted to a minimum time. This minimum idle time is achieved by inserting an extra 1 bit which is called **stop bit** at the end of a character. Thus the complete transmission requires 9 bits although the original character only consists of 7 bits.



Animation 2.2-3: RS-232 transmission of a character

The number of bits transmitted per second using a transmission scheme is called its **bit rate**. Since not all bits are actually used to convey information like the start and the stop bits, it is useful to define the term **data rate** which accounts for the net rate at wich information bits are transmitted. In the case of 7 bit RS-232 transmission with start and stop bit the data rate is 7/9 of the bit rate.

The typical bit rate for RS-232 transmission lies between 300 bit/s (early modems) and 20 kbit/s. It can be seen that RS-232 transmission is carried out over short copper lines with low bit rates.

1 The RS-232 standards allows voltages between 3Vand 15V

Exercise 2.2-1:

The following character string shall be transmitted employing ASCII coding (the ASCII table is given in the Appendix) and RS-232 transmission:

`Feu`

a. Look up the decimal ASCII value for the three characters and convert them into binary representation.

b. Draw the resulting RS-232 signal diagramm.

Note: The least significant bit (LSB) is transmitted first.

Example: Given is the following bitstring: 1110. Here the left 1 is the most significant bit (MSB) and the 0 to the right is the LSB.

2.2.2 Limitations of physical transmission media

To understand the problems which arise in data transmission, it is essential to explain the following basic terms and their relationship to each other:

- bit rate,
- bandwidth of a transmission medium,
- noise on a transmission medium.

When a signal is transmitted over a physical medium, it does not look as perfectly shaped as displayed in Fig. 2.2-2. In fact, the real shape of a rectangular signal will look more like that presented in Fig. 2.2-4. The following details can be observed:

- the slopes of the real signal are not as steep as the slopes of the ideal signal due to limited bandwidth of the transmission channel
- the amplitude of the signal is smaller due to attenuation
- the shape of the real signal is broadened due to dispersion
- additional noise interferes with the original signal

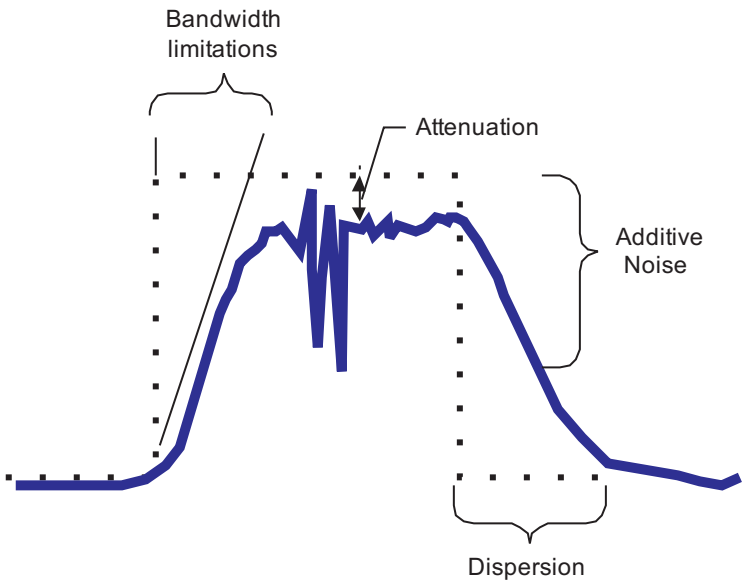
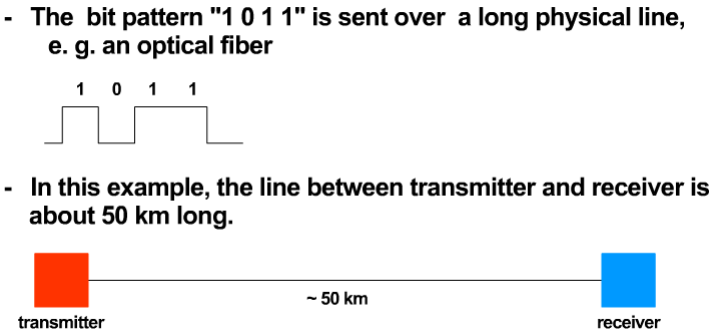


Fig. 2.2-4: Ideal (dotted) and real signal on a transmission line

Attenuation causes the signal to loose energy while it travels down a transmission line. A weak signal is harder to detect for the receiver. In practice, attenuation limits the possible length of a transmission line. For long-haul lines this means, that the signal has to be reconstructed by intermediately spaced amplifiers (repeater).

Dispersion is a kind of delay distortion which occurs because the spectral components of a signal travel at different velocities. As a consequence, the signal broadens and different parts of the signal start to interfere with each other. This phenomenon is called **intersymbol interference** (ISI).



Animation 2.2-1: Dispersion on long transmission lines

An ideal transmission system does not output any electrical signal in absence of an input signal. In practice, however, there are several sources of background interferences wich perturbe the original signal. In communications engineering these undesired disturbances are called **noise** and can occur in diverse forms:

- **Thermal noise** is present in all electrical devices and transmission media. It occurs at all temperatures above absolute zero and can be seen as spontaneous fluctuation due to thermal interaction between the free electrons and the vibrating atoms in a conducting medium. It is made up of random spectral components of continously varying amplitude.

- **Crosstalk** is is caused by unwanted electrical coupling between adjacent lines. This means that a signal transmitted over a line is picked up by another line which runs next to it. A typical crosstalk effect is when another call can be heared in the background during a telephone call.

- **Impulse noise** is picked up from the surrounding environment and consists of short but heavy bursts. It is often caused by switching of electronic devices (man-made noise) but can also be provoked by lightning discharge during a thunderstorm (atmospheric noise).

The rate at which data can be transmitted over a channel is vastly influenced by its **bandwidth**. In the real world no device can change a physical signal instantly (this also holds for biological systems). The bandwidth of a transmission medium is directly related to the speed at which bits can be sent over that medium. If the sender attemps to transmit changes faster than the bandwidth of the channel allows, some of the changes will be lost. Bandwidth is measured in **cycles per second** [1/s] or **Hertz** [Hz]. The term bandwidth is not uniquely defined, and can be different depending on the application.

Two common definitions can be seen in Fig. 2.2-5. Basically, the bandwidth of a channel specifies, which spectral components of a signal will pass the channel without attenuation. All spectral components at higher frequencies than the cut-off frequency f_c will be dropped by the channel. This channel model can be seen in Fig. 2.2-5a. In practise, channels with such a rectangular shaped frequency response function are not possible. Real world channels will have a response function as displayed in Fig. 2.2-5b. Here, f_c specifies the frequency at which the amplitude of the signal will be reduced by a factor of $1/\sqrt{2}$ (3dB - see Appendix).

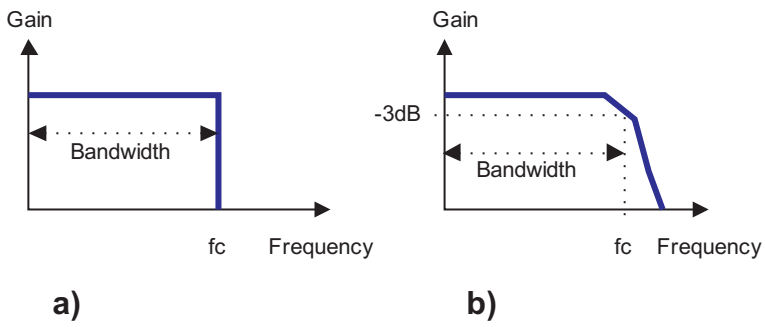


Fig. 2.2-5: Two alternatives for the definition of the bandwidth of a channel

2.2.3 Long Distance Transmission - Digital Modulation

There are many occasions where the above mentioned simple transmission scheme like RS-232 does not work:

- For long distance links, the attenuation is too high for efficient transmission.
- The scheme does not work when the signal has to be transmitted over channels with certain frequency and bandwidth constraints, e. g. over radio channels. In this case the data signal has to be matched to the channel properties.

Researchers have discovered, that a continuous, oscillating signal will propagate further than other signals. The most basic oscillating signal is a sine wave. In communications engineering sine waves are used as **carriers** for other signals, like TV-signals, signals from radio stations and also data signals. In order to transmit information on top of a carrier signal, it has to be modified by the information signal. This modification is known as **modulation**. A sinusoidal carrier signal can be written as follows:

$$s(t) = Asin(\omega t + \phi) \qquad \text{2.2-1}$$

It can be seen from Equation 2.1 that the sine carrier posesses three parameters which can be modified by an information signal. These are its amplitude A, its frequency ω, and its phase ϕ. A carrier signal and its parameters are displayed in Fig. 2.2-6.

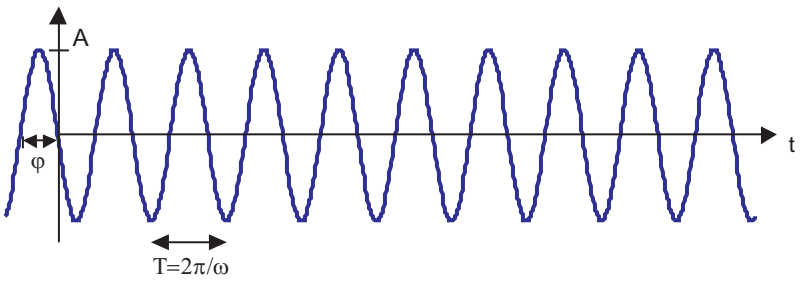


Fig. 2.2-6: Sine carrier signal

According to the carrier's parameters, there are three alternatives for modulation. These are **amplitude modulation** (AM), **frequency modulation** (FM), and **phase modulation** (PM). Amplitude modulation varies the strength of the outgoing signal in proportion to the information beeing sent. Frequency modulation varies the frequency and phase modulation the phase of the carrier in proportion to the information signal. The terms AM, FM, and PM are commonly used for modulation with analogue information signals, like in television and radio applications

When modulation is used to transmit digital information, like data from a computer, the transmission is called **digital modulation**. In digital modulation the carrier signal is switched (shifted) proportional to the binary signal as it alternates (keys) between binary 1 and 0. According to the three modulation types presented above the three basic digital modulation types are **amplitude-shift keying (ASK)**, **frequency-shift keying (FSK)**, and **phase-shift keying (PSK)**. Fig. 2.2-7 shows how a binary signal is transmitted by using digital modulation.

It can be seen that ASK uses two different amplitude levels to represent binary 1 and 0. The carrier signal is shifted between these two amplitude levels according to the binary data signal. In FSK, the two binary states are represented by different frequencies. The carrier is switched between the two frequencies according to the binary signal. Finally, in PSK the frequency and amplitude of the carrier is kept constant while the carrier is shifted in phase as each bit in the data stream is transmitted. The scheme presented in the example is called **coherent PSK** where the carrier uses a 180 degree phase difference to represent the change from binary 0 to binary 1 and vice versa.

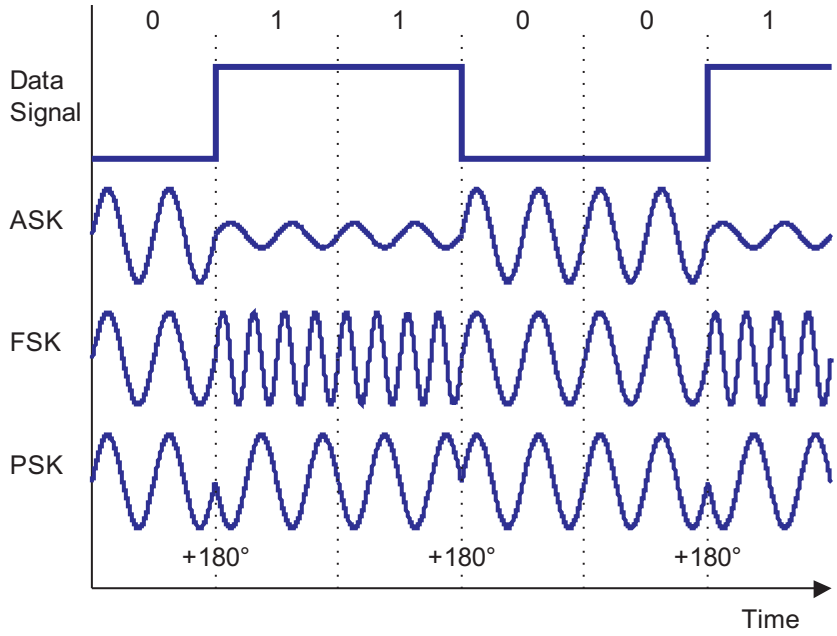


Fig. 2.2-7: Digital Modulation

In the modulation schemes presented above the physical carrier signal is only switched between two different states. To increase the bit rate it is possible to add more levels to the carrier signal.

For example, instead of using two different amplitude levels, four amplitude levels can be used to represent a combination of two bits or eight levels can be employed to represent 3 bits. The bit rate is then two or three times the signalling rate (the rate at which the carrier signal changes). The signalling rate is also called **baud rate**. The relation between signalling rate and bit rate can be expressed as follows:

$$R = R_s \log_2 M \qquad 2.2\text{-}2$$

where R is the bit rate, R_s is the signalling rate, and M is the number of signal levels.

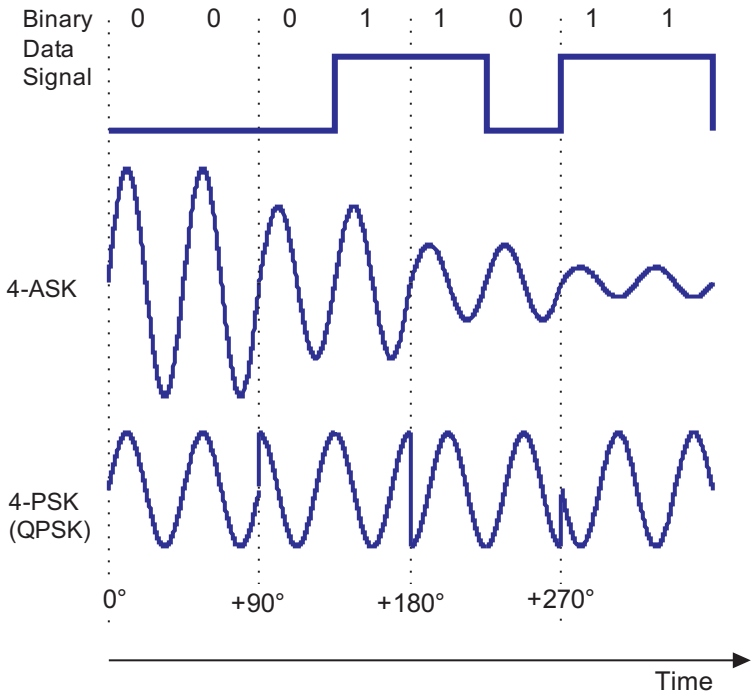


Fig. 2.2-8: Multilevel modulation: 4-PSK (QPSK) and 4-ASK

(Exercise 2.6.2)

Exercise 2.2-2:

The characters "Feu" are transmitted employing QPSK.

a. *a) Convert the 7 bit characters into 8 bit characters by inserting a "0" as MSB. The LSB is transmitted first.*

b. *b) Draw the resulting QPSK signal (according to Fig. 2.2-8 in the course text).*

c. *c) What is the baud rate for a bit rate of 2400 bit/s?*

The multilevel signal can be plotted in a diagram that shows the possible constellations of amplitude and phase levels. Such a diagramm is called **constellation pattern**. The constellation pattern of QPSK is shown in Fig. 2.2-9.

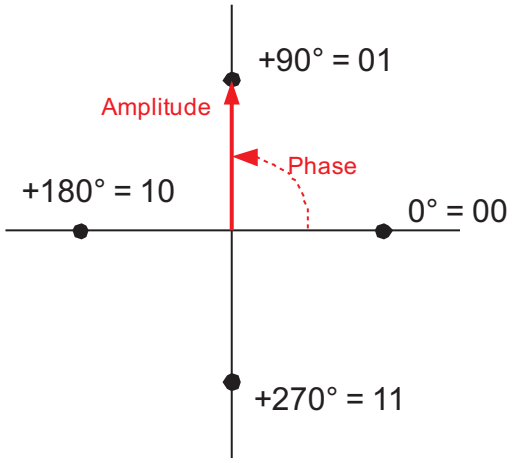


Fig. 2.2-9: Constellation pattern of QPSK

The arrow from the origin of the coordinate system to the constellation point represents the sinusoidal carrier and is a vector. The length of this vector corresponds to the amplitude, the angle between 0degree and the vector corresponds to the phase of the constellation point. Hence, amplitude modulation corresponds to varying the vector length, phase modulation corresponds to a variation of the vector angle. In the constellation pattern for QPSK all four points have the same amplitude and thus there is no amplitude modulation. Each of the four points has a different angle in the constellation pattern which results in a four-level phase modulation.

Higher bit rates can be achieved by further increasing the number of phase levels. In practice, the number of phase levels is limited by the noise introduced during transmission. Hence, to further increase the bit rate, it is common to employ both phase modulation and amplitude modulation. Such a modulation scheme is referred to as **quadrature amplitude modulation (QAM)**. An example of QAM with twelve phase and three amplitude levels resulting in 16 levels per signal element (hence 4 bit per level) is shown in Fig. 2.2-10. This 16-point constellation is known as 16-QAM.

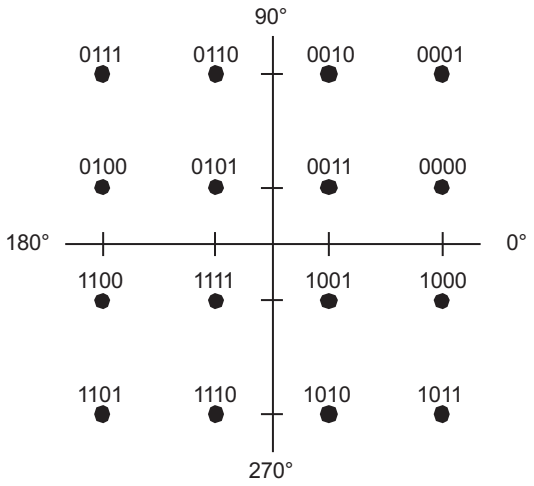


Fig. 2.2-10: Constellation pattern of 16-QAM Modulation

(Exercise 2.6.3)

Exercise 2.2-3:

Draw a possible constellation pattern for

a. 8-PSK

b. 8-QAM

Multilevel ASK and PSK are often used in conjunction in high bit rate modems. In practice, the number of signal levels and thus the possible bitrate is restricted by the channel's impairments like limited bandwidth, and noise level. In fact, the highest possible bitrate for a channel with additive noise and limited bandwidth was determined by **Claude E. Shannon** in 1948:

$$C = B \log_2 \left(1 + \frac{S}{N}\right) \qquad 2.2\text{-}3$$

where S is the average signal power, N is the random noise power,and B is the bandwidth of the Channel. C is called the **channel capacity** and is measured in bit/s. The channel capacity can be interpreted as follows: if the bit rate R is less than $C(R < C)$, then it is theoretically possible to achieve reliable (error-free) transmission through the channel. On the other hand, if $R > C$, reliable transmission is not possible regardless of the amount of signal processing performed at the transmitter and receiver [Pro94].

Shannon's theorem gave birth to a new field in communications engineering which is now called **information theory**.

(Exercise 2.6.4)

Exercise 2.2-4:

What is the capacity of a channel with bandwidth B = 3100 Hz and a signal-to-noise ratio of 30dB?

Solution hints: *The following relation may be useful:*

$$\log_a x = \frac{log_b x}{log_b a}$$

2.3 Internet Infrastructure

In the past when national telecommunication monopolies still existed, services and network access were supplied by one single telecommunication company. Since the advent of liberalization in telecommunications a lot has changed and the former "One company does everything" paradigm has shifted to a general business model which differentiates between services and roles in telecommunications. For the operation of the Internet the important roles which can be identfied are the **carrier,**the **Internet service provider** (ISP) and the **content provider**. It must be emphasized that a company can impersonate all of these roles. It is possible that one company concentrates only on carrier services, whereas another company provides the complete spectrum of telecommunication services. A content provider is normally only interested in the effective dissemination of his products and will cooperate with an ISP who will be in charge of the technical aspects of Internet publishing. The distinction of roles is necessary to define interfaces for services. This is a vital facet of liberalization on the telecommunication market, because it allows competition to emerge in different areas of telecommunication services.

2.3.1 Carrier

A carrier provides the transport network over which voice and data communication is carried out. Besides former monopolists other companies like e.g. public power suppliers have built private national telecommunication networks. In Germany, for instance, companies like RWE, VIAG, Thyssen and Deutsche Bahn AG have built their own telecommunication networks and are now able to compete with German Telekom for customers. In Section 1.7 it was shown that a general national communication network consists of three layers - carriers can be present in all three of them. Especially in the long-haul and regional domains, the competition among the carriers is quite strong. In Germany it could be observed, that charges for national phone calls significantly dropped since the fall of the telecommunication monopoly in 1998, which is a direct consequence of increased competition. In the access network, though, the situation is different. Here, the infrastructure is still owned by the ex-monopolist German Telecom (which means that local phone calls are still expensive). Other companies are eagerly trying to bridge this last mile with their own

equipment. The different technologies which are employed in the access network as an alternative to leasing or purchasing of copper lines from the ex-monopolist are described in Section 2.4.

Carriers can be categorized according to the geographical area they operate in. Typical categories are:

- international carrier (supercarrier),
- national carrier (e.g. former monopolists),
- regional carrier, and
- city carrier.

In colloquial language a permant connection provided by a carrier is often called **leased line**. From a technical point of view the services a carrier provides are either leasing of physical lines or leasing of transmission capacity on physical lines.

2.3.1.1 Dark fiber

The dark fiber is a special form of a leased physical line. The carrier provides an optical fiber link without any further services or equipment. The customer has to connect the fiber to his own transmission systems. The customer has absolute freedom as to which transmission technology will be used. Hence, also operation and maintenance is completely controlled by the customer.

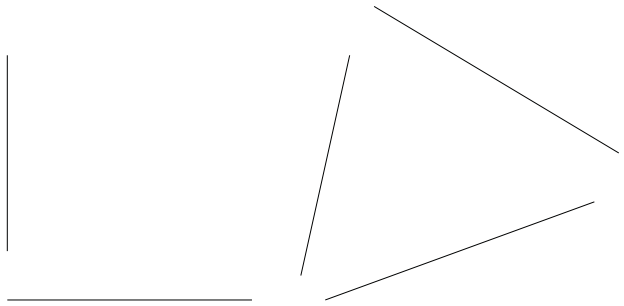


Fig. 2.3-1: Optical fibers without transmission equipment

Although other transmission technologies, like radio transmission and transmission over copper lines, do exist, optical fiber transmission has an outstanding importance in current and future telecommunication transport networks. In parts of the access network where copper lines still outweigh optical fibers and the capacity demands of the customer are moderate, copper lines are still used to achieve connectivity for customers.

2.3.1.2 Transmission Capacity

In cases where customers do not want to setup their own transmission equipment, it is provided by carriers and customers can rent a certain amount of transmission capacity (normally in terms of Mbit/s) from the carriers. For the customer this scenario has the advantage that all technical systems, maintenance and operation are provided by the carrier. Hence, a customer can concentrate on providing his services without having to deal with the low-level technical issues of signal and data transmission.

The transmission capacity provided by the carrier does not need to be static. In fact, real life telecommunication applications tend to be quite dynamic with regards to traffic demands. For instance, many enterprises need high transmission capacities during the office hours when their employees are working at their computer terminals and produce high volumes of data traffic. During the night, the traffic requirements are often considerably lower. Since enterprises only want to spend money on transmission capacity they really use, they will sign a contract with a carrier which exactly regulates the provisioning of capacity and its distribution over the day. Typical data rates for leased lines in Europe and the USA are displayed in Table 2.3-1 and Table 2.3-2.

Tab. 2.3-1: *Standard data rates in the USA*

USA	T0/DS0	T1/DS1	T2/DS2	T3/DS3	T4/DS4
Channels	1 ISDN	24*T0	4*T1	7*T2	3*T3
Data Rate	64 kbit/s	1.544 Mbit/s	6.312 Mbit/s	44.736 Mbit/s	139.264 Mbit/s

Tab. 2.3-2: *Standard data rates in Europe*

Europe	E0	E1	E2	E3	E4
Channels	1 ISDN	32*E0	4*E1	4*E2	4*E3
Data Rate	64 kbit/s	2.048 Mbit/s	8.448 Mbit/s	34.368 Mbit/s	139.264 Mbit/s

Fig. 2.3-2 shows an example network from the viewpoint of a customer and the carrier. In Fig. 2.3-2a the traffic demands of a customer who wants to set up a corporate network are displayed. The physical transmission network is provided by the carrier and can be seen in Fig. 2.3-2b. The capacity configuration of the different links is left to the carrier and is completely transparent to the customer (the customer only sees the configuration as displayed in Fig. 2.3-2a). Furthermore, the carrier can use the same physical links to accomodate traffic from other customers. The connections between the different fiber links are established by cross connects (XC). These cross connects are switches which are centrally managed by the carrier and are configured to dynamically route traffic from one link to another link.

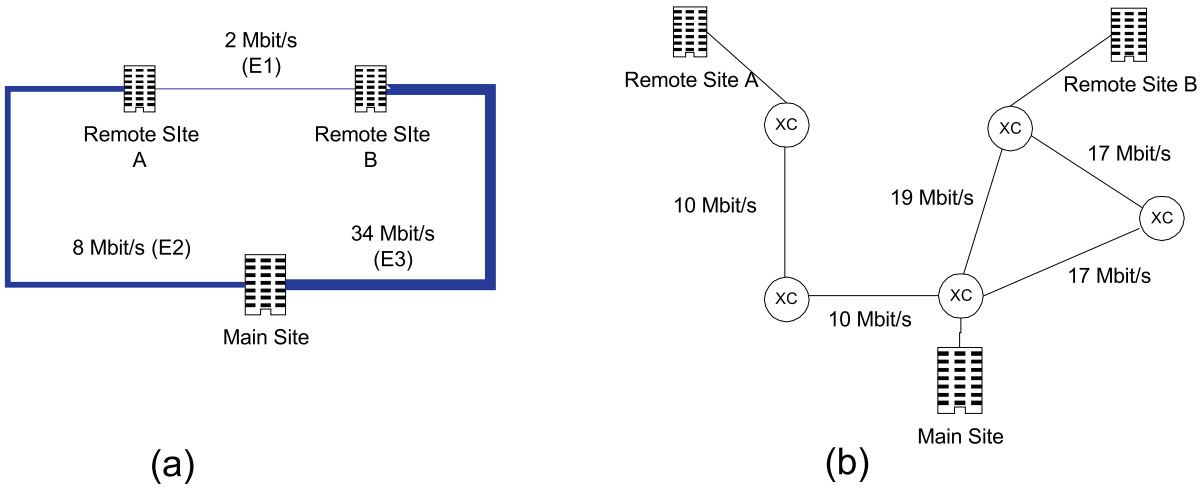


Fig. 2.3-2: a) Logical corporate network consisting of a main site and two remote sites. b) Corresponding physical carrier network and resulting traffic flow on physical links.

A carrier is able to offer transmission capacity to multiple customers over the same physical links. The interest of the carrier is to make maximum use of the available physical recources while it must be ensured, that the capacity demands of the customers are met. Since the behaviour of data traffic is hard to predict due to its bursty nature, many carriers today only use half of the available capacity to leave a headroom for peaks in the data stream.

Large Internet service providers need a long-haul network to interconnect their different routers. Many ISPs sign a contract with a carrier who is able to provide the network infrastructure they need. In this case, the network which interconnects the ISP's routers is called the **backbone.**The backbone of a typical national ISP may look similar to that displayed in Fig. 2.3-3. A backbone can be set up with both of the above mentioned carrier services.

In Section 1.7 it was shown that long-haul networks typically posess a mesh topology. Real world backbones like that displayed in Fig. 2.3-3 often differ from the ideal structure, in that they are not fully meshed. The mandatory link redundancy is still achieved, though. Currently (2000), typical bit rates on the optical links are 2.5 - 10 Gbit/s. Due to the constant increase in Internet and voice traffic the capacity will soon reach 1 TBit/s.

Internet backbones can have different topologies and utilize diverse transmission technologies depending on the ISP's size and area of operation. For instance, regional backbones often posess a ring topology instead of a mesh topology. Hence the term backbone does not relate to size and topology of a network but to its functional property, which is the interconnection of an ISP's routers with high capacity physical links.

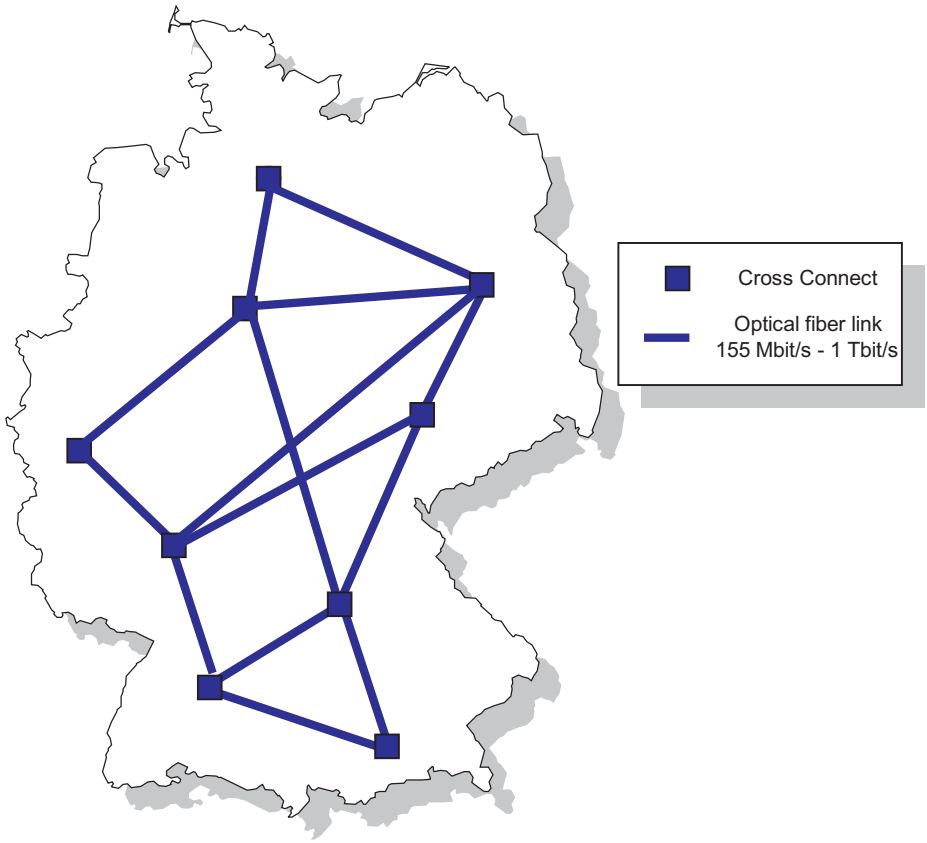


Fig. 2.3-3: Long-haul network of an imaginary national carrier. This network can serve as a backbone for national Internet service providers. ISPs can either lease capacity or the physical line (dark fiber).

Exercise 2.3-1:

Visit the following web site:

http://www.cybergeography.org/atlas/isp_maps.html

2.3.2 Internet Service Provider

The Internet Service Provider (ISP) is the instance which offers Internet services to enterprises, small offices and home offices (SOHO), and private customers. Typical services offered by ISP's are

- operation of a regional, national or global Internet-backbone (IP-backbone), which is basically a router backbone on top of a carrier backbone,
- Internet access via leased lines or dial-up connections,
- provisioning and administration of hardware for the customer (e.g. web servers), also known as **housing**,
- **content-hosting** (e.g. web sites),
- operation of servers for Internet services like e-mail and file transfer,

- special services like inter-LAN connections (VPN - virtual private networks).

An ISP normally leases physical lines or transmission capacity from a carrier and provides its own hardware to create **points-of-presence** (POPs) for the Internet. A point-of-presence is an access point for the Internet which supplies a confined geographical region with Internet services. The number of POPs an ISP operates depend on its size. A regional ISP will normally only run a few POPs, whereas a national ISP will have POPs distributed all across the country.

To allow the exchange of data between different ISP's a common exchange point has to be created. The agreement of different providers to exchange data between their networks is called **peering** agreement. In fact, the Internet heavily depends on such peering agreements because they are essential for the efficient transmission of data through the Internet. In many countries central **Internet exchange points** (IXP) exist, where all major ISPs are present (Fig. 2.3-4). Without such central exchange points it could happen that traffic between two national ISPs would take a very long way through the Internet, because no bilateral peering agreement between these two ISPs exists. For example, in Germany, before the installation of the exchange point in Frankfurt (DE-CIX) it was common, that Internet traffic between two German ISPs was first routed across the ocean to the USA and then re-routed back to Germany, simply because there existed no direct connection between these two ISPs apart from this trans-atlantic link. Hence, the existance of exchange points leads to short transmission paths between different ISPs. Internet exchange points are normally operated on a non-profit basis by an organisation consisting of several major national ISPs. Besides their importance for national connectivity, IXPs also provide the exchange point for international data traffic.

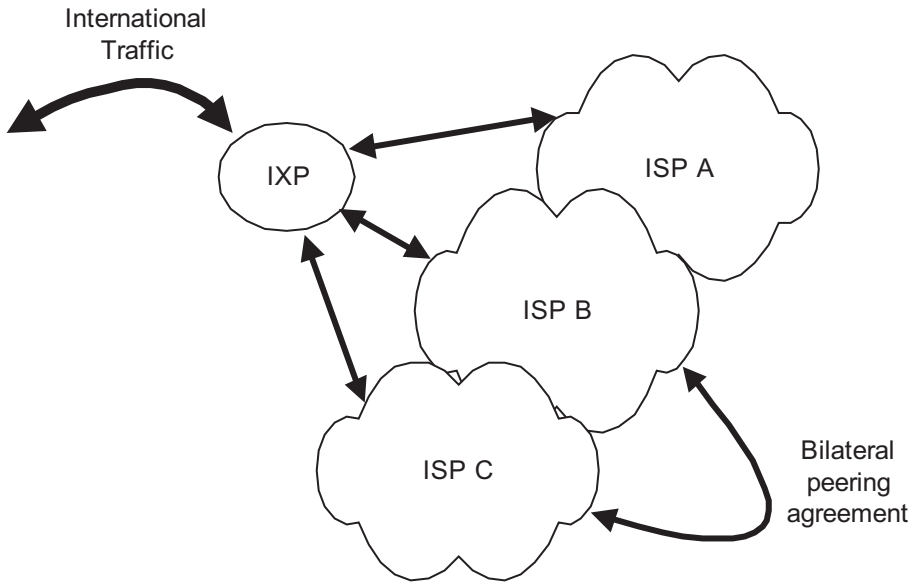


Fig. 2.3-4: Interconnection of different providers through an Internet Exchange Point (IXP).

The concentration of traffic at exchange points also has some disadvantages:

- since IXPs can only handle a constrained volume of traffic they often represent a bottleneck for national and international traffic flow,
- due to their strategic value for the operation of the Internet, they are potential targets for terroristic and criminal acts. This threat becomes more and more significant with the increasing economical and political importance of the Internet. Although the Internet is often described as a higly distributed and fault-tolerant network, the reality is that a concerted attack on a small number of nodes can render it into a conglomeration of unconnected network islands.

The core business of most ISPs is to provide business and private customers with Internet access. The connection from the customer to the ISP's point-of-presence is established either by dial-up (private customers) or by leased-line connections (enterprises). If the ISP is also a carrier, it is able to offer integrated (and thus cheaper) services to its customers.

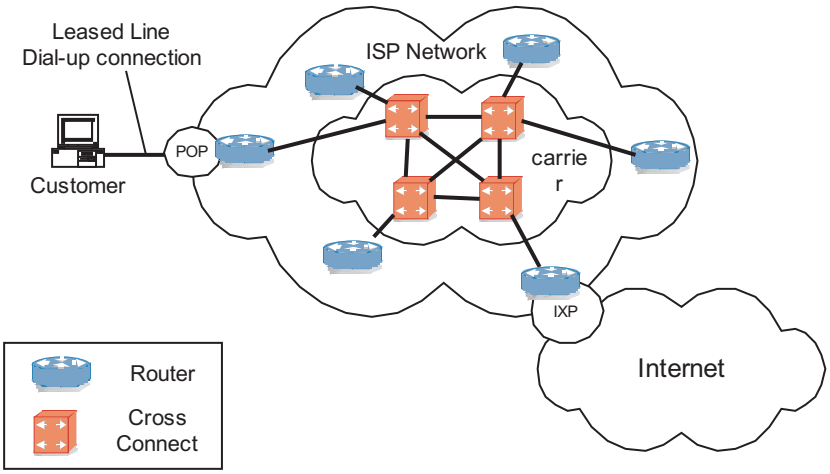


Fig. 2.3-5: A customer connects to the Internet via an ISP

Fig. 2.3-5 shows the relationships between customer, ISP and carrier and thus presents how a computer finally connects to the Internet.

- The customer is connected to the POP of an ISP either by a dial-up connection (normally private customers and small offices) or a leased line (normally business customers).
- The ISP transports the customer's data through his network, which is a router network (logical network). The routers are connected by physical lines which are leased from a carrier (the ISP can sign contracts with several carriers).
- At an Internet exchange point or bilateral peering point the ISP connects to the Internet and thus becomes part of the Internet.

2.4 Internet Access

In order to take part in communication over the Internet, a computer must have some kind of physical connection to an ISP's point-of-presence. Today, the global public telephone network with almost a billion subscribers forms a reliable communication network, that is able to nearly connect any point on Earth with any other point. Future information and multimedia services which shall be provided over the Internet platform however require a **broadband** connection (>2 Mbit/s) to the Internet. Since the analogue telephone channel is not able to provide a data rate of more than a few kbit/s other access technologies are currently emerging. These new access technologies can be divided into several catagories according to the utilized physical medium. The most promising of these technologies are

- DSL (digital subscriber line) which makes use of the twisted-pair copper subscriber line between the local exchange office and the customers premises,
- cable modems which are connected to the cable TV network (CATV),
- wireless local loop (WLL) radio links.

Beside the mainstream there are some niche technologies, which have not yet proved that they can be deployed both economically and technically. These are broadband **satellite access** and **powerline technology**. Although the satellite is able to cover vast areas while supporting personal mobility, it will have to compete with much more cost effective technologies on the ground. Furthermore it can only offer a low individual capacity per user within the area of coverage. The idea of transmitting data over powerline cables is intriguing, but in practice many problems arise, the most severe being the electromagnetic compatibility, since powerline cables tend to act as radio antennas (they were originally not meant to carry frequencies greater than 50 Hz).

The long term goal in the local loop is, to replace the existing copper plant with high capacity optical fibers which run right into the customer premises (Fiber to the Home - FTTH). This solution is not expected within the foreseeable future. In the meantime, the existing copper lines and alternative technologies are being utilized. It is expected that the availability of high-capacity access to every home will greatly stimulate the development of the information society, and bring areas like teleteaching and e-commerce a major step forward.

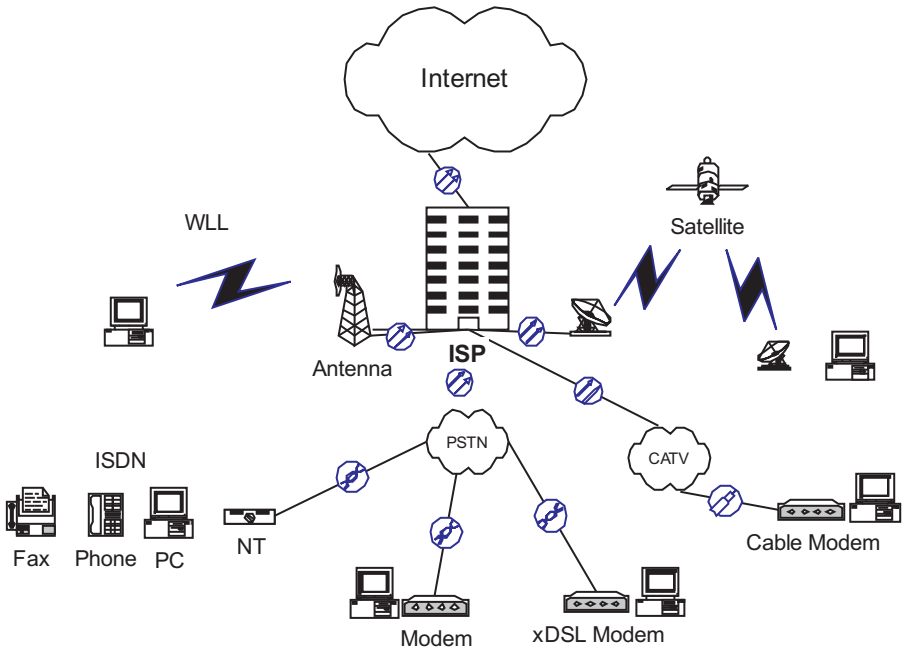


Fig. 2.4-1: Access technologies for the local loop

2.4.1 Access via Modem

The access through the **Public Switched Telephone System** (PSTN) is still for most people worldwide the only viable method to connect to the Internet. This means, that the public telephone system, which was originally planned to only carry voice traffic is used to transmit data signals. The device that accepts a serial stream of bits as input (e. g. RS-232) and produces a modulated carrier as output (e. g. QPSK) that can be transported over telephone networks is called a **modulator**. The modulated signal is transmitted through the telephone channel and is transformed back into a serial bit stream at the receiver's end. The device that accepts a modulated carrier signal and outputs a serial bit stream is called a **demodulator**. The combination of both modulator and demodulator is called a **modem**. Hence, a computer which is connected to a modem can simultaneously act as a sender and receiver. The transmission mode in which both communication partners are able to send and receive simultaneously is called **duplex** communication. The mode in which both communication partners take turns in sending and receiving is referred to as **half-duplex**. Today, most modern data communication links are operated in duplex mode.

In order to achieve an effective transmission, the data signal has to be matched to the properties of the telephone channel. Its most important properties are the limited bandwidth (300 Hz - 3.4 kHz) and the relatively high noise level.

For an average telephone channel with a signal-to-noise ratio of 35 dB, the maximum data rate can be calculated according to Shannon's law (Equation 2.3) as follows:

$$C = 3100 \times \log_2 (1 + 3162)[bit/s]$$
$$= 36044 bit/s$$

It should be noted that the signal-to-noise ratio will be different for every individual subscriber line, mainly due to varying length.

Fig. 2.4-2 shows how a computer connects to the provider's POP through the public telephone network. Inside the network, the data will probably be transported digitally[2]. Hence, the carrier signal has to run through an A/D converter when it enters the telephone network, and pass through a D/A converter when it leaves the network. The quantization noise that is created during the A/D conversion adds to the overall noise level of the telephone channel and further decreases the possible bit rate. At the provider's point of presence, a modem pool serves incoming connection requests from the provider's customers. The modem pool is connected to an **access router**, which can be seen as the gate to the Internet.

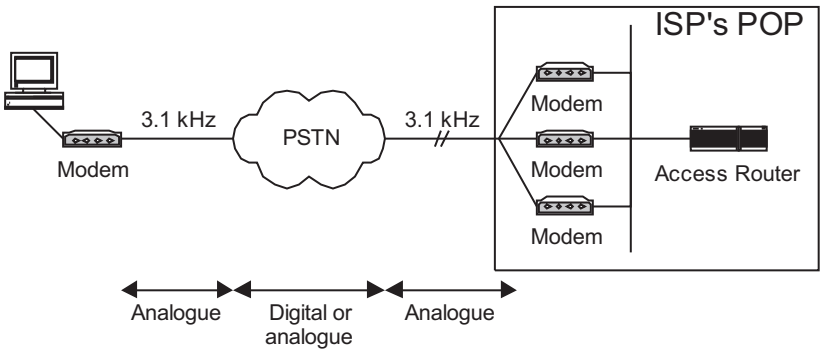


Fig. 2.4-2: Access via modem

In Table 2.4-1 the evolution of modem technology is presented. It can be seen that early modems relied on multi-level modulation techniques. Following modems employed sophisticated coding schemes like Trellis Code Modulation (TCM) which improves QAM by including **forward error control (FEC)**. As the number of amplitude and phase levels increases with each successive, more complex modulation method, both the transmitting and detecting circuitry become more complicated and costly. Finer distinctions between detected data signals in terms of phase and amplitude make it more difficult for the receiving modem to decide whether the signal is a "0" or a "1". As discussed previously, Shannon's theorem gives the upper bound of the transmission speed as function of the signal-to-noise ratio for a given transmission line.

2 Today, most modern telephone networks are digital

Tab. 2.4-1: *Modem Milestones*

Date	Standard	Rate (bit/s)	Modulation
1968	V.26	2400	4-PSK
1972	V.27	4800	8-PSK
1976	V.29	9600	16-QAM
1984	V.32	9600	2D TCM
1994	V.34	28800	4D TCM
1998	V.34 (revised)	33600	4D TCM
1998	V.90	56000/33600	PCM/4D TCM

It is interesting to see, that it took the communications engineers approximately 30 years to even come close to the theoretical Shannon bound. V.34 Modems mark the end of the traditional modem development process, being already close to the theoretical limit. V.34 modems are able to operate at a maximum bit rate of 33600 bit/s. This bit rate is often not achieved on low quality telephone lines, and thus the modem is able to fall back to lower bit rates (e. g. 28800 bit/s).

The V.90 modems which are sold today, are operated at two different bit rates for the upstream and downstream directions. For the upstream direction, the modem exactly behaves like a V.34 modem. On the downstream from the provider to the customer V.90 modems can achieve up to 56000 bit/s. This is possible, because the provider is digitally connected to the PSTN. Fig. 2.4-3 shows the difference between V.34 and V.90. An important trait of V.90 connections is the missing of an A/D conversion in downstream direction. Since every A/D conversion (ADC) introduces quantization noise (see Fig. 1.3-6), the removal of A/D converters leads to an increase of the signal-to-noise ratio. Hence, a V.90 modem can assume a better signal-to-noise ratio than a V.34 modem. Nevertheless, in practice, this assumption often proves to be unrealistic, which is confirmed by the fact that only a fraction of these modems actually reach the maximum bit rate of 56 kbit/s on the downstream from the ISP to the customer. If the telephone line from the PSTN to the customer is too poor (low S/N), the modem operates in V.34 mode for both directions.

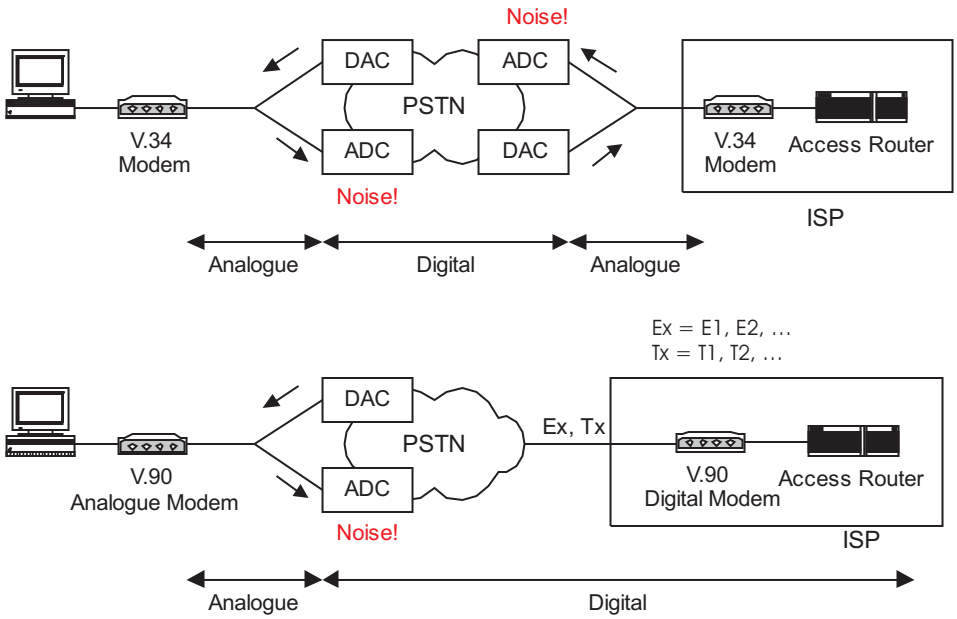


Fig. 2.4-3: Comparison of V.34 and V.90 technology. V.90 only works when ISP is digitally connected to telephone network

2.4.2 Access via ISDN

The fact that a telephone network operator is not able to guarantee the availibilty of a fixed bit rate due to the varying quality of the analogue telephone channel, raises the demand for a more reliable data connection than the 3.1 kHz plain old telphone service (POTS) can provide.

One of the first efforts to provide large-scale digital services to subscribers was launched by telephone companies under the name **Integrated Services Digital Network (ISDN)**. ISDN provides digitized voice and data services to subscribers over conventional local loop wiring. Supplementary ISDN services for voice communication are caller identification, call forwarding, call waiting, and conference calling. Further applications of the ISDN services include computer communication, high-speed facsimiles, and low bit rate videophones.

The ISDN standards (ITU) specify the **basic access** and the **primary access** for users.

For basic access, the telephone network operator guarantees that the customer will have access to two channels, each with a bit rate of 64 kbit/s (duplex), which can both be used for data and voice transmission. Such a channel is also called **B channel**. Furthermore, the two B channels can be bundeled, thus resulting in a bit rate of 128 kbit/s. The **D channel** is used for the exchange of signalling information at a bit rate of 16 kbit/s. Hence the combination 2B + D results in a bit rate of 144 kbit/s. Additionally, synchronisation information (48 kbit/s) are transmitted which leads to an aggregate bit rate of 192 kbit/s on the subscriber line.

The primary access is 30B + D (64 kbit/s) in Europe, and it is 23B + D (64 kbit/s) in the United States, Japan, and Canada. Primary access is aimed at commercial users.

Since ISDN data transfer is carried out over the same telephone line as POTS the following question arises: How can ISDN go so much faster on the same phone lines that supposedly were already close to operating at the theoretical limits with 33.6 kbit/s and 56 kbit/s voiceband modems?

The answer is: The ISDN channel is not limited to the 3.1 KHz bandwidth of the analogue telephone channel. ISDN employs baseband transmission instead of modulation and makes use of a greater part of the line's available bandwidth (POTS only uses a fraction of the available bandwidth). The utilized bandwidth of an ISDN channel is shown in Fig. 2.4-5. Since ISDN does not employ modulation a modem is not needed and usually an **ISDN card** is installed inside the computer which interfaces to the ISDN S-bus (see Fig. 2.4-4). The network termination (NT) interfaces to the twisted pair subscriber line and acts as transmitter and receiver for incoming and outgoing data.

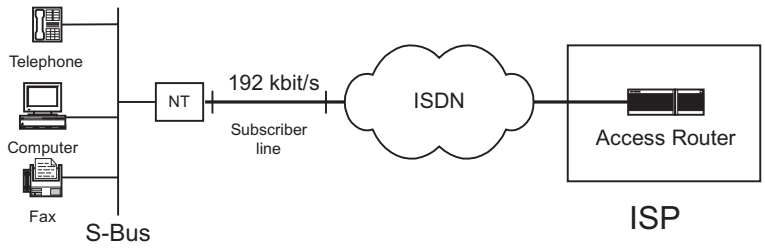


Fig. 2.4-4: Access via ISDN

A subscriber line that is utilized to transmit digital information without the constraints of the analogue telephone channel is called **digital subscriber line (DSL)**. ISDN was the first worldwide effort to establish DSL applications in the local loop of public telecommunication networks.

Besides the higher bit rates, ISDN has further advantages for Internet applications:

- The connection setup time is much lower than for modem connections. Normally an ISDN connection is established in less than a second.
- When a single B-channel is used for Internet access, the other B-channel is still free for telephone calls.
- The delay which is introduced in an ISDN card is considerably lower than the delay of a modem (introduced by the complex modem circuitry). This is important for interactive applications which need fast response times (ideally less then 100 ms).

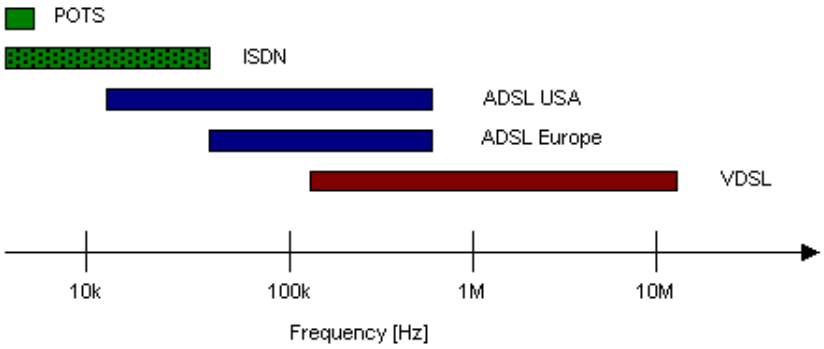


Fig. 2.4-5: Bandwidths of several DSL technologies in comparison to POTS. (logarithmic scaling!)

Exercise 2.4-1:

Examine the efficiency of ISDN basic access in comparison to current V.34 modem data transmission. Assume a bandwidth of 80 kHz for the ISDN basic access. Express the efficiency in terms of bit/s/Hz.

2.4.3 Access via xDSL

The recent developments in DSL technologies are significantly exceeding the ISDN data rates. The family of different technologies that are employed on the copper subscriber line with bit rates in excess of the ISDN rates are called xDSL. xDSL make use of much higher frequencies than ISDN does. Two xDSL technologies, ADSL and VDSL, are shown in comparison with ISDN and POTS in Fig. 2.4-5. Fig. 2.4-6 shows how a computer can use xDSL to connect to a service provider. The xDSL technologies must asure that telephony services are still available in parallel to the new data services. This is accomplished by utilizing splitters at the customer's premises and the exchange office. At the office the data signal is processed by a **digtal subscriber line access multiplexer (DSLAM)** which concentrates the traffic from all connected subscriber lines. The data traffic is transported over broadband leased line connections to the ISP's point of presence.

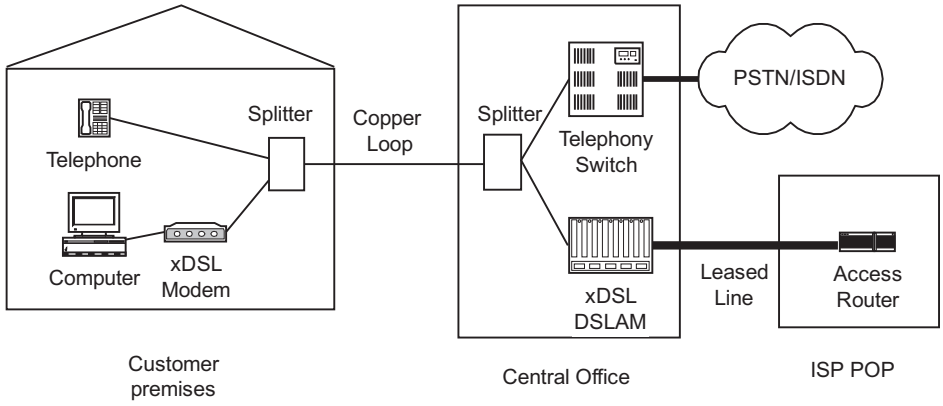


Fig. 2.4-6: Access via xDSL

At the moment, the most prominent member of the xDSL family is **ADSL**, which stands for **asymmetric digital subscriber line**. ADSL is offered in different flavours depending on the length and quality of the subscriber line. Typical data rates are 1.5 - 6.144 Mbit/s downstream, and 128 kbit/s-512 kbit/s upstream. The asymmetric nature of the upstream and downstream bitrates leads to ADSL being ideally suited for Internet applications with asymmetric traffic behaviour. Typical examples are:

- Access to WWW pages
- Distance learning and training
- Telemedicine
- Multimedia content streaming
- Software downloads

All these applications have in common that they require a much higher data rate on the downstream from the Internet to the customer than in the upstream direction from the customer to the Internet. Hence, ADSL is well suited for private customers who make use of such asynchronous Internet applications.

For the European version of ADSL it is obligatory that the spectrum of the ADSL data signal does not interfere with the widespread ISDN (In the USA ISDN is far less popular.). For this reason, the spectrum of European ADSL was shifted to be placed above the ISDN spectrum (80 kHz for most European countries).

The two competing transmission techniques for ADSL are **Carrierless Amplitude- and Phasemodulation (CAP)** and **Discrete Multi Tone Modulation (DMT)**. CAP is a special form of 64-QAM and avoids transmission of the carrier frequency. DMT divides the available frequency spectrum into 255 subchannels (225 data channels + 30 control channels) and uses 64-QAM for each of the subchannels. The transmission equipment frequently adjusts the bitrate for each of the subchannels during transmission in order to maximize throughput of the ADSL connection.

The next generation xDSL technology **VDSL (Very High-Speed Digital Subscriber Line)** does not rely on twisted pair transmission alone, but assumes a combination of optical fiber and short copper twisted pair lines. Such a hybrid concept can be realized by installing an optical fiber close to the costumer premises in the local loop. The bit rate that can be delivered to the end user depends on the length of the copper loop. It can vary from 14.5 Mbit/s for long-range VDSL (1.5 km) to 58 Mbit/s for short-range VDSL (0.3 km). Fig. 2.4-7 shows the different concepts:

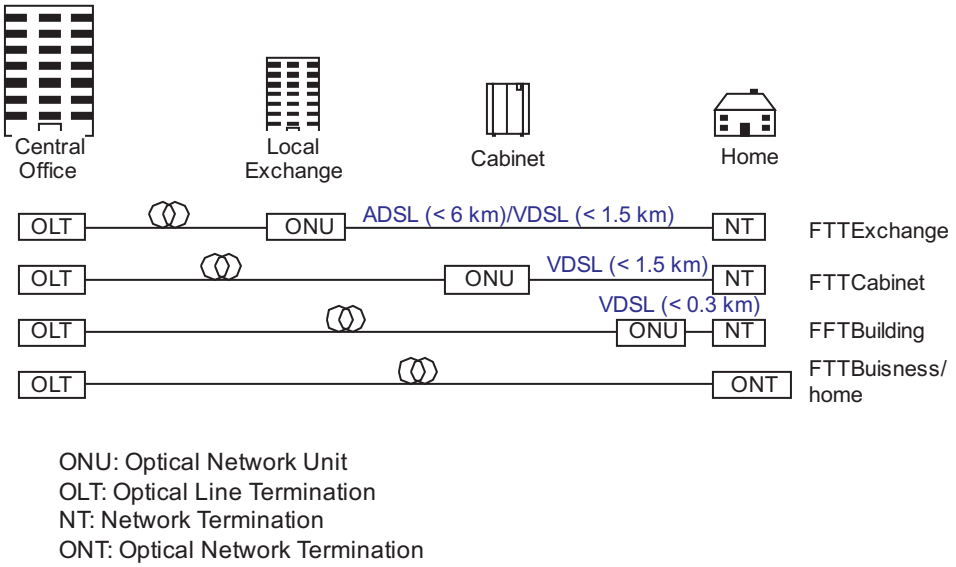


Fig. 2.4-7: Fiber to the x topologies

- Fiber To The Exchange (FTTEx): This is the topology suitable for ADSL technology. If the subscriber line is very short (< 1.5 km) VDSL can also be employed.
- Fiber To The Cabinet (FTTCab): An optical fiber runs from the local exchange to the street cabinet which is a concentration point for multiple subscriber lines. From there, the loop will continue as a regular copper twisted pair line.
- Fiber To The Building (FTTB): The optical fiber runs right into the basement of the building. The in-house cabeling consists of copper twisted pair lines.
- Fiber To The Business/Home (FTTH): All links are purely optical. Even terminal equipment like computers and telephones are optically connected.

It must be noted that although the FTTH topology offers the highest capacity (almost unlimited!) it is not likely to be realized within the foreseeable future due to the immense costs for installation of an all optical local loop. VDSL enables leveraging of the existing copper plant to offer services equivalent to those of FTTH, at much reduced costs.

2.4.4 Alternative Access Technologies

The motivation for liberalizing the telecom market is economical. New carriers and operators are supposed to compete with the former monopolist (incumbent) in order to achieve more attractive tariffs and faster development of services. The process, reffered to as **unbundling**, of allowing alternative operators to use copper twisted pairs installed and owned by the local or national monopoly telephone operator is currently underway in many parts of the world. In Europe many countries such as Austria, Germany, Denmark and Sweden have obliged the national incumbent to offer unbundling. Many countries created Telecommunication acts that basically allow anybody to operate a telecommunication network. The regulator’s role is to

grant licences that authorize new operators to offer telecommunication services. Additionally, the regulator has to decide how unbundling is to be carried out. He has to create an environment where new entrants have a fair chance to compete with the incumbent. On the other hand the regulator has to be aware not to put the incumbent's existence at risk by granting a competitive edge to the new operators. A widespread procedure is to fix the price new entrants have to pay for either renting or purchasing of existing subscriber lines from the incumbent operator.

For many new operators the purchase or rental of parts of the existing copper plant, which is still owned by the incumbent, is not acceptable. Instead they often try to built their own local loop infrastructure without relying on the existing subscriber lines. In the following a few of the alternative local loop technologies are shortly described.

2.4.4.1 Internet Access via Cable TV Network (CATV)

In many countries the national cable TV network is the second largest telecommunication network besides the public telephony network. It is thus extremely attractive for new operators since it opens the door to a vast number of private households (businesses rarely have a cable TV connection). The availability of CATV networks for Internet access heavily depend on the national situation. In Germany, the drawbacks of CATV Internet access are:

- A CATV network is a shared medium. This means that many households share the same physical line. Consequently, the available bit rate must also be shared and can become very low in some urban areas where many households share a single cable.
- Originally, CATV networks were not meant to transport signals in both directions. The amplifiers inside the CATV network will not let pass signals from the residentials to the broadcast head end. In order to allow bidirectional signal transport many CATV networks have to be refitted with new amplifiers. The costs for this upgrade process are tremendous.

The connection to the Internet is established through a cable modem which typically use variants of QAM to transmit data. Maximum data rates are 47 Mbit/s downstream and 10 Mbit/s upstream. It must be noticed that these are aggregate bit rates which have to be shared by several households. For example, when a single physical cable is shared by 100 households, each will have a share of 470 kbit/s in the downstream.

2.4.4.2 Wireless Local Loop (WLL)

Wireless Local Loop technologies, which are also often referred to as Radio in the Local Loop (RITL, RLL), represent a new radio-based access technology with cellular architecture offering flexible high-capacity connections to businesses and private users. WLL comprises a multitude of different technologies such as **DECT** (Digital European Cordless Telecommunications) and **LMDS** (Local Multipoint

Distribution Systems). WLL is a very attractive local loop solution for new operators, who do not own infrastructure in the access network and quickly want to set up their own network. Furthermore, WLL is well suited for rural areas in which the installation of copper or fiber technology is very expensive. For the provisioning of WLL services, special frequency bands are specified and auctioned by the regulator.

For example, in Germany DECT systems may use frequencies between 1880 and 1900 MHz. The maximum bit rate in DECT systems which must be divided between the upstream and downstream channel is 1152 kbit/s. Eventually, this bit rate also has to be shared among multiple users. LMDS systems may use the bands 2.54-2.67 GHz, 3.4-3.6 GHz, and 24.5-26.5 GHz. The available bit rates are significantly higher than in DECT systems and can be up to 38 Mbit/s in the downstream.

2.5 Summary

Baseband transmission is the easiest way to transfer data signals over physical media. Here, the binary digits of the data signal are directly encoded as voltages on a physical line. This method falls short when data signals have to be transported over long distances or radio links. In this case, the transmission signal has to be matched to the properties of the channel. This is achieved by modulation which is also called bandpass transmission. If data signals are transported on top of a carrier signal the transmission is referred to as digital modulation. The achievable bit rate on a transmission medium is vastly influenced by the medium's bandwidth and the signal-to-noise ratio. The relationship between these impairments and the theoretical bit rate is given by Shannon's Law which introduces the notion of channel capacity.

The Internet is a complex structure consisting of a multitude of physical networks which are interconnected by routers. Large physical networks are operated by carriers who offer leased line services to their customers. The Internet service provider operates a router network and offers Internet services to business and private customers. The routers are connected by leased lines which are rented from a carrier. The leased line is also the favourite Internet access type for businesses. The traditional way for private users to connect to the Internet is through the public telephony system. Modems can only offer a restricted bit rate due to the bandwidth restrictions of the telephony channel. New technologies for the wire based local loop are xDSL technologies which can offer considerably higher bit rates over the same copper twisted pair that POTS employ. In the course of deregulation alternative access technologies are well suited for new operators to install their own infrastructure in the local loop.

3 Protocols at the Network Layer

3.1 Goal of the Chapter

In this chapter, we first explain the basics of naming and addressing in the Internet. The structure of IP addresses and the different types of addresses which are used in the Internet are outlined. After an explanation of the concept of naming in the Internet, the functionality of the Domain Names Service (DNS) translating IP names into IP addresses is described.

In the second main part of this chapter, the protocols operating at the network layer such as IP and ICMP are explained. In the context of the IP header and the operation of the IP protocol we also briefly outline some routing issues in the Internet.

3.2 Introduction

There are many protocols operating at the TCP/IP protocol stack. As we have seen in Section 1.8.4.2, the TCP/IP protocol stack does not specify the physical and link layer protocols. It rather specifies how these layers will be interfaced by the Internet. Many physical networks can interface with the Internet such as the PSTN as we have seen in Section 2.4.

At the network access layer, no universal TCP/IP protocol is available, since this layer exclusively handles local network transmission. ARP and RARP (Address Resolution Protocol and Reverse Address Resolution Protocol) operate at the network access layer where they are used on any network that needs to convert internetwork addresses to physical addresses which may be hard-coded in the network interface hardware. Although ARP and RARP are concerned mostly with issues at the network access layer, they can also be depicted as being part of the network layer operating in parallel to the Internet Protocol. In the following, we will consider the ARP and RARP protocols in the context of the network layer.

The dominant protocol at the network layer is the Internet protocol (IP) which is used for internetwork routing of data between hosts and across network links. ICMP (Internet Control Message Protocol) is a kind of helper protocol that works with IP to pass error and status information between hosts and routers (see Section 3.4 for more information about IP). ICMP messages are carried within IP traffic, so ICMP is sometimes depicted as operating on the layer above IP. But its function is strictly related to the network layer.

Internet routing, the set of methods for making sure that network traffic is passed between hosts and networks efficiently, needs its own set of protocols. These routing protocols ensure that routers are able to detect changes in internetwork routes and to keep data flowing across the internetwork. For more information about routing in the Internet see Section 3.4.1.1.

As many operations at the network layer are concerned with the concept of naming and addressing in the Internet we will first explain this topic before we proceed with the network layer protocols.

3.3 Networks Addresses and Names

Two hosts connecting across an internetwork have much in common with two people participating in a conversation via a telephone connection. A circuit between the two participants who are communicating is established. None of the two persons has to be concerned with what type of equipment the other is using, nor does either of them have to be aware of the route the telephone company uses to establish the circuit. It is possible that different technologies are used by the people involved. But all these details are irrelevant to the people on the phone.

When setting up a circuit, a unique identifier for the destination telephone is required for the party intiating the circuit. When dialed, any telephone number in addition to its area and country codes will connect the dialer to a particular telephone.

The same is true on an internetwork where each node has a unique identifier used to establish connections with that node. If another host connected to the internetwork invokes that network address when starting a network application program the data will be transmitted to this host if it is accessible over the internetwork.

TCP/IP networks have some similarities to telephone networks, but they are different in one important way. Traditional telephone networks apply circuit switching (see Section 1.4, Fig. 1.4-1), which means that an electrical circuit is build up among the participants of a telephone call. TCP/IP networks employ packet-switching in datagram mode. This means that data is divided into packets that are handled individually. Each packet is sent along the best route that is currently available. So, if a link breaks down, the router can send the packets along another route.

In this way, the network delivers packages of data from a sender to a receiving host. The destination address is used to target a uniquely identified host, and the return address is used by the destination host to respond.

3.3.1 Naming and Addressing in the Internet

In the Internet,**IP addresses** are used to identify hosts and route data to them. Hence, every node in a TCP/IP network must have a valid IP address and should have a valid name. Name and address must be unique in the network. A host name is translated into its IP address by looking up the name in a database of name-address pairs. Every link in a network must have an address.

IP network addresses and names actually identify the link of connected nodes which means the network interface of connected nodes. Consequently, a node with more than one network interface could have as many addresses and names as network interfaces.

3.3.2 The Internet Protocol Address Space

The network address space is defined by the length of the network address and by rules for allocating addresses within the space. IP addresses comprise a length of 32 bits (4 bytes), usually represented in dotted decimal format. This means that the addresses are expressed with decimal digits, each byte seperated by a dot.

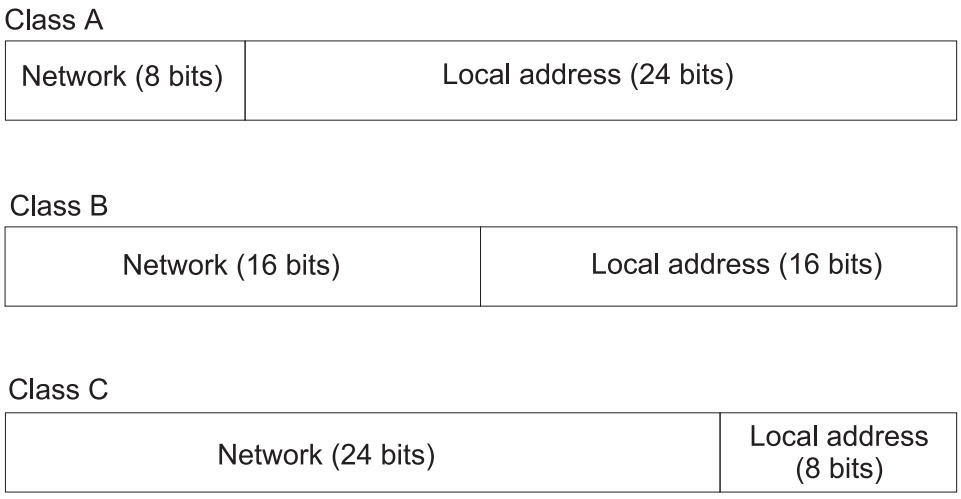
Example 3.3-1:
An IP address may look like this: `a.b.c.d`, where each letter represents one byte. For example, the central WWW server of the University of Hagen which provides the homepage of the university has the IP address `132.176.114.35.`

IP addresses are assigned through national registries that distribute network addresses. Network addresses are distributed to organizations, e. g. companies, internet service providers, universities, which are resonsible for correctly numbering all the attached hosts.

The network address is an IP address in which the least significant bits are set to zero. The number of bits of the IP address which are set to zero depends on the address type. The most significant bits on the left identify the network, the least significant bits identify the individual hosts in the network.

Example 3.3-2:
The network address of the university of Hagen is `132.176.0.0`. So, the 16 least significant bits on the right of the address are set to zero. The hosts belonging to this network may have addresses such as `132.176.1.1`, `132.176.1.2`, ...

Five types of IP address classes have been defined, named Class A, B, C, D and E. The formats of class A, B, and C which are the **traditional address classes** are displayed in Animation 3.3-1.



Animation 3.3-1: Traditional address classes

When developing these address classes the idea was that only few, very large organizations would need more than 65.000 hosts as allowed by class A. It was assumed that more organizations would need smaller networks with up to 65.536 hosts in class B or up to 256 hosts in class C.

In addition to Classes A, B, and C two special address formats have been defined: Class D and Class E. Class D addresses are used for IP multicast. Multicasting distributes a single message to a group of computers distributed over a network. Multicasting is for example applied in multipoint videoconferences in the Multicast Backbone (MBone). Class E addresses are reserved for experimental use.

Finding out to which class the address of a network belongs can be easily realized by looking at the first byte of the address.Table 3.3-1 shows the permissible IP address ranges for each network class.

Tab. 3.3-1: *IP network address classes*

Address class	Address range	Number of networks per class	Number of local addresses per class
Class A	0.0.0.0 - 127.255.255.255	126	16.777.216
Class B	128.0.0.0 - 191.255.255.255	16.384	65.536
Class C	192.0.0.0 - 223.255.255.255	2.097.152	256
Class D	224.0.0.0 - 239.255.255.255	Multicast address space	-
Class E	240.0.0.0 - 247.255.255.255	Experimental use	-

Example 3.3-3:
The central WWW server of the University of Hagen has the IP address `132.176.114.35`. Looking at Table 3.3-1 we can readily find out that the address belongs to Class B as the first byte has the number 132 lying in the range between 128 and 191.

3.3.3 Network Names

The numerical IP addresses are well suited for processing in routers and hosts, but they are difficult to remember for humans. For the end user it is much easier to work with names than with numbers identifying networks and hosts.

Consequently, IP networks are named. The resulting system of names is administered in a distributed database called Domain Name System (DNS) and used by the Internet software to resolve network and host names to find out their IP addresses. The DNS is explained in more detail in Section 3.3.5.

A domain name is a hierarchical name which is registered for an organization. Each level gives more information about the concerned domain. Originally, seven**root domains** indicated with three characters have been defined (see Fig. 3.3-2). These domains are usually used for networks in the United States.

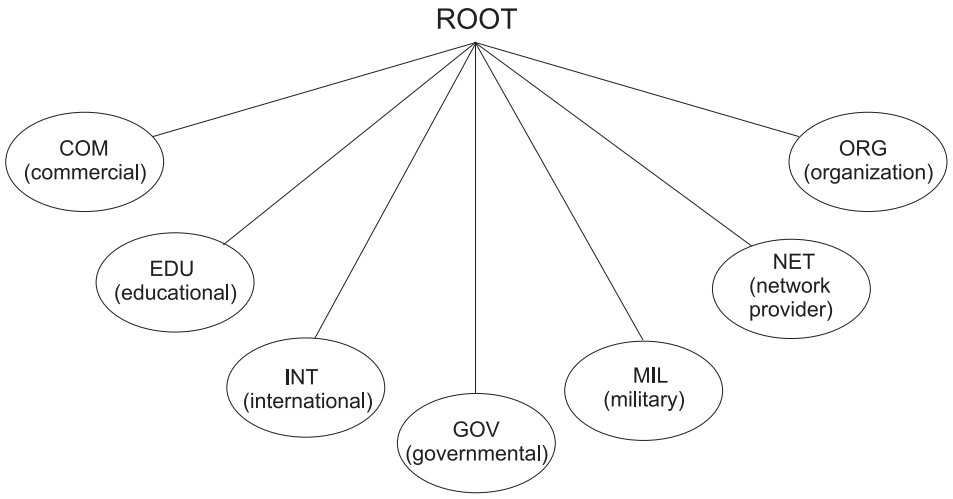


Fig. 3.3-2: Root domains in the Internet

Another set of domains is taken from a list of a two-character country code which has been defined by the International Organization for Standardization (ISO), the country domains. Table 3.3-2 shows some examples of country domains:

Tab. 3.3-2: *Exemplary list of country codes*

Country	Code
Denmark	.dk
Germany	.de
France	.fr
The Netherlands	.nl
United Kingdom	.uk
United States	.us

Within each root or top-level domain, second-level domains are assigned according to further criteria. This may result in network domain names which form a name chain consisting of several links (names). At each level, a domain can be further subdivided.

Example 3.3-4:
The domain name `ub.fernuni-hagen.de` indicates the library of the university of Hagen (in German: **U**niversitäts-**B**ibliothek), which belongs to the domain of the University of Hagen `fernuni-hagen.de`.

Domain names have to be unique, but within different domains subdomain names may be duplicated.

Example 3.3-5:
The subdomain name ’ub’ may coexist in other domains like `ub.fernuni-hagen.de` and `ub.uni-siegen.de`. In both domain names, the library of each university is identified by ’ub’. This is possible as the subdomain name is used in different domains.

3.3.4 Host Names

Just as domain names have to be unique, each host also must have a unique name within its domain. Host names can be duplicated on different networks, under different network domains or subdomains.

Example 3.3-6:
The university of Hagen may identify different hosts with the name ’www’. The hosts

`www.ub.fernuni-hagen.de` and

`www.ice-bachelor.fernuni-hagen.de`

may coexist as they belong to different subdomains. In the above addresses,'ub' identifies the subdomain of the university library while 'ice-bachelor' indicates the subdomain of the bachelor of science in information and communication engineering.

A fully qualified domain name includes the complete domain name with the host name. So, the fully qualified domain name of any particular host uniquely identifies the host, first in its network and second in the whole Internet.

3.3.5 Domain Name System (DNS)

The **Domain Name System** makes use of a structure of distributed servers which administer the different network domains and host names used in the Internet. The DNS allows people to use host names instead of host IP addresses as it translates host names and domain names into IP addresses. Since the Internet consists of several millions of hosts, it is impossible to keep this information in a single DNS server. Consequently, the task of administrating domains and hosts across the entire Internet is organized in a distributed way.

Each network of an organization must have at least two DNS servers (name servers). The administrator of the network has to update the list of IP addresses and associated host names belonging to the network.

If a host needs the IP address of another host of the local network, it queries the local DNS server. When the host needs an IP address for a host outside the local network, the local DNS server contacts a root name server, depending on the top-level domain of the fully qualified domain name which is queried.

These root name servers correspond to the top-level domains such as .com, .org, .de, .uk, etc.. Each of the root name servers keeps a list of all registered second-level domains. This process continues until a domain name server has been found which can directly point to the requested host.

Example 3.3-7:
A host from somewhere in Austria attempts to connect to the host of the university library of hagen named `www.ub.fernuni-hagen.de`. The local name server to which the Austrian host is connected requests information from the root name server for the '.de' domain (see Fig. 3.3-3). The root server responds with the IP address of a name server serving the '.fernuni-hagen' domain which is maintained at the University of Hagen. The austrian name server queries that name server of the University of Hagen and receives the IP address of the name server serving the domain '.ub.fernuni-hagen.de', which is maintainted at the unversity library. Querying this server returns the IP address of the specified host 'www'. Now, the austrian host can communicate with it.

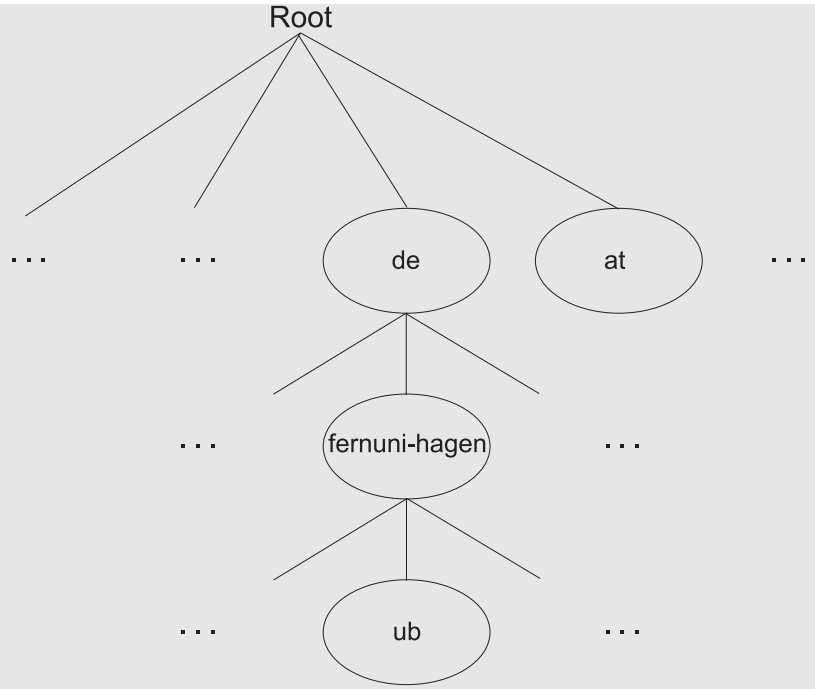


Fig. 3.3-3: Following the DNS tree when seeking a specific host

3.4 The Internet Protocol (IPv4)

The**Internet Protocol** defines rules for packaging network traffic into IP datagrams, and it defines rules for transporting these datagrams across the network.

Currently used and widespread protocol version of IP is version 4. For version 6 (IPng) called additional IP next generation is explained later.

IP uses various tools to ensure that messages can be transported to their destination: datagrams can be fragmented or broken up into pieces small enough to fit into intermediate networks. There are mechanisms for deleting datagrams if they have not been delivered to their destination after a certain amount of time has elapsed. Furthermore, there are mechanisms for securing access and authenticating IP datagrams.

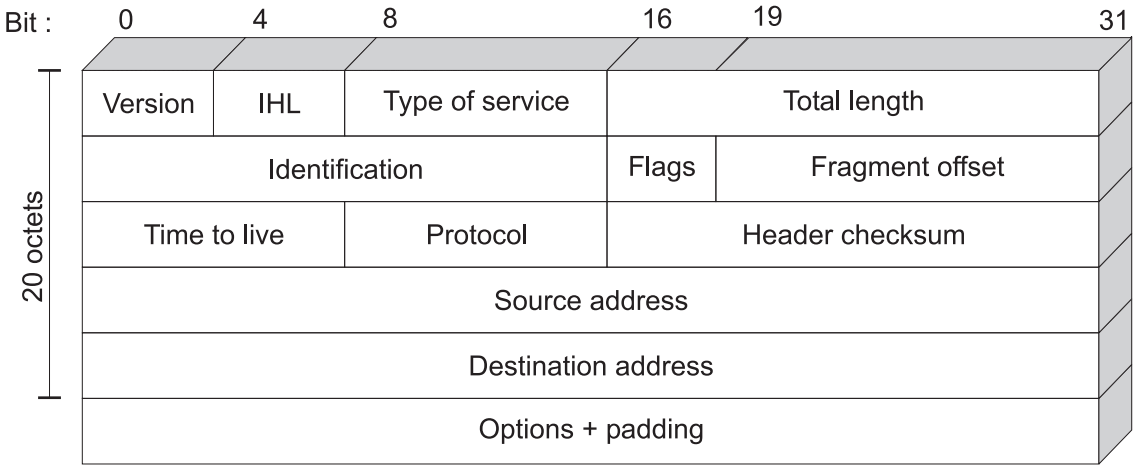
Despite all these functions, the IP service is connectionless and unreliable. Each datagram is sent across the internetwork independent of every other datagram and with neither a guarantee that it will be delivered in any particular order nor that it will be delivered at all.

Protocol designers prefer to put reliability and security functions at higher levels rather than at lower levels. Putting those services at the network access layer, for example, would require every single node to have the ability to process them. Leaving security and reliability to higher levels means that they can be implemented as end-to-end services that need to be processed only by the originating node and the destination node. Performance at the lower layers will therefore not be affected.

At the network layer, IP datagrams consist of header and data as we have already seen in Section 1.8.4.2. The header contains detailed and extensive information describing routing information for the datagram such as the source and destination address. IP headers have a length of at least 20 bytes. All the IP headers are organized into four-byte words since nodes and routers normally process four bytes simultaneously (see Animation 3.4-1).

The first word identifies the datagram: The version of the IP protocol is indicated, the header length, how the datagram is handled (type of service) and the datagram length. The next word provides information about fragmentation: a unique ID number for the datagram, flags for fragmentation control, and the fragment offset. The third word includes the time to live (TTL), which indicates how long an IP datagram may travel around the network (see Section 3.4.1), the originating transport protocol of the carried content and a checksum for the header itself. The fourth and fifth words are the source and destination IP addresses, and an optional field can be added when IP options are needed.

By looking at the meanings of the different IP datagram header fields (as shown in Animation 3.4-1), it becomes easier to understand how datagrams are created and routed through the network.



Animation 3.4-1: IP Header Fields

3.4.1 IP Datagram Fields

The following fields are integrated in an IP datagram header:

Version: This field indicates the version of the used IP protocol. The current version of IP is Version 4 (IPv4), but the latest version IPv6 has already been specified and deployed experimentally.

Header Length: This field consists of four bits and indicates the hederlength of the concerning IP datagram. The maximum value is 15 (2^4-1), thus the maximum length of an IP header is 60 bytes.

Type of Service: This field consists of eight bits, but only four bits are actually used to make Type of Service (ToS) requests of IP routers. The Type of Service option allows routers to handle IP datagrams differently. For example, the ToS bits can indicate how the datagram should be processed: to minimize delay, to maximize throughput, etc. But as some routers ignore the ToS field entirely while others use the field to make routing decisions or decide which traffic should be protected against discard, the specifications in the ToS field can only be considered as a recommendation.

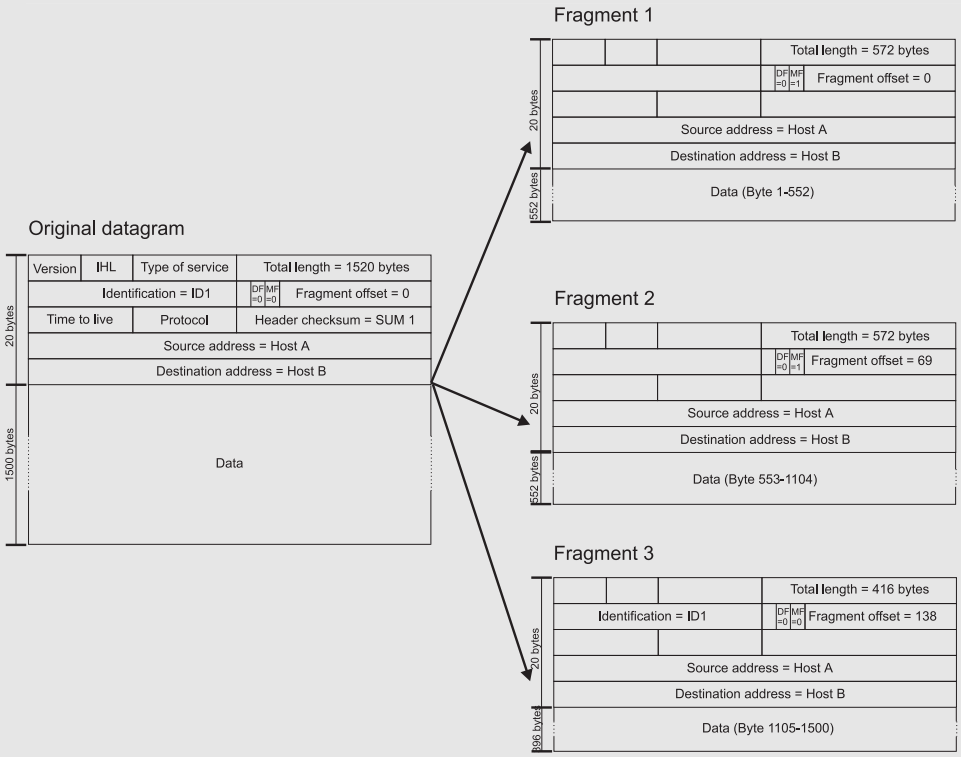
Datagram Length: The value stored in this field represents the entire datagram length, including the header. Due to the length of this field of 16 bits, the maximum size of an IP datagram is limited to a size of 2^{16}-1=65.535 Bytes.

Datagram Identification: The originating host specifies an unique 16-bit identifier for every datagram. This identification is required as the datagrams may be fragmented on their way through the internet. Fragmentation of IP datagrams can occur when they cross network boundaries between high speed networks and slower networks which may have different maximum transfer units (MTUs). When the datagrams are split, the identification field contains the same identifier for all the resulting datagrams. Given the value from this field the receiver can reassemble the fragmented datagrams back into the original datagram.

Flags: The first of these three bits is unused, the other two are used to control the way the datagram is fragmented. If the ’Don´t Fragment’ (DF) bit is set to 1, the datagram must not be fragmented. If the datagram has to be fragmented, the router throws it away and sends an error message to the sending host. When the ’More Fragments’ (MF) bit is set to 1, this means that the datagram is one of several fragments, but not the last one. If the MF bit is set to 0 this indicates that this datagram is the last of a number of fragments or that the datagram has not been fragmented.

Fragment Offset: This number informs the receiver how many units the current datagram is apart from the start of the orginal datagram. The value indicated in the fragment offset represents units of 8 bytes.

Example 3.4-1:
A datagram has a size of 1.520. It includes a header which has a size of 20 bytes. Somewhere on the path, the maximum transmission unit (MTU) is 576 bytes. Consequently, the datagram has to be divided into 3 fragments . The fragment offset value for the first fragment datagram will be 0 because the fragment begins at the start of the original datagram. The fragment offset of the second fragment will be 69 = 552 Bytes / 8 Bytes because the second fragment starts after the first 552 bytes of the original datagram. The third fragment starts after 1104 bytes in relation to the original datagram, hence the fragment offset for the third fragment is 138 = 1104 Bytes / 8 Bytes.



Animation 3.4-2: Exemplary fragmentation of an IP datagram

Time to Live (TTL): This field consists of 8 bits and indicates how long a datagram is allowed to exist after entering the internetwork. The maximum TTL is 255. Originally, the TTL was intended to measure the number of seconds a datagram was allowed to exist in transit across an internetwork. The intent was for each router to calculate how long it took to process each datagram, and then decrement the TTL by that number of seconds. In practice, datagrams traverse routers in less than a second, so a simple decrement has been implemented. As a datagram is forwarded, its TTL is decremented by one. So, practically the TTL represents the maximum number of hops that a datagram can make before being discarded (a hop means that a datagram passes a network node, e. g. a router). This restriction prevents that datagrams are forwarded again and again although they cannot be delivered to their

destination because this destination does not exist or is unreachable. Such orphaned datagrams may cause useless and undesired network traffic.

Example 3.4-2:
The TTL of an IP datagram has been set to 255. Hence, the packet may pass a maximum number of 255 routers within the Internet. Each of them decrements the TTL by one. If the packet reaches a router with a TTL=0, the router discards it.

Protocol: In this field, the protocol of the next higher layer is identified. The field may indicate for example that the payload data is a TCP segment or a UDP datagram. UDP (User Datagram Protocol) and TCP (Transport Control Protocol) are transport layer protocols which are explained in Section 4.3 and Section 4.4.

Header Checksum: This field contains a checksum of the IP header. By adding a checksum it is ensured that the datagram header is not corrupted. Nevertheless this field does not add transmission reliability or error detection to the Internet Protocol.

Source/Destination: These fields indicate the IP addresses of the sending host and the destination host. Presently, an IP address consists of 32 bits.

Options/Padding: Up to 40 extra IP header bytes are available to carry one or more options. The options that are included in a datagram are chosen by the sending applications. Examples for options are

- reverse route (the traffic flowing back from destination to the source must follow the same path), and
- record route (a header field contains a list of IP addresses of routers visited by the datagram).

Padding is used to make the header length a multiple of 4 bytes if the options do not end on a 4-byte boundary.

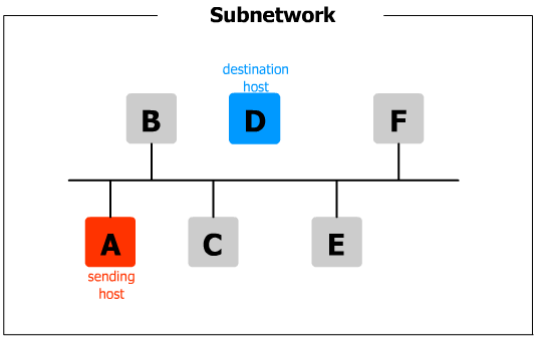
3.4.1.1 IP Routing Issues

An IP node composes a datagram and creates an IP header for it using its own address as the source address and the IP address of the destination host as the destination address. When this datagram is passed down the network protocol stack to the network access layer, the network software must decide where to send the data on the local network interface.

If sender and destination hosts are in the same network, IP routing is performed directly by the source node. The source node encapsulates the IP datagram in a network frame and directly sends it to the destination host. In this case, IP routing involves the delivery of IP datagrams between two hosts or between host and router in the same network. This procedure is called direct routing.

Direct routing takes place when the source node checks the destination address and determines that it is in the same IP network, the same subnet, and the same

physical network. In this case, the host uses the Address Resolution Protocol (ARP, see Section 3.4.2) to send a broadcast to the local network and map the IP address to a link layer address, e. g. an Ethernet address. The node then encapsulates the datagram into a link layer frame and directly sends it to the datagram's destination.



Animation 3.4-1: Direct Routing

If the destination address is on a different subnet or entirely different IP network, the datagram is sent to a router connected to the same local network. The router maintains routing tables to be able to point IP traffic from each local host into the appropriate direction.

When a datagram must be forwarded through one or several routers, the procedure is referred to as **indirect routing**. It relies on routers forwarding datagrams between different networks thus allowing the datagram to reach its destination. Each node keeps a list of nodes and routers that are locally connected to it. Usually, on each subnet at least one or two routers are available as default routers to forward datagrams. A router is called a default router if one or several nodes use it by default for routing issues.

The sending host encapsulates the IP datagram into a link layer frame. The link layer frame is directly transmitted to the default router. The router extracts the datagram from the frame and examines the IP datagram header. It processes the header fields, decrements the time-to-live field and recalculates the header checksum.

The router looks at the datagram's destination address to determine whether it is a local address. If the destination address is on a local network, the router applies ARP (see Section 3.4.2) to find out the destination´s link layer address and then delivers the datagram encapsulated in a link layer frame.

If the destination is not local to any network the router is connected to, the router forwards the datagram to another router. This procedure continues until the datagram reaches its destination network. Each router updates the time-to-live value as well as the header checksum. Intermediate routers also modify the values for the datagram identification and the fragment offsets if it is necessary to fragment a datagram between sender and receiver.

3.4.2 The Address Resolution Protocol (ARP/RARP)

If it is possible to match the hardware address of a computer and the IP address, linking of an IP address to the hardware address becomes trivial. But for the most common network media, this is not feasible. For example, the MAC (media access) address, as used in local networks such as Ethernet, comprises six bytes. Thus, it cannot be mapped directly into an IP address. If you want to pad IP addresses and insert them into the MAC address, you have the problem, that MAC addresses are hard-coded in the network interface card. Because of these problems, in the Internet another approach is supported.

When a host determines that an IP datagram is destined for a node on the local network, it uses the **Address Resolution Protocol (ARP)** to get the address. The Reverse Address Resolution Protocol (RARP) takes a similar approach to allow hosts to find out their own IP address on the basis of their hardware address in the local network.

The operation of ARP is simple. A host whishes to send an IP datagram to another host in the local network. If the host only knows the IP address, it has to determine the hardware address.

The host that wishes to send the IP datagram broadcasts an ARP request to the local network (see Fig. 3.4-3). The request is intended for the host to which the IP datagram shall be transmitted. All the systems in the network process the broadcast request, but only the host with the specified IP address responds. The concerning host recognizes its own IP address in the broadcasted request and responds with its own hardware address. The host originating the ARP request then uses the network address supplied by the responding host to send the datagram to this host (inside a frame).

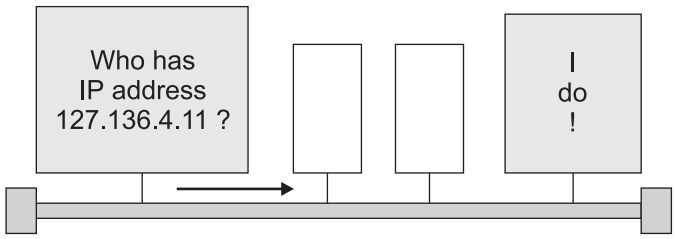


Fig. 3.4-3: Exemplary operation of ARP

As the name implies, the **Reverse Adress Resolution Protcol (RARP**) is simply the reverse of ARP. It is used by diskless workstations to get their assigned IP addresses. RARP requires that at least one host on the network is designated as an RARP server. The RARP source fills its own hardware network address in both the source and target address fields, the RARP server replies with the required IP address of the host.

3.4.3 Internet Control Message Protocol (ICMP)

The Internet Protocol aims at one major task: to move data from its source to its destination. But network layer protocols have to perform more tasks than transferring data. Systems rely on the network layer to coordinate different aspects of their operation including discovery of neighbours, controlling address assignments, and managing group membership. Furthermore, these protocols assist in reporting errors and providing diagnostic support.

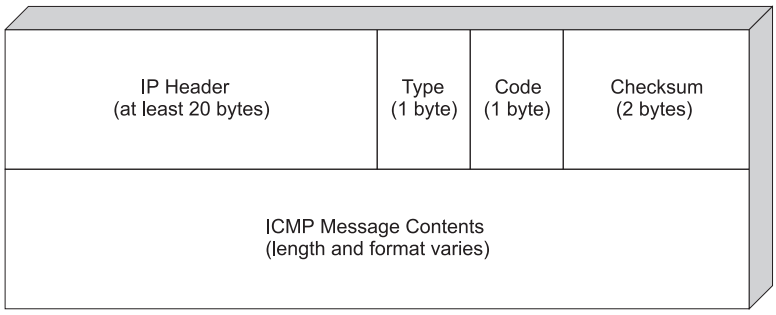
In the TCP/IP architecture, these functions are assigned to the **Internet Control Message Protocol (ICMP)**.

Although ICMP is a network layer protocol which is considered to be a protocol running parallel to IP, it sends its messages inside IP datagrams. Usually, ICMP is used for the following tasks:

- sending error messages about unreachable destinations,
- sending error messages about routes and gateways,
- sending echo requests and replies to indicate the status of reachable hosts, and
- sending error messages about traffic that has timed out (the TTL value has reached 0).

Additionally, ICMP can provide certain information to hosts such as the current time.

As illustrated in Animation 3.4-4 ICMP messages have a simple structure: The first field after the IP header is the **type field** with a length of one byte. This field indicates the function the message fulfills. The following field, the **code field**, has a length of one byte. It includes further information about the content of the message, e. g. a more specific description of an error. The **checksum** is a 2-byte number. It is applied to the ICMP message starting from its type field.



Animation 3.4-4: ICMP message packed in an IP datagram

The message field contains the IP header and the first 64 bits of that data- gram which caused the ICMP message to be sent. The message may also contain the IP address of intermediate routers between systems or a list of available routers in a network.

Table 3.4-1 gives an overview of ICMP error messages.

Tab. 3.4-1: *ICMP error messages*

Message	Description
Destination Unreachable	A datagram cannot reach the destination host or application.
Time Exceeded	The TTL has expired at a router.
Parameter Problem	A bad parameter has been discoverd in the IP header.
Source Quench	A router or destination is congested.
Redirect	A host has routed a datagram to the wrong local router.

Apart from error messages, ICMP also supports messages which provide information about the network, for example if a router is active, about the round-trip time to a certain host and back, or about the address mask. These ICMP messages include:

- **Echo:**request and reply messages exchanged among hosts and routers.
- **Timestamp:** request and reply messages that probe the round-trip time and find out the clock setting of the target system.
- **Address Mask:** request and reply messages that allow systems to discover the address mask which should be assigned to an interface.

Additionally, ICMP messages can be applied to exchange routing information. Hosts often use ICMP to request a list of available routers when they boot up and initialize their routing tables. By periodically broadcasting the current routing preferences, routers ensure that the hosts in their networks do not try to use a router that is inappropriate.

The probably most common use of ICMP is the ping application. **Ping** (Packet InterNet Groper) sends an ICMP echo request to a specific host. The host responds to the ICMP echo request by sending an ICMP echo reply. Hence the purpose of ping is to find out whether the remote host is reachable or whether the network connection for a local host is properly configured and installed.

Example:
To find out if the server of the University of Hagen is active and reachable, try to send a ping message to it by typing:

```
ping www.fernuni-hagen.de
```

The answer will depend on several aspects such as your distance from the university or the ISP your are using. On a Windows 95 system it may for example look like this:

```
Pinging hickory.FernUni-Hagen.de [132.176.114.35]
with 32 bytes of data:
```

```
Reply from 132.176.114.35: bytes=32 time=1ms TTL=254
```

In the first line you see the action, the address ’www.fernuni-hagen.de’ is contacted. The DNS has resolved the address finding out the host name ’hickory’ and the IP address of this server ’132.176.114.35’. In the following line, the answer of the server ’hickory’ is displayed. The server replies within the duration of 1ms. The time to live (TTL) of the received IP packet is displayed as well. This reply from the server is usually repeated three times.

3.5 Summary

In this chapter, first the concept of naming and addressing in the Internet was outlined. The structure of IP addresses and the different classes of IP addresses were described. In this context, the functionality of the Domain Name Service (DNS), a service converting Internet names into addresses, was explained. The hierarchical concept of the DNS was outlined using an exemplary host name resolution. The task of the DNS and the concept of addressing in the Internet introduces basic knowledge which is required to understand the functionality of the Internet protocol (IP). This very basic protocol utilized in the Internet defines the rules to package data into IP datagrams and to transmit these datagrams via a network. Finally, the Internet Control Message Protocol was described which is a kind of helper protocol of IP. ICMP fulfills tasks such as sending error messages and discovery of neighbouring hosts.

4 Protocols at the Transport Layer

4.1 Goal of the Chapter

In this chapter, the operation of the tranport layer of the TCP/IP protocol stack is explained. The basic protocols at this layer, the User Datagram Protocol (UDP) and the Transmission Control Protocol (TCP), are outlined. The concepts of ports and IP sockets are described.

4.2 Introduction

At the transport layer two major and very different transport protocols are operating: TCP (see Section 4.3) and UDP (see Section 4.4). Each supports applications with different requirements concerning performance and reliability of data transmission. UDP supports transactions with no added reliability or guarantee of delivery, on the other hand TCP offers a connection oriented, reliable, and guaranteed delivery service to applications. We want to start our explanations with the UDP protocol, as this one is much easier to understand and includes some basic aspects which are necessary to understand the TCP protocol.

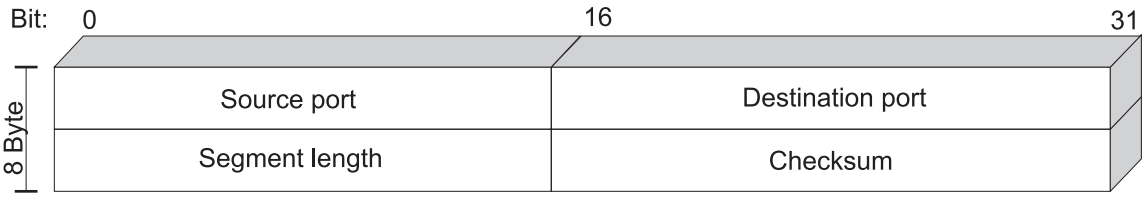
4.3 User Datagram Protocol (UDP)

As already mentioned in earlier sections, the question of network reliability is deferred to the higher layers. The **User Datagram Protocol (UDP)** also defers this task to the higher layers. UDP ist a simple protocol providing no error detection, no error correction, no connection-oriented links, no handshaking, and no verification of delivery order. UDP only offers the basic datagram delivery.

Applications based on UDP are often more simple so that they do not need to maintain connections. UDP applications may consist entirely of requests and replies to requests. UDP provides a connectionless delivery service between two hosts, offering a service over a particular protocol port. UDP delivers messages and responses. But it provides no help with regard to the delivery of datagrams in correct order, whether the datagrams have been delivered at all or detection and retransmission of dropped datagrams.

Without adding any features, UDP is easy to implement and requires minimal overhead. It is often used for low-intensity tasks performed in the background, such as resolving host names through the Domain Name Service (DNS, see Section 3.3.5)

Each UDP datagram contains a single message which may be a request or a reply. An UDP packet is transmitted according to packet-switching in datagram mode. This means that it travels independent from other UDP packets across the network. UDP provides minimal transport layer functions and applies a minimized header structure as depicted in Animation 4.3-1



Animation 4.3-1: UDP header

The UDP header comprises 8 bytes, consisting of four two-byte fields. The first field contains the source port number, the second the destination port number. The value of the third field represents the length of the UDP datagram. The fourth field includes the UDP checksum.

Port numbers refer to processes running on networked systems. Protocols at the transport layer no longer refer to specific nodes or network interfaces. The data link and network layers deal with this task. Transport layer protocols identify source and destination processes.

The **source port** of a datagram is determined by the host which generates it. In contrast to that, the **destination port** is more difficult to find out. The destination port specifies a particular programme or process running on the destination host. The sending host would have to query the destination host and ask for the right port number before it could send its datagrams.

To avoid the necessity to query a remote host for a port number, special port numbers are assigned to particular applications. Client hosts can specify the application they want to connect to by addressing the datagram to that port which is associated with the application. These port numbers are called **well-known ports**. All well-known ports are assigned within the range of 1 and 1023. Ports with higher values than 1023 up to 5000 are called **ephemeral ports** and port values higher than 5000 are reserved for servers that are not well-known across the Internet, for example if a new client/server application is developed.(See Table 4.3-1)

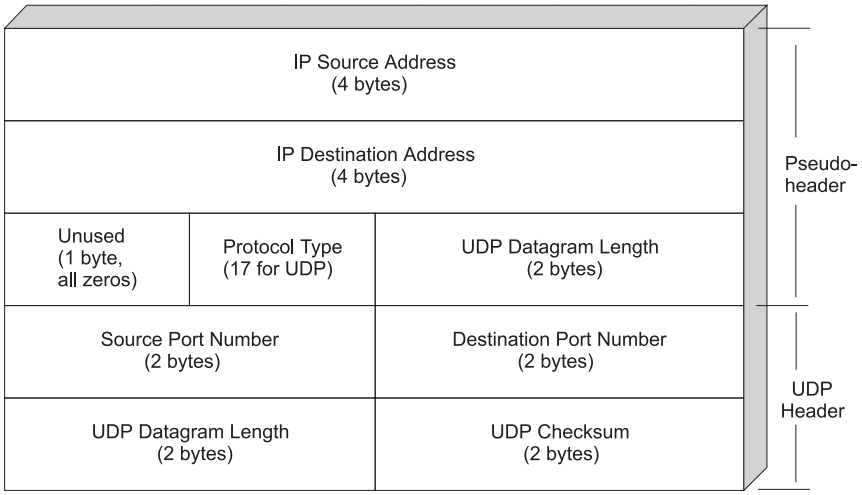
Tab. 4.3-1: *Range of the ports at the transport layer*

Port type	Port number
well-known port	1-1023
ephemeral ports	1024-5000
experimental ports	>5000

The server software needs a specific port number, but the client software does not need to operate with the same port all the time. It is even more preferrable that each client session runs on a unique port. This allows a host to have multiple instances of e. g. telnet or FTP sessions between the same two hosts. For this purpose, ephemeral ports are used.

The UDP **length** field consists of 16 bit. It indicates the length of the entire datagram including header in bytes. Thus, the upper limit of the UDP datagram size is 2^{16}=65.536 bytes. The lower limit is 8 bytes.

The **checksum** is calculated using a **pseudo-header** that uses IP addressing information from the IP header. The source and destination IP addresses and the protocol code for UDP are added to the UDP datagram and a checksum is calculated on the entire datagram plus pseudo-header (see Animation 4.3-2). Including the source and destination IP addresses in the checksum allows the destination host to verify not only that the UDP datagram arrived uncorrupted, but also that the UDP datagram arrived at the desired destination.



Animation 4.3-2: Creation of a pseudo-header to calculate the UDP checksum

The rest of the UDP datagram consists of data. The maximum datagram length can be configured but is usually set to 8.192 bytes.

4.4 Transmission Control Protocol (TCP)

The TCP protocol is crucial to the understanding of TCP/IP networks. Like UDP, the **Transmission Control Protocol** is a transport layer protocol. But in contrast to UDP which offers little support concerning reliability or guarantees, TCP reliably connects hosts across a network.

In contrast to UDP which is connectionless, TCP operates connection oriented using virtual circuits. Each circuit behaves as if a direct two-way connection between the communicating hosts exists. TCP provides end-to-end reliability, requiring that communicating hosts coordinate and agree to make connections and acknowledge receipt of network traffic. Each UDP datagram is an individual message or reply. In comparison to that, each **TCP segment** is related to the segment before and after it. The first and the last segments in a sequence require special treatment.

TCP uses several techniques to enhance connection performance. TCP monitors the link to make sure that segements are neither too large nor too small and are not transmitted too fast or too slow for the virtual circuit between the two systems. UDP interaction works only in one direction: a request goes from the requester to the network, the response comes from the responder back to the requester. TCP interaction is duplex allowing information to flow in both directions in each TCP segment. A host responds to a TCP segment requesting information by sending another TCP segment containing not only the information, but also control information about the currently transmitted segment and which segment it expects next from the remote host.

A further difference between UDP and TCP is that UDP allows broadcast and multicast transmissions. Broadcast means the transmission from one host to every reachable host, mulicast means the packet transmission from a single host to a certain group of hosts. TCP supports only host-to-host communication (unicast communication) because each receipt of a segment must be acknowledged.

Summarizing the above, TCP is different from UDP in three basic aspects:

- virtual circuits,
- reliable connections, and
- performance optimization.

4.4.1 The Virtual Circuit

The concept of virtual circuits has already been outlined in Section 1.4. This type of packet-switching mode is applied in TCP connections as well.

TCP connections behave as if there was a hard-wired connection between processes on the two connected hosts. When a process initiates a TCP connection with another process, the two processes negotiate to open the connection. Each one must agree to participate. But there is no guarantee that a TCP circuit can be initiated and maintained.

Somehow, a TCP virtual circuit is comparable to a telephone link: one person/process initiates the telephone call. The person at the other end has to answer the phone. A conversation takes place if both sides agree to the conversation. If the person being called is unavailable, the conversation cannot be started.

Each connection is identified uniquely with a combination of each host's IP address and port number for the connection. The combination of port number and IP address of the host on which the process is running is referred to as **TCP socket**[Fei99]. A TCP port uniquely identifies a TCP connection. Similar to UDP, servers using TCP for their transport protocol use well-known port numbers for offering different network application services. The process specifies a port number to establish the connection. The IP layer of the networking software indicates the host address.

Example 4.4-1:
A client process connecting to a Telnet server process running on the host with IP address 130.42.88.22 (see Fig. 4.4-1) specifies the Telnet port number (port 23). Hence the TCP socket of the Telnet process on server side is (IP address=130.42.88.22, port=23). The client assigns another TCP port number, which is not a well-known port but an ephemeral port as its own TCP port, in this example port 3358. This results in its own TCP socket: the client's IP address (128.36.1.24) and TCP port number (3358).

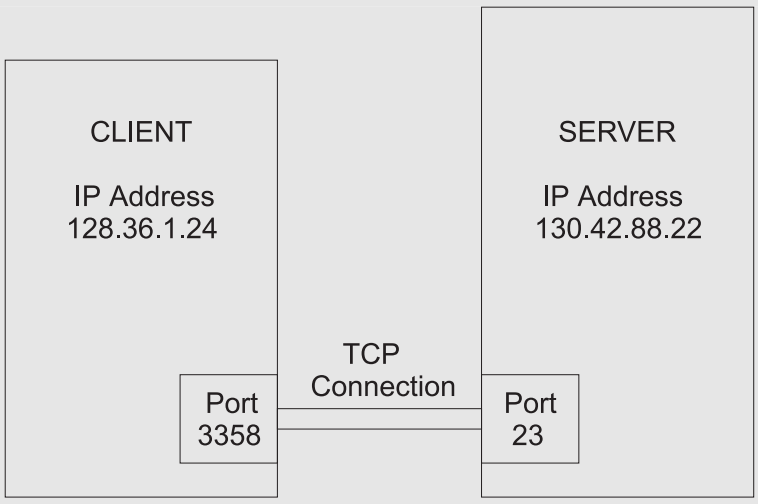


Fig. 4.4-1: Socket address for TCP connection

A single host can maintain more than one TCP connection through a single port because the incoming TCP segments are differentiated by their different source sockets (see Fig. 4.4-2). A server can maintain multiple connections made through a single client host. The server distinguishes incoming TCP segments by the IP addresses of the clients and by the port numbers.

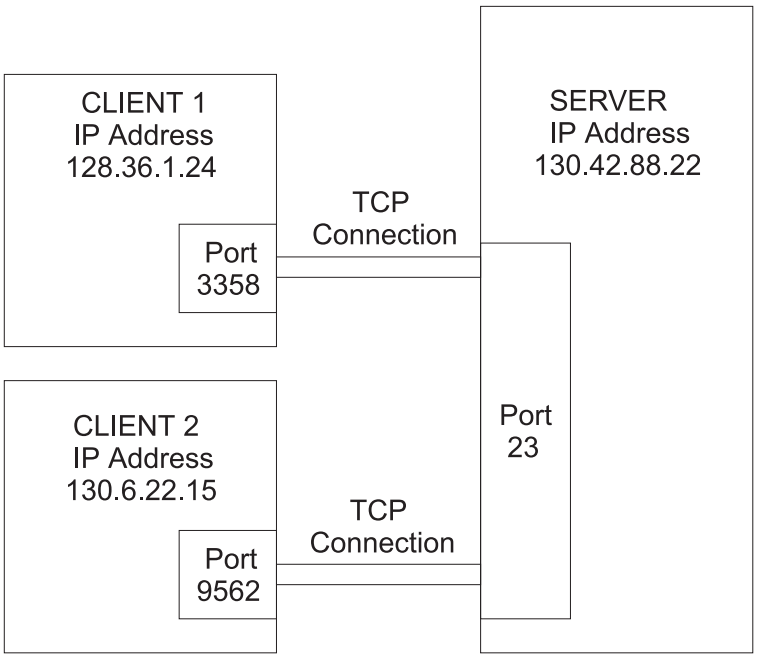


Fig. 4.4-2: Multiple clients connecting to a server socket address

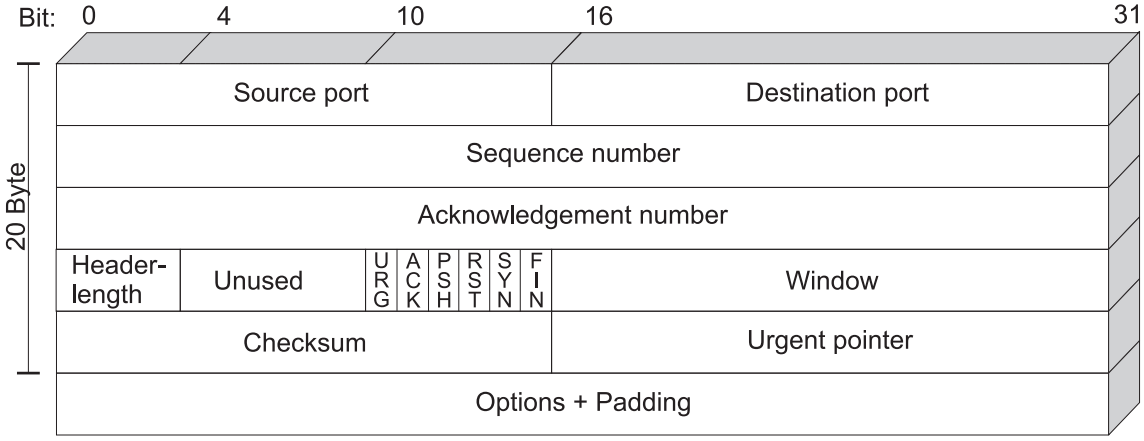
Example:
Client 1 has established a TCP connection to port 23 of the server to perform a Telnet session. When Client 2 establishes a TCP connection for a telnet session as well, it also connects to port 23. The server must be able to distinguish the packets which are arriving at port 23 from the different clients. This differentiation is possible since the clients have different IP addresses and TCP ports which are indicated in the TCP packets.

4.4.2 TCP Reliability Features

Concerning reliability, UDP only offers the checksum which is calculated from the pseudo-header and the UDP datagram. It helps the receiving systems to verify that the datagram arrived at the right address without corruption. The usage of the checksum in a UDP implementation is optional. TCP offers enhanced reliability features. In TCP implementations, the inclusion of a checksum for the entire TCP segment and the TCP header is obligatory. TCP calculates the checksum for a pseudo-header that has the same format as the UDP pseudo-header (see Animation 4.3-2).

Apart from the checksum, TCP implements additional reliability features. A three-step handshaking routine is required to initiate a TCP link. This routine will be described in detail in Section 4.4.4. Furthermore, every segment sent by each host must be acknowledged as soon as it is received. This means that the receiver has to inform the sender, that it correctly received the TCP packet. Every segment includes information about the next segment to be sent. Revocery mechanisms at both ends deal with lost segments, duplicated segments, and segments which arrive out of order. These features are necessary to realize a reliable transport, but they also add overhead which is necessary to implement TCP.

4.4.3 TCP Header Structure



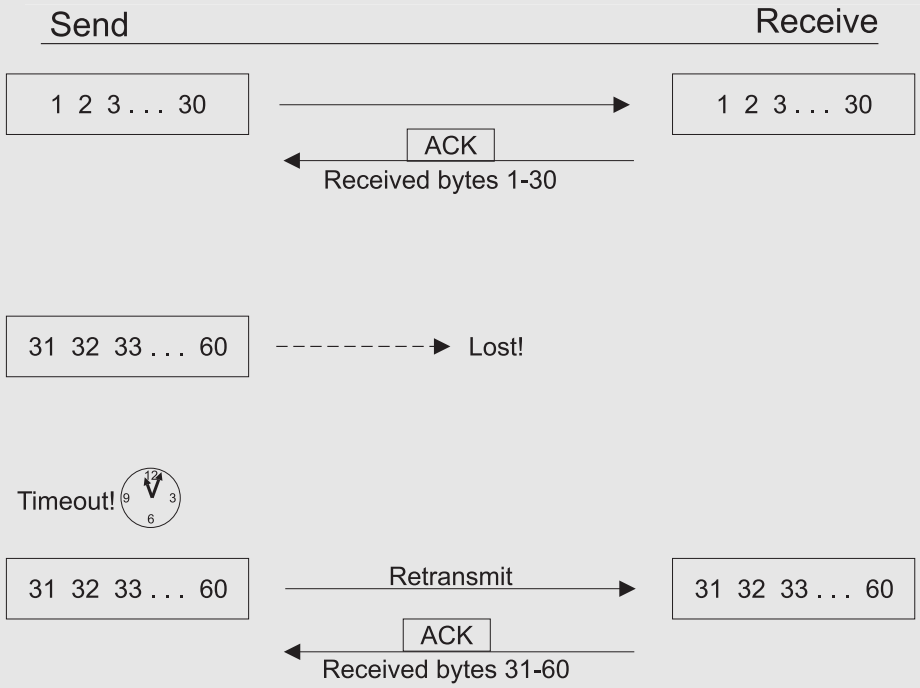
Animation 4.4-3: TCP Header

The standard TCP header comprises 20 bytes, but it can be longer if options are used (see Animation 4.4-3). The first and the second field of the TCP header are the **source and destination port number**, each of them with a length of 2 bytes. The initiating host assigns itself an ephemeral port number, which is usually a randomly chosen value greater than 1023 (see Table 4.3-1). The destination port number is the well-known port associated with the service requested from the remote host.

Each byte of a TCP stream is numbered, starting with an arbitrary number selected by the sending host. TCP connections are duplex which means that data is transmitted in both directions at the same time. Each host selects an arbitrary starting point for numbering the bytes of its own data stream. The **sequence number** in the TCP header indicates the number the sending host has assigned to the first byte in the current segment. The numbering starts at an arbitrary number between 0 and 2^{32}-1 and restarts at zero when the highest value has been reached.

The **acknowledgement number** contains the value of the sequence number which is expected next from the other side. If the acknowledgement does not arrive within a timeout interval, the data is retransmitted.

Example 4.4-2:
The last TCP byte reaching the receiver had the sequence number 30 (see Animation 4.4-4). Thus, the receiver's acknowledgement number will be 31. This value identifies the number of the next byte which is to be sent by the other side. After receiving this acknowledement the sender transmits the next bytes numbered for example 31 to 60. Assuming that the sender does not receive the next acknowledgement with the number 61 within a certain time interval it retransmits the bytes 31-60.



Animation 4.4-4: TCP acknowledgement, timeout and retransmission

The usage of sequence number and acknowledgement number will be explained further in the next section dealing with the three-way handshake protocol.

The field **header length** indicates the length of the TCP header. The length of the TCP header may be extended when using TCP options. The maximum header length is 60 bytes. The header length is a multiple by 4 bytes.

The TCP flag bits are employed to negotiate and manage connections:

- **URG:** If set to 1, urgent data is included in the segment and the urgent pointer in the TCP header is used to point at the urgent data.
- **ACK:** Indicates that the acknowledgement number in the segment header is valid. If this flag is set to 0, the acknowledgement field must be ignored. The flag is 0 as long as no segment from the other side of the connection has been received. In this case there is nothing to acknowledge. Once a connection is established, it should always be set to 1.

- **PSH:** The data should be delivered to the application as soon as possible.
- **RST:** This flag bit indicates an error, the connection must be reset.
- **SYN:** During setup of a TCP connection, this bit is set to indicate that the synchronization of sequence and acknowledgment numbers takes place.
- **FIN:**The sender has no more data to send.

The **window size** is a dynamic value that varies depending on how much data the process at either end of the circuit is willing to accept at any time. The window size is specified as number of bytes that the receiving host accepts within its window. The window size may depend on e. g. the processing speed or the temporal storage capacities of the receiving host. The process at either end can modify its window size at any time during the use of the TCP circuit. Large windows increase the efficiency of the link. Smaller windows mean that the receiving host cannot process the incoming data quickly enough to keep up with the current pace.

The TCP **checksum** has already been mentionned. It is calculated for the whole segment and the TCP pseudoheader.

The **urgent pointer** is only employed if the URG flag bit is set. It is used to indicate the sequence number of the last byte which is part of the urgent data.

Example 4.4-3:
Urgent data can be the interrupt key during a Telnet session. It is used to interrupt other processes running on the respective server. The interrupt key should even be accepted if the server is waiting for the end of a process.

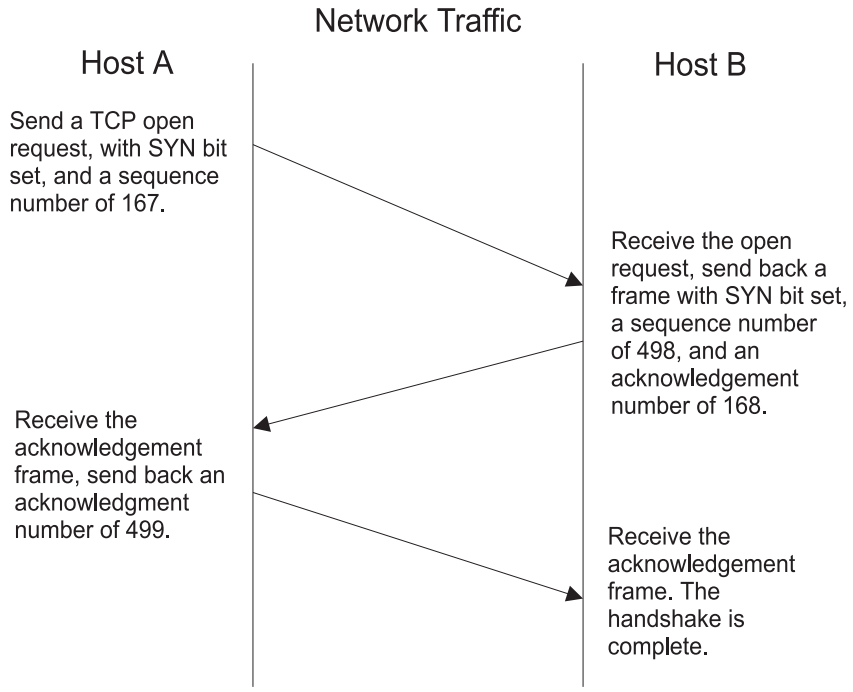
The most common **TCP option** is to specify a maximum segment size. Hosts can avoid fragmentation of TCP segments by letting the host on the other end know the largest segment size it is willing to accept.

TCP has certain limitations when fast networks are used. The window size has to be specified in two bytes, thus allowing a maximum size of 64 kbytes. Where capacity is high and latency low, a small window size is optimal since messages can make the round trip time relatively quickly. If a physical transmission link with the same capacity but higher latency is used (e. g. a satellite link) a larger window size performs better to avoid long delays between transmissions. Waiting for an acknowledgement always causes an additional delay in data transmission. TCP options like sliding windows and time stamping make it possible to support new faster networks by allowing larger window sizes. Furthermore, in high-speed networks the sequence number space can be exhausted very quickly. Time stamping helps to eliminate ambiguity caused by wrapped sequence numbers.

Padding ensures that the TCP segment length is a multiple of 32 bits.

4.4.4 Three-Way Handshake

To set up a TCP connection between two hosts, the **three-way handshake** has to be performed. The name of the procedure stems from the fact that three messages SYN (for synchronization), SYN, and ACK (acknowledgement) have to be exchanged to start the connection. The three-way handshake relies on the fact that in TCP connections all the segments have to be acknowledged to provide a reliable link.



Animation 4.4-5: Steps of the three-way handshake

The opening of a TCP connection consists of the following steps:

1. Host A sends a segment to host B. This segment requests host B to open a TCP connection with host A and informs host B about the opening sequence number which has been randomly chosen by host A. In Animation 4.4-5 the sequence number 167 has been chosen by host A. The SYN flag is set indicating that the circuit is being synchronized.

2. Host B sends an acknowledgement of the initial segment to host A . The ACK flag is set ’on’. The opening sequence number is incremented by one (in Animation 4.4-5 it is incremented from 167 to 168) and written into the acknowledgement field. Host B choses its random sequence number, in our example 498. The SYN flag is set as the synchronization has not been finished yet. When host A receives the acknowledgement, the connection from host A to host B is established, but the link from B to A has to be validated.

3. Host A responds to host B's acknowledgment with an acknowledgement. The sequence number of B is incremented by one and put into the acknowlegement field. In Animation 4.4-5 it is incremented from 498 to 499. The ACK flag is set 'on' as A now knows the correct sequence number for the next segment from B. The SYN flag is set 'off' because the synchronization is complete as A responds with this message. The responding messages gets an incremented sequence number from host A, which is 168 in our example.

When the handshake is completed, both sides can start sending data. The two processes continually acknowledge the receipt of data.

4.4.5 Transmission Windows

Once the connection has been established, each side starts data transmission. While most of the application data may be transferred in one direction, TCP will also transmit data into the other direction to acknowledge the receipt of each segment that has been received. In response, the process sending the application data will also acknowledge the receipt of the acknowledgements of the other side.

But instead of waiting for acknowledgement of the receipt of every TCP segment before sending the next segment, TCP implementations define a certain number of bytes, the **transmission window**, the process will send before it expects an acknowledgement from the other host.

The size of the window is determined on the basis of the maximum number of bytes the host at the other side will accept and on the basis of the time it takes to transmit data from the first host to the second and back again, the **round trip time (RTT)**.

Example 4.4-4:
The sender transmits 30 bytes before it expects an acknowledgement. During the time the sender waits for the acknowledgement, it does not send any further data. If the sender does not receive an acknowledgement after a certain elapsed time, it retransmits the 30 bytes. After receiving the acknowledgement number 31 from the receiver side it continues with the next 30 bytes which have sequence numbers starting from 31 to 60.

4.4.6 TCP Performance Features

The TCP protocol comprises four important performance features which have been added to the original version of the transmission control protocol[Los99]:

- slow start,
- congestion avoidance,
- fast retransmit, and

- fast recovery.

The **slow start** addresses the problem of regulating the speed of transmission of segments by observing how fast the other side acknowledges the segments that have already been sent. If the acknowlegdements arrive very fast, the transmission window can be increased. If the acknowledgements come in slowly, the transmission window may be decreased.

The **congestion avoidance** is usually implemented in conjunction with the slow start. It provides a mechanism to deal with lost packets. Congestion avoidance slows down data transmission speed when the sender receives the indication that packets have been lost.

Duplicate acknowledgements may be caused by lost segments or by segments that have been delivered out of order. If a segment is lost, the basic TCP implementations have to wait for a timer to expire before retransmitting the missing segment. In case of **fast retransmit**, the waiting time can be avoided. If three or more duplicate acknowledgements are received in a row, the TCP process can assume that a segment is missing and decide to retransmit it.

Receiving duplicate acknowlegements means that somewhere along the route there may be a congestion. When the fast retransmit algorithm is active, the**fast recovery** algorithm dictates that the TCP implementation starts the congestion avoidance algorithm.

4.5 Summary

In this chapter, the the transport layer protocols which are employed in the Internet were explained. The basic tranport protocols in the Internet are the User Datagram Protocol (UDP) and the Transmission Control Protocol (TCP). UDP provides connectionless delivery service between two hosts without delivery guarantees. In contrast UDP, TCP operates connection-oriented and provides transmission reliablity. This feature is realized e. g. by using a checksum in the TCP header, performing the three way handshake which leads to the acknowledgement of all the data which has been sent during a TCP connection, and by initiating retransmission of lost packets.

5 Telnet and Remote Utilities

5.1 Goal of the Chapter

During the development of the Internet its dominant application was the remote access to computing resources. Via a terminal session the user was able to connect to remote hosts as if the terminal was connected directly to the host.

As e-mail and later on the World Wide Web were developed, these applications became of growing importance. Remote terminal sessions have lost their position as the dominant Internet application.

A couple of years ago, the only type of Internet access was the shell account. A user dialed into a Unix host, started the terminal emulation software, and used the Unix hosts Internet access for e-mail, file transfer, and terminal sessions with other Internet hosts.

Terminal sessions have remained a useful tool for controlling different network devices, as well as accessing mainframe and multiuser systems.

This chapter outlines the most important terminal emulation protocols for remote computing, the telnet protocol and the remote utilities.

5.2 Telnet

In the 1970ies most computing was performed through remote terminal sessions. A user accessed the mainframe via a terminal connected to a large multiuser system. The terminals were usually connected to the terminal server as shown in Fig. 5.2-1.

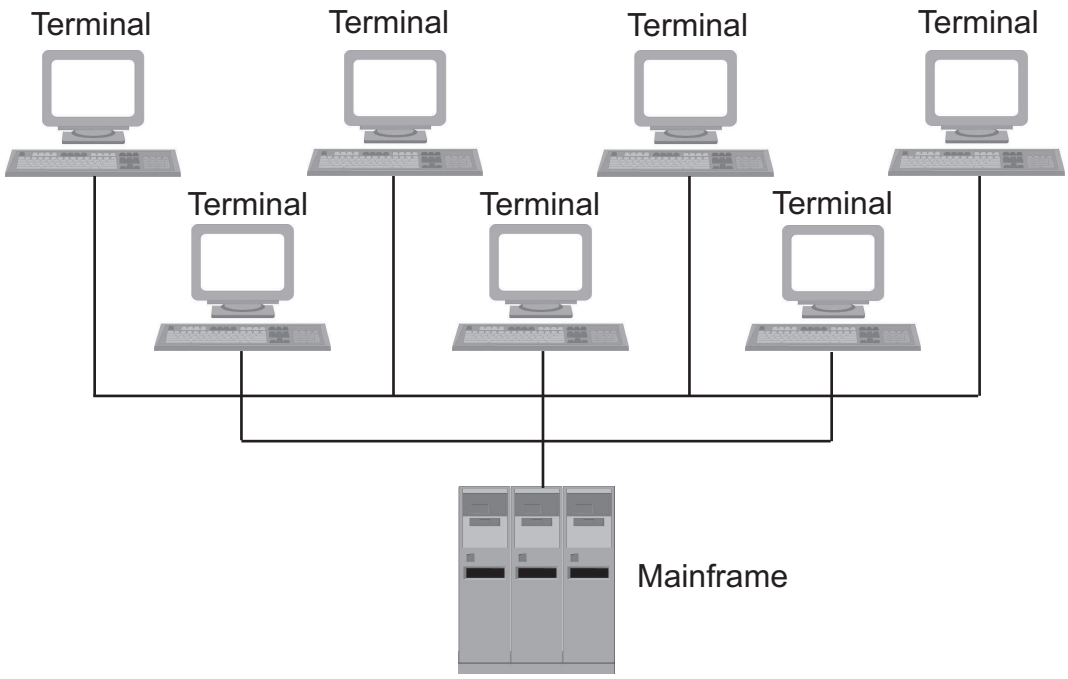


Fig. 5.2-1: Access to the mainframe via directly wired terminals

Each vendor sold ist own terminals. So if you wanted to connect to e. g. an IBM mainframe you also needed an IBM terminal. As long as companies only had one mainframe computer and the adequate terminals of one vendor, this structure worked allright. But as an increasing number of companies extended their resources to several mainframes which were bought from different vendors, problems arouse. Since the terminals of one vendor were not compatible to the mainframes of another vendor, the first alternative was to install several terminals, one for each mainframe. As the development also led to connecting mainframes to networks another solution had to be found.

Telnet, the Telecommunications Network Protocol, was developed to solve the problem of interoperating between different computer platforms. With telnet a user can log in remotely to any other type of system across an internetwork. Telnet terminal emulation was the first TCP/IP application.

The problem of standardizing on a single terminal was reduced as most of the computer vendors decided to handle input according to the DEC VT-series terminals. The most important exception was IBM who kept building mainframes that required different remote terminal emulations to handle its specific terminal. Commonly known are the 3270 and the 5250 terminal emulations. Both are supported in common telnet software implementations. So, telnet provides emulation of various types of terminals, which can be used to access e. g. Unix computers, VAX/VMS systems or IBM mainframes

5.2.1 The Network Virtual Terminal

Interoperable terminals are not a problem when all terminals use the same keyboard and mouse and when all computers use the same encoding format for data..

But due to the different vendors of computers and terminals there exist several variations of keyboards and mouses. There are several variations of the DOS/ Windows/ Intel personal computer keyboards. Apple Macintosh computers, DEC VT-terminals, and IBM 3270 terminals have several other unique keys.

The mouse of different operating systems is also implemented in some variations. While Macintosh uses a single button mouse, Unix often uses three button mouses, and a Windows PC may have a two or a three button mouse.

Another obstacle in the interoperation of terminals is data encoding. Most systems use ASCII, the American Standard Code for Information Interchange, but often it is applied in different variations. Apart from the ASCII Code, the major exception in data encoding is supported by IBM who uses **EBCDIC** (Extended Binary Coded Decimal Interchange Code) and slightly different variations of EBCDIC for encoding mainframe data.

To deal with these difficulties telnet uses the **Network Virtual Terminal (NVT)**. It has been invented to allow different systems to interoperate through telnet. The NVT allows telnet clients to convert input from users into a form usable for the remote host. By using NVT, the server can modify output data so that it can be displayed from the remote client. Consequently, each terminal type and each host type requires one converter.

The converter interprets the key presses and mouse moves into a standard format. Pressing the Enter key on a PC or the Return key on a Macintosh results in the same value on the NVT. Each host uses a converter that accepts NVT input and interprets it so that it is usable by the specific type of system.

So, both systems, client and server use the virtual terminal emulation (see Fig. 5.2-2). The real terminal interacts with the local telnet client program. The telnet client program has to accept the keystrokes from the user’s keyboard, interpret them and display the output on the user’s screen in a manner that is consistent with the emulation in use. The telnet client opens a TCP connection to the telnet server that is accessed at the well-known port 23. The telnet server interacts with the applications and assists in emulating a native terminal.

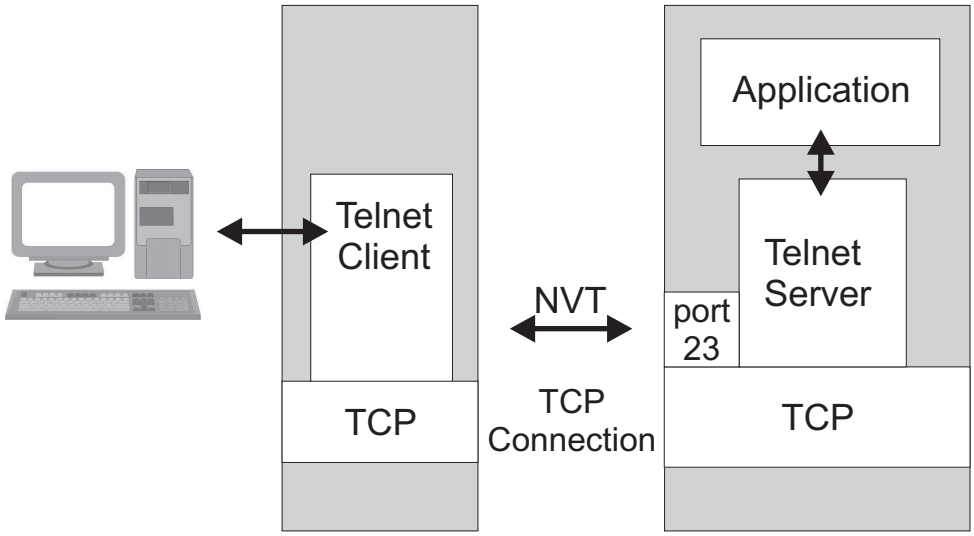


Fig. 5.2-2: Telnet client and server

The NVT protocol was modeled to operate in half-duplex and a line-at-a-time mode. NVT data is encoded using 7 bit ASCII code which is padded to 8 bits via an initial 0 bit. Each line ends with the combination of an ASCII Carriage Return <CR> and Linefeed <LF>.

After sending a line, the client waits to receive a response from the server. The server sends its data followed by a Go Ahead command, indicating that the client now can send another line.

5.2.2 Common Terminal Types

Usually, client and server stay in NVT mode for a short time, just long enough to negotiate the type of terminal to be emulated, such as ASCII VT 100 or IBM 3270.

5.2.2.1 ASCII Terminals

ASCII terminals are used to connect to Unix and DEC computers. These terminals use remote echoing of each character. This means, that each character is sent to the remote host and returned back to the client before it is displayed on the users screen. Futher it operates in full duplex mode. Characters flow in both directions at the same time. The client does not have to wait for a Go Ahead command to proceed.

ASCII terminals support interactive full screen applications but at the cost of a high amount of network overhead. The implemented ASCII character set is larger than that of NVT.

A lot of different ASCII terminals have been implemented offering slightly different features, e. g. VT52, VT100, VT220, ... The VT100 terminal is commonly used for remote login to Unix computers.

5.2.2.2 IBM 3270 and 5250 Terminals

3270 terminals operate in Block Mode. A user works with a screen of data at a time. When the user presses the Enter key, the information on the screen is sent to the server. The keyboard is locked while the host processes the data. Afterwards the host sends back one or more screens of data and unlocks the keyboard. 3270 terminals use EBCDIC 8-bit data encoding and operate in half-duplex mode.

5250 terminals are used to access AS/400 computers. They have similiar characteristics as the 3270 terminals.

5.2.3 Using Telnet

In a telnet session, the user initiates a connection to a remote host. The client and server negotiate a connection. The user enters his user ID and passoword. If these steps are completed successfully, the terminal session starts. From now on, the telnet session looks like any terminal session with a remote host. At the end of the session, usually the user has to exit the telnet client manually. Depending on the configuration and implementation the client may also close automatically.

Most of the user interaction with telnet takes place during starting and stopping it. When used from a command line, like DOS or Unix, the session starts with the command:

```
telnet [hostname|IP address] [port]
```

The use of the destination host name or address is optional. If you do not use a host name or address, telnet is simply started and you get a telnet prompt. If you want to connect to a specific host you can alternatively indicate the destination by using the host name or its IP address. If you use the host name, the name is resolved into an IP address, e. g. by DNS. In contrast, the IP address is used directly.

Telnet connections are established by default to the well-known port for telnet (23). But telnet can also be forced to connect to a non-well-known port by indicating the desired port in the command line.

Often, IBM 3270 or 5250 emulation is implemented separately. To access the IBM hosts you have to type

```
tn3270 [hostname]
```

```
tn5250 [hostname]
```

The start of a telnet session from a command line is shown in Fig. 5.2-3.

```
kronenbourg:/home/sommer>telnet amstel
Trying 132.176.12.12...
Connected to amstel.
Escape character is '^]'.


SunOS 5.7

login: sommer
Password:
Last login: Tue Aug 22 10:44:00 from Krombacher.fernu
Sun Microsystems Inc.   SunOS 5.7       Generic October 1998
amstel:/home/sommer>
```

Fig. 5.2-3: A telnet session started from the command line

Using telnet on a GUI (Graphical User Interface) system is slightly different. GUI telnet clients offer more options than in text-only implementations, eg. there are options for display, colors, fonts used, keyboard mappings, saving all or a part of a session in log file, or storing the information needed to access a frequently visited host. In Fig. 5.2-4 a GUI telnet session is shown.

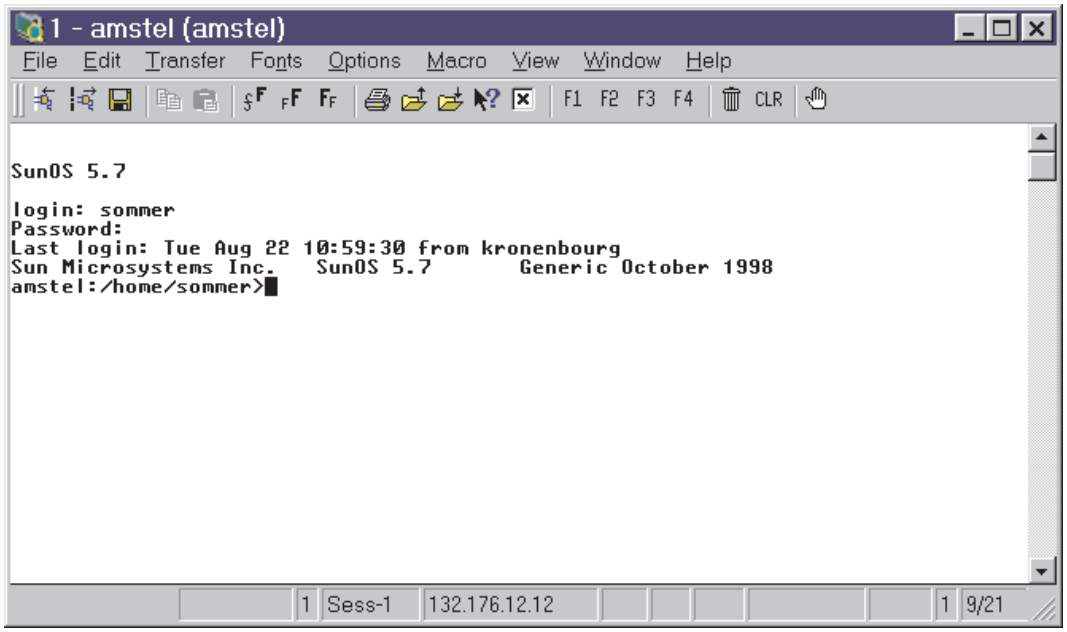


Fig. 5.2-4: A telnet session with a graphical user interface

5.2.4 Telnet Features

If you start telnet without indicating a remote host to which you want to log in, you will get a telnet prompt. Commands you can now enter are listed in Table 5.2-1.

Tab. 5.2-1: *Telnet commands available on a usual Unix or Linux implementation*

Command	Explanation
close	Close current connection
logout	Forcibly logout remote user and close the connection
display	display operating parameters
mode	try to enter line or character mode (’mode ?’ for more)
open	connect to a site
quit	exit telnet
send	transmit special characters ('send ?’ for more)
set	set operating parameters (’set ?’ for more)
unset	unset operating parameters (’unset ?’ for more)
status	print status information
toggle	toggle operating parameters (’toggle ?’ for more)
slc	change state of special character (’slc ?’ for more)
auth	turn on (off) authentication (’auth ?’ for more)
z	suspend telnet
environ	change environment variables (’environ ?’ for more)
?	print help information
!	invoke a subshell

The ! command invokes a subshell which means a new operating system command line. This is useful for running another program on the local host while the terminal session is still open.

The commands mode, send, set, unset, toggle, and environ are all used to modify certain parameters to customize the telnet connection.

But most of the command options are not relevant for everyday use of telnet. Normally these would only be modified for problematic connectios.

5.2.5 The Telnet protocol

Telnet works according to the client server model as already explained in Section 1.5. A daemon program at the server listens continously to the network for requests to open a telnet connection (see Fig. 5.2-5)

The telnet client, when receiving the user input from the local computers maps it onto the NVT. The data is passed to the server via the established telnet connection. The server maps the input from the NVT onto its own terminal and forwards the output to the server program which processes the data. The resulting output form the server program in turn is mapped into the NVT and passed onto the telnet client.

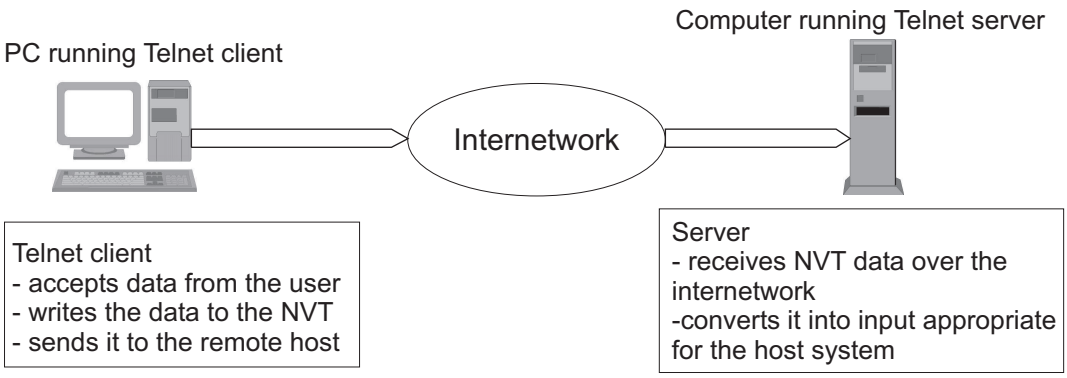


Fig. 5.2-5: Telnet server listening for a request to open a telnet session

Telnet applies TCP as a transport protocol as it requires a reliable virtual circuit between the two host interacting with each other. The client must be able to send data reliably to the server and the server has to respond reliably.

A telnet session is opened by indicating the IP address or the name of the remote host. If the host name is used, the client has to resolve the name to know the corresponding IP address. Therefore the DNS is used which has already been explained in Section 3.3.5. After that the client system initiates a TCP connection with the remote host. Building up a TCP connection among two hosts has been outlined in Section 4.4. The client tries to establish the virtual circuit by sending to the well-known port number 23 of the remote host. The client assigns itself a randomly chosen ephemeral port number. The remote host supporting telnet sessions runs a telnet daemon which accepts the request to connect over TCP port 23.

Each host, client or server can maintain several telnet connections at the same time. The number of TCP connections is only restricted by the amount of resources available. Also, a client can establish more than one telnet session to the same server. The data of the different telnet session can be differentiated by the ephemeral port numbers which are assigned to the sessions. This means that the telnet sessions all have the same destination IP address, destination IP port and IP source address, but the source ports are different. (see also Section 4.4.1 and Fig. 4.4-2).

5.3 Remote Utilities

Telnet provides robust, reliable, and secure remote login across an internetwork. For smaller TCP/IP networks, especially for Unix systems, the family of remote utilities or **r-utilities** has been developed. The r-utilities are simpler than general TCP/IP applications which they resemble, because they were designed to operate between Unix systems. They reduce the overhead wich is required to negotiate connections between dissimilar systems.

The r-utilities solve the security issue by allowing remote access based on the source system in use. This means that remote users are allowed to access a host if the user ID is listed in a special file. On Unix systems this is the .rhosts file that includes the originating host name coupled with a user ID. This may be sufficient for closed work

groups on Unix hosts. But many experts consider this method to be sensitive for security breaches if implemented on systems with less built-in security like personal computers.

The r-utilities offer the advantage that they are simple to implement. But in general they lack some of the features and options that are provided by more widely used TCP/IP applications. For example the r-login offers a basic remote login service including a simple terminal emulation. In contrast to that, telnet supports various different terminal emulations as well as other options. Table 5.3-1 list a number of remote utilities.

Tab. 5.3-1: *A list of common remote utilities*

Program Name	Utility	Purpose	Alternate Application
rcp	Remote copy	Copy a file from a remote host	FTP
rexec	Remote execute	Executes a command on a remote host	telnet
rlogin	Remote login	Remote terminal session	telnet
rsh	Remote shell	Execute commands on a remote host using shell scripts	telnet
rwho	Remote who	Display current users of remote host	Finger

5.3.1 Remote login connections

Just as telnet, r-login also uses the TCP transport protocol to initiate a connection among client and server. Once the connection has been established, the client sends user information to the server including the user ID on the client system and on the remote system. Usually the server also aks for a password. If the server receives a valid password, the session is continued, if not it is terminated.

A feature which makes r-login very convenient to the user is the possiblity to create a special file on a system that provides r-login service, that includes the user ID of authorized users and the names of the hosts from which they connect to the remote host.

If a user who is listed in the .rhosts file connects from his/her regular system he/she is not prompted for a password. This feature includes a significant threat of the security of r-utilities.

5.3.2 Features of remote utilities

The major advantage of remote utilities, is that they are simple to implement. But they support only few options and do not offer the level of interoperability of generally usable TCP/IP applications. So, nowadays the remote utilites are becoming of less importance.

5.4 Summary

Via Telnet a user can establish a terminal session to a remote host and access the computer resources as if he/she were connected directly to the host.

In the 1970ies, most computing was performed using terminals and mainframes. The terminals of different vendors were not compatible to the mainframes of different vendors. This lead to the development of Telnet allowing the interoperation of different computer platforms.

The terminals of different vendors use different keyboards, mouses, and data encoding formats. This problem was solved in the Telnet protocol by introducing the Network Virtual Terminal (NVT).

The Remote utilities have been developed for Unix systems. They are much simpler than TCP/IP applications which they resemble but they also support only few options.

6 IRC

6.1 Goal of the Chapter

The goal of this chapter is to understand what the Internet Relay Chat (IRC) is, what it is useful for and how it works. Special emphasis is put on the client-server-interaction while user commands, expansions and characteristics of IRC networks will be considered as well.

The IRC network (Internet Relay Chat) is a virtual meeting-point for users from all over the world to meet and chat in real-time. There are several independent networks of IRC Servers that enable users to log on to IRC.

6.2 Introduction

There has always been a need for people from various locations to meet and talk to each other. If there is a meeting somewhere and you are interested in what is about to be discussed, you go there and join the meeting. When you are no longer interested or your time is up, you just leave the meeting.

What you will need for a "real" meeting is information about

- what is being discussed
- where and when the meeting is scheduled
- how to get there
- who will host it and who is in charge
- who participates
- maybe a means to talk to people privately

The **IRC (Internet Relay Chat)** is a virtual meeting point. There are several "rooms" where people with various interests can chat in a so-called point-to-multipoint- or one-to-many-communication. You can also "rent" an own room and invite others to join you or leave a room at any time. Furthermore, you can talk to people you meet privately at any time and exchange documents or chat in a "secure" one-to-one communication.

The most significant advantages of a virtual meeting compared to a "real" meeting are

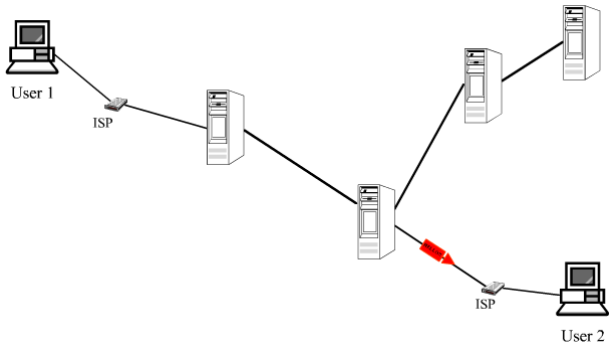
- no need to physically "go" anywhere, i.e. meet people from all over the world independent from where you are
- listen to several meetings at one time

IRC was originally developed in 1989 for **BBS (Bulletin Board System)** users to chat with each other easily. It was called "Internet Relay" Chat because chat messages might be transmitted over several servers ("relay stations") before they are

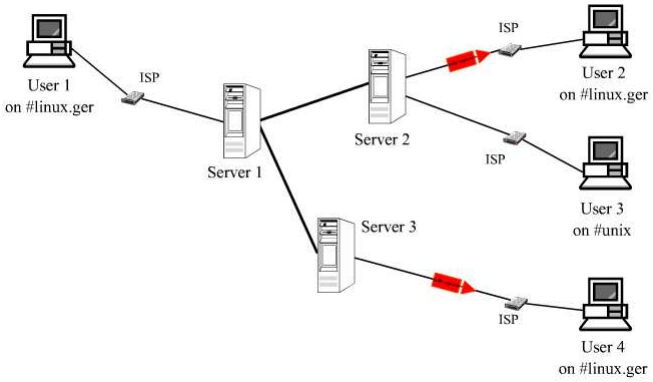
actually delivered to the receiver in "real-time" like with a long-distance telephone call that gets switched over several relay stations before it reaches the peer.

6.3 The Internet Relay Chat - System

As mentioned before, an IRC-server can be seen as a relay that manages informations for other users and from other servers of the same network. One of the most noticeable characteristics of the IRC protocol is that it allows users to gather in forums, called channels, providing a means for multiple users to communicate with each other. An IRC Client-software is used in order to connect to a server of the IRC network and give the necessary commands to the server and handle replies and other messages. A chat session can either be private (messages exchanged between two users no matter what channel one is on - "one-to-one" or public (i.e. everyone on the same "channel" will see what is being written - "one-to-many").



Animation 6.3-1: One-to-one Communication



Animation 6.3-2: One-to-many Comunication

IRC can therefore be called a teleconferencing system that is well suitable to run on several machines in a distributed manner.

6.4 The Internet Relay Chat - Protocol

The IRC protocol serves text-based conferencing. It was continuously developed since it was first documented formally as part of the RFC 1459 [RFC1459] in May 1993. The IRC protocol was developed on TCP/IP-based systems; however, this does not mean that this needs to remain the only field in which IRC can be used.

A typical setup involves a single process (the server) forming a central point for clients (or other servers) to connect to, performing the required message delivery/multiplexing and other functions. This distributed model, which requires each server to have a copy of the global state information, is still the most flagrant problem of the protocol as it imposes a limit to the maximum size a network can reach.

The IRC protocol provides several methods of transferring data between clients and, just like with other transfer mechanisms like E-Mail, the receiver of the data has to be careful about how to handle it. Note that **any** kind of data can be exchanged on the basis of the IRC Protocol. To learn more about security issues with the IRC protocol, see [IRC].

6.5 Chatting on the IRC Network

6.5.1 The IRC Server

The **IRC Server** forms the backbone of IRC, providing a point which clients may connect to in order to communicate with each other, as well as a point for other servers to connect to, altogether forming an IRC network. The only network configuration allowed for IRC servers is that of a spanning tree (Fig. 6.5-1), where each server acts as a central node for the rest of the net.

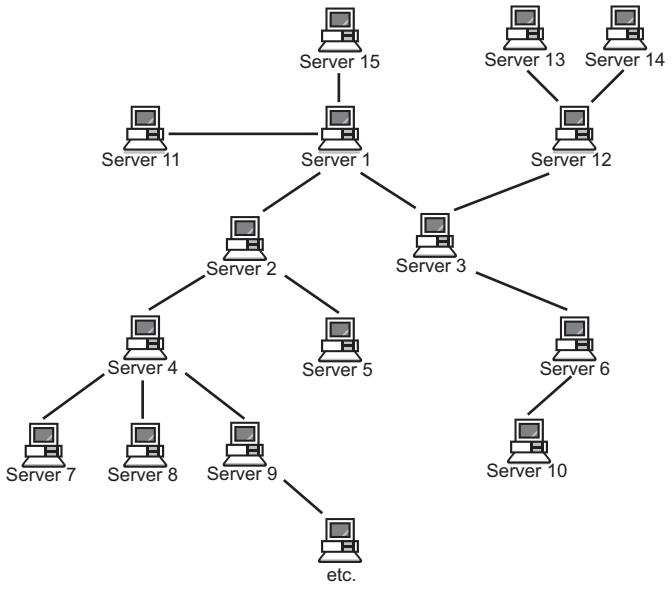


Fig. 6.5-1: Format of IRC server network

Inter-IRC-Server communication is provided via the TCP/IP protocol over which the text messages are carried and passed from one server to the other in a reliable manner. Messages adressed to a channel must be passed on to every server that has users in that channel while private chat messages between two distinct users only need to be exchanged between the two servers that these users are connected to. The servers behave just like relay stations for telephone calls which lead to the wording Internet Relay Chat. A channel is a named group of one or more clients which will all receive the messages that are addressed to that channel.

A server application for IRC is on the one hand dedicated to relaying datagrams to peering servers and on the other hand to provide the IRC Service to its clients. An IRC Client Application connects to the IRC Server and offers an interface for user interactivity.

To allow a reasonable amount of order to be kept within the IRC network, a special class of clients ("Server Operators") is allowed to perform general maintenance functions on the network. Server Operators are able to perform basic network tasks such as disconnecting and reconnecting servers as needed e.g. to prevent prolonged use of a bad network routing.

A controversially discussed feature of operators, however, is the ability to remove a user from the IRC network by ’force’, i.e. operators are able to close the connection between any client and server and prevent reconnection so that the user is then "banned" from the server.

6.5.2 The IRC Client

An **IRC client** is any machine connecting to an IRC Server that is not another IRC Server. Each IRC client is distinguished from other clients by a unique nickname having a maximum length of nine (9) characters. However, there are certain grammar rules for what characters may and may not be used in a nickname, like with filenames. In addition to the nickname, all servers must have the following information about all clients: the Internet name of the host that the client is running on or its IP Adress, the full name of the client on that host and the server to which the client is connected.

As described above, the user needs to provide information about what server he wishes to connect to. By choosing a server, the user also selects an IRC Network since every server is a member of a distinct IRC Network. In this example, an IRCNet Server is chosen. The user can choose any nickname he likes in order to hide his "real" identity or type in authentic information.

Below is an example of how a typical IRC-Setup looks like (using mIRC for Win32 in this case): see Fig. 6.5-2

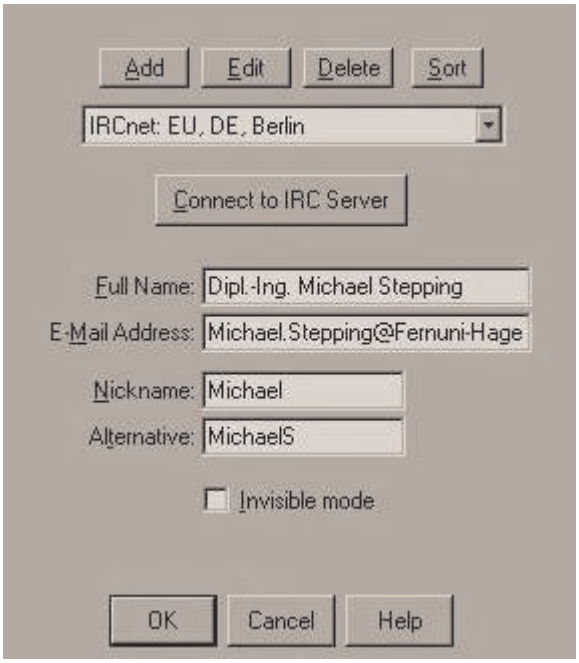


Fig. 6.5-2: A typical IRC Setup

Note that a user should always choose a server that is close to him to prevent unnecessary network lags. IRCNet Servers, for instance, are located all over the world and can be connected to, but users sitting in Germany should always choose a german IRC server.

After a successful connect, the user is in the so-called "0"-Channel. This is not a real channel and only channel-independent commands will work here.

The client controls its interaction with the server using text-based commands. Commands are marked with a starting /. The basic commands will work on almost any client while some clients provide additional commands and graphical user interfaces for convenience.

Example 6.5-1:
Here are some examples. Assumed, your nickname was "Michael" and you are interested in chatting with german-speaking people concerned with the Linux Operating System. This is what your status window will tell you upon successful connection establishment to the german IRCnet server:

Explanation: You are connected to the IRCNet using the server with the Internet Name fu-berlin.de. This IRC network consists of 58 servers with a current total of 50191 users. 1125 of these are directly connected to fu-berlin.de at this time. If any of the fu-berlin users wishes to contact any of the other 50191 users, fu-berlin serves as a relay station to other IRC servers. In the case of a direct chat between two fu-berlin-users, this server serves as a relay station for these two users only.

```
[13:31] *** Connecting to irc.fu-berlin.de (6665)
Welcome to the Internet Relay Network MStepping!~Michael.S@fernuni-hagen.de
```

```
Your host is fu-berlin.de, running version 2.10.3p1
This server was created Sat Jul 29 2000 at 16:38:01 MEST
fu-berlin.de 2.10.3p1 aoOirw abeiIklmnoOpqrstv

There are 50191 users and 7 services on 58 servers
149 operators online
6 unknown connections
24995 channels formed
I have 1125 users, 1 services and 1 servers

Message of the Day, fu-berlin.de

Local host: mpeg4.fernuni-hagen.de (132.176.255.200)

End of MOTD command.
```

6.5.3 Channels

A **channel** is created implicitly when the first client joins ("opens") it and the channel ceases to exist when the last client leaves ("closes") it. While a channel exists, any client can reference the channel using the name of the channel.

Channels names are strings (beginning with a ’&’ or ’#’ character) with a length of up to 200 characters. Apart from the the requirement that the first character has to be either a ’&’ or a ’#’; the only restriction on a channel name is that it may not contain any spaces (’ ’), a control G (^G or ASCII 7), or a comma (’,’ which is used as a list item separator by the protocol).

There are two types of channels allowed by the protocol. One is a distributed channel which is known to all the servers that are connected to the network. These channels are marked by the first character being a ’#’. The other type is a channel that only clients on the server where it exists may join. These are distinguished by a leading ’&’ character and can be called "local" channels. On top of these two types, there are various channel modes available to alter the characteristics of individual channels. See the description of the MODE command for more details on this.

To create a new channel or become part of an existing channel, a user is required to JOIN the channel. If the channel doesn’t exist prior to joining, the channel is created and the creating user becomes a channel operator. If the channel already exists, whether or not your request to JOIN that channel is honoured depends on the current modes of the channel. For example, if the channel is invite-only, (+i), then you may only join if your are invited. As part of the protocol, a user may be a part of several channels at once, but a limit of ten (10) channels is recommended for both experienced and novice users.

Basic commands:

Tab. 6.5-1: *Basic IRC Commands*

Command	Explanation
`/channels`	Receive a complete list of the channels that the server provides (24995 in the given case)
`/who`	Receive a complete list of users online at the moment (50191 in this example)
`/join #linux.ger`	You enter the channel #linux.ger. It might look as shown in Fig. 6.5-3.
`/who #linux.ger`	Some information about the channel linux.ger if not automatcally shown by your client upon joining that channel. Users with an @ are channel operators (e.g. the user Sunset) Users with an * are IRC operators (none here)
`/ping #linux.ger`	gives information about how long a signal between you and every user on #linux.ger will run.
`hello experts!`	everyone on #linux.ger sees your message with your name and perhaps your timestamp
`/whois Sunset`	gives information about the user Sunset: Sunset is ~Sunset@ducati.unix-ag.uni-siegen.de * Sunset Sunset on @#linux.ger @#mtw Sunset using Uni-Paderborn.DE [131.234.10.2] [irc.upb.de] University of Paderborn Sunset End of WHOIS list. As you can see, the user sunset is connected to a different IRC Server (irc.upb.de) but he is on the same IRC network (IRCNet) so that he is visible here. He is currently a channel operator on #linux.ger and #mtw.
`/query Sunset`	lets you enter a private chat with Sunset.
`/mode #linux.ger +o alc`	would give alc operator status in #linux.ger if you were an operatoryourself
`/join #mychannel`	lets you join #mychannel and will give you operator status since you are the first and so far the only user on this channel. Note that channel operators control IRC-channels and IRC operators control IRC-servers.
`/invite alc #mychannel`	lets you invite alc to join #mychannel
`/leave #linux.ger`	stop viewing the ongoing conversation on #linux.ger
`/nick Tom`	changes your nickname to Tom if available. Note that you can easily change your identity by changing your nickname; however you cannot choose what host and server information will be replied upon /WHOIS.
`/quit goodbye`	lets you quit your IRC connection leaving a quit message "good bye" to all those who were on the same channel as you or involved in a private chat with you

Fig. 6.5-3: Channel view

6.5.3.1 Channel Operators

Channels have hosts, so-called **channel operators** or short: chanops. They are considered to 'own' their channel. In recognition of this status, channel operators are enabled to perform certain changes to the status of a channel and to the participance of users. As an owner of a channel, a channel operator is not required to have reasons for their actions, although if their actions are generally antisocial or otherwise abusive, it might be reasonable to ask an IRC operator to intervene, or for the users just leave and go elsewhere and form their own channel.

The commands which may only be used by channel operators are:

Command	Explanation
KICK	Eject a client from the channel
MODE	Change the channel's modes such as defining a maximum number of users and/or the necessity to become invited prior to joining that channel
INVITE	Invite a client to an invite-only channel (mode +i)
TOPIC	Change the channel's topic

A channel operator is identified by the '@' symbol preceding their nickname whenever it is associated with a channel (i.e. replies to the NAMES, WHO and WHOIS commands) [Bri97].

6.5.4 The IRC Specification

The protocol as described herein is for use both with server to server and client to server connections. There are, however, more restrictions on client connections (which are considered to be untrustworthy) than on server connections.

For a complete list of available IRC commands, refer to the documentation of the various IRC clients available for your platform:

- UNIX: ircii

- Linux: sirc
- Win32: mIRC
- Macintosh: Ircle

6.5.5 IRC networks and known problems

When you connect to an IRC server you will be able to chat with anyone currently connected to the same IRC network. Several public IRC networks have evolved over the years with the smallest possible IRC network being a single server for IRC to which clients can connect.

The largest IRC networks are the so-called EFNet (the original IRC network, with often more than 32,000 users online), Undernet, IRCnet, DALnet, and NewNet to name a few.

6.5.5.1 Netsplit

IRC networks can suddenly become split (so-called **netsplits**), i.e. suddenly a large amount of users becomes divided from the others due to a broken Server-Server-Interconnection or a problem with a server itself. After a few minutes, the users that got disconnected may show up again. Netsplits can last from seconds to several days, depending on the underlying problem. If a server is shut down for a longer period, users must connect to a different server to re-establish communication.

If the IRC network becomes disjoint because of a split between two servers, the channels on each side are composed only of those clients which are connected to servers on the respective sides of the split, with a channel possibly ceasing to exist on one side of the split. When the split is healed, the connecting servers announce to each other who they think is in each channel and the modes and statusses of that channel. If the channel exists on both sides, the JOINs and MODEs are interpreted in an inclusive manner so that both sides of the new connection will agree about which clients are in the channel and what modes the channel has.

6.5.5.2 Lag

Another common issue is the so-called "**lag**", i.e. a noticable time that passes between dispatching a text message and its appearance on the receiver’s screen. Lags can be minimized by choosing a server physically close to the user. It can be measured by entering the commands /CTCP nickname PING or /CTCP #channel PING. These useful commands became necessary due to the quickly growing amount of IRC networks and users in the world and are called the

6.6 Client to Client Communication

Client-to-client communication [ZRM94] became necessary to overcome the limitations of the IRC server network described above and was developed after the original specification of the IRC standard [RFC1459].

6.6.1 The Client-To-Client Protocol (CTCP)

Written in 1994, the **CTCP** specification follows the original IRC RFC and it is the authoritative document for the client-to-client protocol [ZRM94].

It is meant to be used as a way to send structured data (such as graphics, voice and different font information) directly between users' clients, and, in a more specific case, place a query to a user's client and getting an answer.

Some CTCP command/reply-pairs are obligatory to all IRC clients while others might not be supported by all clients. The following commands are mandatory:

command	valid reply
FINGER	Returns the user's full name, and idle time
VERSION	The version and type of the client.
SOURCE	Where to obtain a copy of the client software that is used.
USERINFO	A string set by the user (never the client coder)
CLIENTINFO	Dynamic master index of what a client knows.
ERRMSG	Used when an error needs to be replied with.
PING	Used to measure the delay of the IRC network between clients.
TIME	Gets the local date and time from other clients.

If you want to know what CTCP commands a user's client will reply to, type in `/CTCP <nickname> CLIENTINFO` and what you get will be like this:

```
[Sunset CLIENTINFO reply]: VERSION CLIENTINFO USERINFO ERRMSG
FINGER TIME ACTION DCC UTC PING ECHO CAST128ED-CBC SED  :
Use CLIENTINFO <COMMAND> to get more specific information
```

Send one of these commands to a client and see what happens.

As you can see, an explanation of the complete scope of possible CTCP messages would be beyond the scope of this course unit. However, the most popular use for the CTCP protocol messages is the DCC (Direct Client connection) interaction between two clients which will be explained now:

6.6.2 The DCC (Direct Client Connection) Protocol

DCC allows users to have a secure connection while still being in an IRC protocol. The reason is that DCC uses direct point-to-point-TCP connections between the clients taking part. Packets can be sent directly, and there is no dependance on IRC-server links (or their current workload). In addition, since only the initial handshake for DCC connections is passed through the IRC network and ongoing traffic is passed through the client-to-client-connection, it is not possible for IRC-server operators to see DCC messages or observe the data that is being exchanged.

6.6.2.1 Setting up a Socket

The initial socket for a DCC connection is created by the side that initiates (offers) the connection. This is a TCP-socket bound to INADDR_ANY, listening for connections. This initiating client should send its details to the target client using the CTCP command DCC. This command takes the form:

```
"/CTCP DCC type argument address port"
```

where the arguments have the following meaning:

argument	explanation
`type`	the connection type, default: SOCK_STREAM
`argument`	the connection type dependant argument
`address`	the host address of the initiator as an integer.
`port`	the port on which the initiator listens to the inbound connection.

The address and port should be sent as ASCII representations of the decimal integer formed by converting the values to network byte order and treating them as an unsigned long and unsigned short, respectively.

6.6.2.2 Connection

The following DCC connection types are known in IRC:

- CHAT to carry a secure conversation - argument: the string chat
- SEND to send a file to the recipient - argument: the file name

For more information on DCC features, see [Rol94].

6.7 Automation and security issues

6.7.1 Clones

Clones are multiple clients from the same person. However, if it is obvious that someone is running multiple clients from different domains, they are also considered clones. Generally, cloning itself is not a hard problem - clones will only take up regular server connections. But clones are often being used to "flood" users or "take over" channels.

6.7.2 Network Flooding

There are two ways of **flooding** on IRC.

The first is CTCP flooding, which means that an "evil" user tries to get a regular user to flood the server with CTCP responses, tripping the server's flood protection, and therefore terminating the user with the message "Excess Flood". **Flood protection** scripts may prevent this from being very effective, but the real problem is, of course, the impact on the network bandwith and latency.

If 20 clients flood a user for 10 seconds, sending five 100 byte CTCP requests per second, that is 20 * 100 * 5 = 10kByte/s, or 100kByte of data total over the 10 seconds. Since a typical network is maintaining about 20000 users or more, this behaviour poses a real threat to normal communication.

The second way of flooding, which is not related to IRC (but is frequently the result of conflicts on IRC), is **ICMP flooding** (Internet Control Message Protocol). This is usually done from a reasonably fast link (2 Mbit/s or higher) and means flooding a user or server with ICMP packets (such as the IP tool "ping" uses). What results from this is called "**Denial Of Service**" and is unlawful.

6.7.3 Scripts

IRC Scripts are a collection of commands that a client will execute. They are meant to provide a user with additional commands to those that an ordinary client software provides. Like a batch-file, a script might execute several IRC commands in a row to - for example - connect to a specific server, change your nick and enter a specific channel. Another possible use for a script is to evaluate the user information given by the IRC server and then use it to trace the route that a signal will follow from your machine to a user and determine its run-time. Scripts can be downloaded for almost any client software on any platform from the Internet. However, a large number of available scripts have been modified so that they - besides their otherwise good behaviour - can impose great damage to a client's system. It is a good advice not to use a script that is not self-written. Scripts can be as dangerous as "**trojan horses**" since they behave like executables that can for example send information about the contents of your hard-drives to a specific user (probably the one who offered the script) or even delete it. Other scripts might give a remote user complete control

over your desktop and IRC software so that you will suddenly harass your friend without typing a word or find your computer shutting down for no obvious reason.

6.7.4 IRC-Bots

"**Bot**" is the short form of robot. A robot generally refers to any automated program or client that does not have a person sitting behind it.

Bots are another form of automatisation of IRC. They are mainly used by IRC or channel operators to keep a channel established and moderated while the operator himself is not sitting at his computer watching the users. Bots can decide whether a user that enters a channel is to be treated friendly, neutral or as a threat. This is usually done by comparing the new user's IP adress and **nickname** to corresponding database entries and therefore deciding on the base of the user's past behaviour.

- Treating a user friendly could mean greeting him, giving him operator-status and/or providing him with statistical information about the channel.
- Treating him as neutral could also mean greeting him but otherwise leaving him alone.

When a user is considered a threat, a bot that is given operator-status will probably add this user's IP adress to the channel's list of banned users and eject this user from the channel, maybe with a message about the reasons for this. A bot can also inform IRC operators about the repeated attempt of an evil user to enter certain channels or perform harassments to a channel.

However, there are a couple of reasons why bots are frequently considered to be a problem themselves. First, they take up resources on IRC that could be used for regular client connections. The primary reason, though, is because bots (and their underlying scripts) are not only used to protect and control but also to flood and harass users. Should an evil bot ever be given operator status, it is most likely that the bot will kick and ban every user on that channel and therefore block the channel. Such a "blocked" channel can only be unlocked by an IRC operator ejecting that bot from the server and re-opening the channel.

6.8 Useful abbreviations

Here are some abbreviations that are typical for chatting on IRC and their meaning:

phrase	meaning
brb	I will be right back.
bbiaf	I will be back in a flash.
bbl	I will be back later
ttfn	ta ta for now. [Bye, bye]
np	no problem
imho	in my humble opinion
lol	lots of laughter
j/k	just kidding
re	hi, i'm back!
wb	Welcome back!
rtfm	Please consult your operation manual to find an answer
rotfl	rolling on the floor laughing

6.9 Summary

This chapter introduced briefly the history of IRC development. To understand the real-time multi-user feature of a IRC network the basic communication over point-to-multipoint-connections are described. Protocol primitives as well as protocol services are introduced. Using IRC clients is described as well as possible leaks in the IRC networks for security attacks.

7 Email

7.1 Goal of the Chapter

The goal of this course unit is to understand the structure of email messages and the way email messages are transported. The interaction between an email client and an email server as well as the interaction between email servers are explored.

Electronic mail (or **email** for short) is a widely used, easy, and fast communication service provided on many different kinds of computers. It is mostly used to transfer short notes of text messages form one user to one or more recipients, but also multimedia content is supported.

7.2 From paper mail to electronic mail

To write a letter to a friend, one needs some information and a few tools. The information is the address of the friend in form of name, street, zip code, and town name. As tools in most cases a sheet of paper, an envelope, and a pencil (or a typewriter) are sufficient. On the plain paper, beside the actual content, date and subject are noted. The envelope is labeled with the address (name, street, and town name) of the addressee and with the same information about the sender. The letter is put into the envelope, prepaid with a stamp, and deposed in the next mail box.

Its the same as with email - except for the stamp.

Like regular mail, electronic mail is a point-to-point communication. One person originates the communication by composing a message and then sending it to one or more recipients. The message is delivered to the mailbox of each recipient, where it can be read.

Basically, electronic mail involves transferring a computer file from one user account to another user account. In the following we will use this paradigm of paper mail to explain the way electronic mail works. In Fig. 7.2-1 the two kinds of mail can be seen side by side.

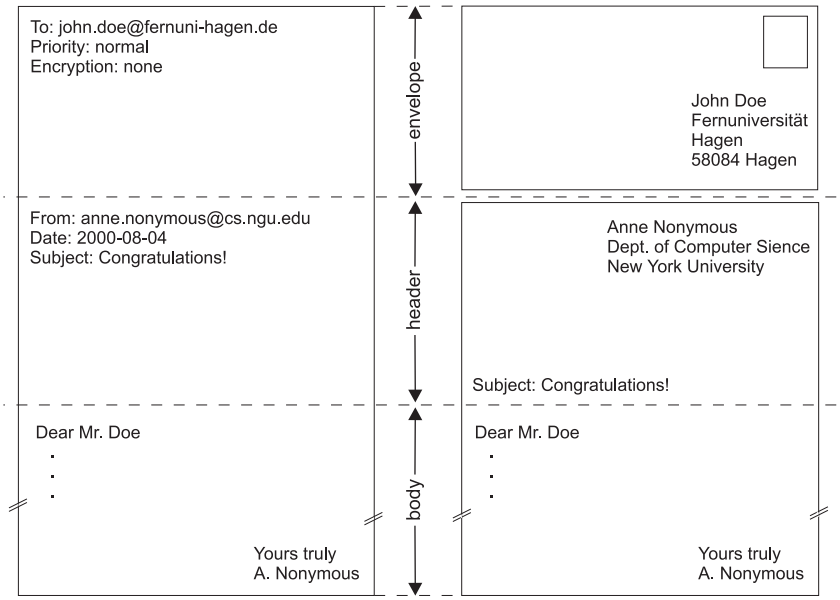


Fig. 7.2-1: Electronic mail and paper mail

The equivalence between the combination of the labeled envelope and the head of the letter on one side and the header of the email on the other side can clearly be seen. It is analogous to the matching of postal and email body.

7.3 Structure of email messages

The structure of email messages was defined some years ago, the RFC 822 ("**Request for Comments**", the de facto Internet standards) [RFC822] shows the definition in detail and is used as basic literature in this course unit.

The email message has three basic parts: The message envelope, the header, and the message body. The message header lists information about the sender, the recipient, the posting date, the subject of the message, etc. The message body contains the actual message being sent.

The message header consists of several fields. Take a look at the following example of an email message (all data of the email is shown; an email client will suppress some of them while displaying, some more fields can be displayed in other situations):

```
Received: from SpoolDir by LGKS1 (Mercury 1.31); 10 Jul 100 16:22:48 MET
Return-path: <anne.nonymous@cs.nyu.edu>
Received: from elm.fernuni-hagen.de by lgks1.fernuni-hagen.de (Mercury 1.31)
11 Jul 100 01:41:00 MET
Received: from slinky.cs.nyu.edu by elm.fernuni-hagen.de via Internet
with ESMTP; Tue, 11 Jul 2000 01:37:50 +0200
Received: from localhost (anne.nonymous@localhost)
by slinky.cs.nyu.edu (8.9.1/8.9.1) with ESMTP id TAA27290
for <Thomas.Demuth@FernUni-Hagen.de>;
Mon, 10 Jul 2000 19:37:48 -0400 (EDT)
From: Anne Nonymous <anne.nonymous@cs.nyu.edu>
To: thomas.demuth@FernUni-Hagen.de
```

```
Date: Mon, 10 Jul 2000 16:21:33 +0200
MIME-Version: 1.0
Content-type: text/plain; charset=ISO-8859-1
Content-transfer-encoding: Quoted-printable
Subject: Thanks
Priority: normal
X-mailer: Pegasus Mail for Win32 (v3.12a)
X-PMFLAGS: 34078848 0 1 Y0FC54.CNM

Dear Mr. Demuth,

thank you for sending the research report.

Regards,
Anne Nonymous
```

The header of this message consists of the fields from the first "*Received*" to "*X-PMFLAGS*", the body starts with "Dear Mr. Demuth,". Header and body must be separated by a single line.

Headers also pick up information as a message is sent to its recipient. The email software adds hidden text that itemizes the message's size, origination location, and other miscellaneous information. By the time the message arrives at its destination, it will have a whole paragraph of additional text that recounts the routers the message passed through.

7.3.1 The email header

The header consists of "fields". Each field has a name (and therefore a meaning), a value, and is terminated by a newline/line feed. The field name must be composed of printable ASCII characters, the body may be composed of printable ASCII characters including newline/line feed.

7.3.1.1 The header fields

The most important and often used header fields are explained in the following. Details can be found in [RFC822]. The minimum required fields are "*Date*", "*From*", and "*Bcc*" or "*To*"; all others are optional.

"**To**": This field contains the mailing addresses to which the message is to be sent. If more than one address is listed, they must be separated by commas.

"**Subject**": The contents of the "*Subject*" field should be a piece of text that says what the message is about. The reason "*Subject*" fields are useful is that most mail-reading programs can provide a summary of messages, listing the subject of each message but not its text.

"**Cc**": This field ("**Carbon Copy**") contains additional mailing addresses to send the message to. This works like "*To*"' except that these readers should not regard the message as directed to them.

"**Bcc**": This field ("**Blind Carbon Copy**") contains additional mailing addresses to send the message to, which should not appear in the header of the message actually sent. Copies sent this way are called blind carbon copies.

"**From**": This field is used to specify the sender, when the account he is using to send the mail is not his own. The contents of the "From" field should be a valid mailing address, since replies will normally go there.

"**Received**": The names of the sending and receiving hosts and time-of-receipt may be specified in this field. The "*via*" parameter may be used to indicate what physical mechanism the message was sent over and the "*with*" parameter may be used to indicate the mail- or connection-level protocol that was used, such as the SMTP mail protocol, or X.25 transport protocol. Some transport services queue mail; the internal message identifier that is assigned to the message may be noted, using the "*id*" parameter. When the sending host uses a destination address specification that the receiving host reinterprets, by expansion or transformation, the receiving host may wish to record the original specification, using the "*for*" parameter. For example, when a copy of a mail is sent to a member of a distribution list, this parameter may be used to record the original address that was used to specify the list.

"**Reply-to**": This field directs replies to a different address. Most mail clients automatically send replies to the "*Reply-to*" address in preference to the "*From*" address.

"**In-reply-to**": This field contains a piece of text describing a message for replying to. Some mail systems can use this information to correlate related pieces of mail.

"**References**" (**Message Threading**): This field is usually maintained and added automatically. "*References*" describes the thread (chain, series) of messages that "came before" the actual one. Each message on the Internet has a unique identification number that is kept in the "*Message-ID*" field. "*References*" is a list of the message-ids from previous messages in this thread. When a new message is composed and sent, the mail server adds the "*Message-ID*" field. When someone replies to that message, the user's email program will create a "*References*" field and copy the original message-id into it. If someone else replies to that reply, this "*References*" field gets the previous references, plus the previous message-id. And so on.

Example 7.3-1:
An example should make this easier to see. Let's look at a thread with three messages. user-A sends a message to user-B. This original message header might have:

```
From: user-A
To: user-B
Subject: Something very important
Message-ID: <123@fernuni-hagen.de>


Now user-B replies to user-A. The reply header looks like:
```

```
From: user-B
To: user-A
Subject: Re: Something very important
References: <123@fernuni-hagen.de>
Message-ID: <246@fernuni-hagen.de>


The third message in this thread is from user-A to user-B,
with a copy to user-C. user-A also decides to change the subject:


From: user-A
To: user-B
Cc: user-C
Subject: Another thing (was: Re: Something very important)
References: <123@fernuni-hagen.de> <246@fernuni-hagen.de>
Message-ID: <789@fernuni-hagen.de>
```

The "*References*" field will keep getting longer as this thread goes on. It lets one find all the messages in the thread by searching for the message-ids.

"**X-**" (**Own Header Fields**): The RFC 822 transport standard carefully specifies what header fields are legal in mail messages. Any field name starting with the characters X- (the letter X followed by a dash) won't be adopted by a new message header standard. According to RFC 822, "any field which is defined in a document published as a formal extension to this specification will have names beginning with the string 'X-'." That means the standard cannot anticipate every useful header, it allows for user-defined fields as extensions to the standard. The convention is to begin a user-defined field with "X-", because the standard guarantees that no official extension will use a header with those names. So the X-headers can be seen as a free "playground" for anyone's use.

7.3.1.2 Structure of an email address

Email messages employ a two-part addressing scheme. A valid internet email address is:

thomas.demuth@fernuni-hagen.de

Email has its own adressing system, called *domain name addressing* (corresponding to the *domain name system* of the Internet (Section 3.3.5)) as the electronic equivalent of postal addresses. Like these postal addressses, email addresses go from specific to general, in order to route the message to the right computer and mailbox (or person). Generally, email addresses have a *username*, one or more *location identifiers*, and a *domain*. An "@" ("at sign") symbol separates the user name from locations and domain.

Explanation of the individual parts:

The first part ("*username*") specifies the user to whom the email is to be delivered. The user is identified by the username attached to their computer account. The username for an individual is assigned when their computer account is created, and typically cannot be changed by the user. There is no standard scheme for usernames that applies universally. There are some restrictions on usernames, though. The name must not contain certain reserved characters like the "at sign" (@), the exclamation mark (!), and the percentage sign (%), since these characters have special meaning to email software.
The second part describes the host computer where the user's account resides (host address). The address of the host machine can take several forms. Here we discuss only the most common form, the one employed on the Internet. The email address of a host computer is specified in at least two parts, separated by a dot. The last component of the address is called the domain of the host computer. In addition to a domain name, a valid host address must contain at least one subdomain name. The subdomain further specifies the institution that owns the host machine. The subdomain may refer to a specific part of the institution's computer network, or may refer to a specific machine. For example, the host address **"ks.fernuni-hagen.de"** describes the academical institution named "**fernuni-hagen**" and a machine at that site named "**ks**" ("Kommunikationssysteme").
Analysing the email address above from the right to the left (or from general to specific), one can see the country, the email shall be sent to. "**.de**" represents Germany, and in Germany the email shall be forwarded to the FernUniversität in Hagen ("**.fernuni-hagen.de**").
The addressed user there is Thomas Demuth ("**thomas.demuth**"). In many large organisations, users are given their addresses in this form: `<first name>.<lastname>` to avoid collisions between users with similar names.

7.3.2 The email body and MIME

The mail body is completely free of format, one can compose an email in any desired way. Corresponding to the standard, the body is separated by a blank line from the header. In the early days of computers email was not "8 bit" clean; users were able only to write emails in straight (7 bit) ASCII text. That means that only textual email is supported by RFC 822 email; nontextual byte content can therefore be clipped to seven bits.

Therefore, in 1992 the MIME standard was developed. **MIME** stands for "**Multipurpose Internet Mail Extensions**" and enables email to transport multimedia data, still being conform to RFC 822.

Before MIME, users converted their binary files like spreadsheets or other office documents with the (UNIX) commands "**uuencode**" and "**uudecode**". These commands are used to transmit binary files over transmission mediums that do not support other than simple ASCII data. Uuencode reads the input and writes an encoded version. The encoding uses only printable ASCII characters and includes the mode of the file and the operand name for use by uudecode.

But, since the introduction of MIME no user has to care about this any more while writing emails (As a consequence of this, one can attach any binary file to an email.).

The following RFC's define Multipurpose Internet Mail Extensions, replacing the RFC 1521 which is not valid anymore:

- RFC 2045: MIME Part One: Format of Internet Message Bodies
- RFC 2046: MIME Part Two: Media Types
- RFC 2047: MIME Part Three: Message Header Extensions for Non-ASCII Text
- RFC 2048: MIME Part Four: Registration Procedures
- RFC 2049: MIME Part Five: Conformance Criteria and Examples

For references see [RFC2045], [RFC2046], [RFC2047], [RFC2048], and [RFC2049].

These standards (or definitions) redefine the simple structure of messages in RFC 822 format to allow for

1. textual message bodies in character sets other than US-ASCII,
2. an extensible set of different formats for non-textual message bodies,
3. multi-part message bodies, and
4. textual header information in character sets other than US-ASCII.

It is important to understand, that MIME does not change the structure of an email. In fact it encodes the arbitrary data in ASCII to be transmitted in a standard email message. Therefore, older email programs (**email clients**) can handle MIME messages, too.

To accomodate arbitrary data types and representations, each MIME message includes information that tells the recipient the type of the data and the encoding used. MIME information resides in the RFC 822 mail header - the MIME header files specify the version of MIME used, the type of the data being sent, and the encoding used to convert the data to ASCII.

Above, the possible encoding of binary messages or attachments with uuencode/base64 has been mentioned. The advantage of MIME is a quite more comfortable handling for the user. In addition to the defintion of encoding and formatting, MIME does also determine the methods modern email clients can detect and handle such messages autonomously, i.e. starting the appropriate application. For this reason, MIME messages contain *meta information*, defining the player for a sound file or a program to work with graphics.

MIME is a flexible way to embedd multimedia attachments in internet messages. Data can be nested and instructions to a front end program can be embedded to output the information in dependence of available resources. In this way, a memo can be read out or displayed alternatively as text, if no sound device is available. Furthermore, links to external data can be integrated in MIME messages.

The following listing illustrates a MIME message that contains a picture in standard GIF represenation. The GIF image has been converted to a 7-bit ASCII representation using the *base64* encoding (for explanation of *base64*, see below).

```
From: Anne Nonymous <anne.nonymous@cs.nyu.edu>
To: thomas.demuth@FernUni-Hagen.de
Date: Mon, 10 Jul 2000 16:21:33 +0200
MIME-Version: 1.0
Content-type: text/plain; charset=ISO-8859-1
Content-transfer-encoding: Quoted-printable
Subject: Congratulations
Priority: normal
X-mailer: Pegasus Mail for Win32 (v3.12a)
X-PMFLAGS: 34078848 0 1 Y0FC54.CNM
Content-type: Image/GIF; name="medal_of_honour.gif"
Content-disposition: attachment; filename="medal_of_honour.gif"
Content-transfer-encoding: BASE64



R0lGODdhWAK+AKIAAP///9vb27a2tpKSkm1tbUlJSSQkJAAAACwAAAAAWAK+AEAD/wi63P4w
ykmrvTjrzbv/YCiOZGmeaKo6B7G+8BkchtI2Rh3vfJf3wGCpoBMaj47BweaSDBiHaOO2MDxt
hqj2SVgClBMaoLssLGcDwSHAPghsEbVCPZgFFnIW6VBodB9/ClkAWmsLXohSfHheDHk2V1MG
bF1qdwp2iG8/bmFvbUuGhGIEOmAaREiqqxOpDqlmUUUABn1TWm+ljo0HaY0zOLYRVg9KwoRv
CgNZTW7AhwNmcDaNh8kXp1RO1Vm9h7OHAcZfoZd/1QAzVOiCxwynDWpS4YQPP9MjzH7s8Ay6
tE1o1RC1QFqEagTfgZsTytazBWjSeckTy8VDfCQCsYLirv8ToQIzatV4dqfLMVGXNkb418AV
AyveIElwM4jWvDUGAk4rYCtmAQKxttSDxqKWGR2i7o0iojPdiEA1ZUhgI4ANnpTGCFwbRaOW
.
.
.
```

The MIME field "*MIME-Version 1.0*" declares that the message was composed using version 1.0 of the MIME protocol. MIME is downwards compatible, that means, that every MIME capable email client can likewise handle ASCII messages.

Two more fields (not used in the example email) are "*Content-Id*", acting as a non-ambiguous identifier (and functionally identical to the standard header "*Message-ID*" (see above)) and "*Content-Description:*", a text describing the content, understandable for a human reader to inform him before the content is decoded. Thus the receiver of the message can decide if it is reasonable for him to decode the message.

The "**Content-Transfer-Encoding**" header declares in the example, that base64 encoding was used to convert the image to ASCII. There are five different encoding schemes available (and another one for further extensions):

"*7bit*": ASCII text, using 7 bit characters, to directly transfer the content without further encoding. This format is limited to 1.000 characters per line.

"*8bit*": ASCII text, using 8 bit per character (values from 0 to 255) and thus violating the encoding scheme of RFC 822. Nevertheless it is used, causing the expected consequences.

"*binary*": Like "*7bit*" without the limitation to 1.000 characters per line, but with the consequences of the "*7bit*" mode.

"**base64**": This encoding is also called "**ASCII armor**". Groups of 24 bits are encoded in four units of six bit. Each of these units is then encoded als regular ASCII character ("A" for 0, "B" for 1, and so on). Carriage returns and line feeds are ignored.

"*quoted-printable*": This encoding only affects bytes exceeding the limit of the seven-bit-ASCII-code. They are transformed into an equal sign, followed by the hex code of the character (for example: "=E9" for an "ä"). This type of encoding is useful, if special, national characters are used, or if it is not known, if the receiver of a message can handle base64-encoded attachments.

The "*Content-Type*" declaration specifies the type of information. Seven types are defined, each with one or more subtypes; types and subtypes are separated with a slash ("/").

The purpose of the Content-Type field is to describe the data contained in the body in such a manner that the receiving user agent can pick an appropriate agent or mechanism to present the data to the user, or otherwise deal with the data in an appropriate manner. The value in this field is called "*media type"*.

The most often used types and subtypes are listed in Table 7.3-1

Tab. 7.3-1: *MIME types and subtype (selection)*

Type	Subtype	Description
text	plain	unformatted ASCII text
	richtext	basically formatted ASCII text
image	gif	GIF picture
	jpeg	JPEG/JPG picture
audio	basic	sound data
video	mpeg	MPEG movie
application	octet-stream	non interpretable byte sequence
	postscript	printable postscript document
	rfc822	MIME message in RFC 822 format
message	partial	message has been split before transfer
	external-body	link to external ressource
multipart	alternative	same message in different formats
	parallel	the parts of the message have to be displayed at once
	digest	each part is a RFC 822 compliant message
	mixed	several parts of independent encoding

The understanding of MIME is not only important to realise the structure and interpretation of email messages. The Hypertext Transport Protocol (HTTP), used to transfer Web pages, also uses MIME for the transport of the pages.

It is the task of the sender's email client to interprete the data to send and to select the right content and sub types. In some cases the sender has to interfere and select the types manually. In the example the *Content-Type* declaration specifies that the data is a GIF image, *image* is the content type, and *gif* is the subtype. To view the image, a receiver's email client must first convert from base64 encoding back to binary, and then run an application that displays a GIF image on the user's screen.

In the example above an additional MIME header field can be seen: *Content-disposition*. This field gives more information about the following MIME element. Like in the case of the *Content-type* field further parameters can be added to the line. In the example one can see the name of the image and that it is an attached element. For more information see [RFC1806].

7.3.2.1 Multipart MIME messages

One very important type mentioned above is *multipart*. This type allows messages consisting of several parts; these parts are clearly separated from each other. With the multipart content type, email gets considerable flexibility.

The subtype *mixed* allows a single message to contain multiple, independent sub-messages, each one having its own content type and encoding. Mixed messages make it possible to include different multimedia data in one message.

The subtype *alternative* allows a single message to include multiple representations of the same data. This approach is usefull for sending the same message in different formats (audio, video, or text). The email client of the receiver should be able to decide the part of message which will be used, with respect to the hardware capabilities of the computer.

Subtype *parallel* permits a single message to include subparts that should be used together (like separate audio and video data that must be played synchronously).

Subtype *digest* allows a single message to contain a set of other messages that are RFC 822 compliant. This feature is very useful for receiving the daily or weekly email of a mailing list at once.

The following example shows a multipart mixed MIME message:

```
Date: Fri, 21 Jul 2000 14:02:28 +0200
From: Anne Nonymous <anne.nonymous@cs.nyu.edu>
Organization: MAD Dept.- Department for Mathematics and Depressions
X-Mailer: Mozilla 4.7 [en] (Win98; U)
MIME-Version: 1.0
To: thomas.demuth@fernuni-hagen.de
Subject: Betreff
Content-Type: multipart/mixed;
boundary="------------675454AB4619496261BDECC9"

This is a multi-part message in MIME format.
--------------675454AB4619496261BDECC9
Content-Type: text/plain; charset=us-ascii
```

```
Content-Transfer-Encoding: 7bit

This is plain text (see Content-Type)
--------------675454AB4619496261BDECC9
Content-Type: image/jpeg;
name="et-online.jpg"
Content-Transfer-Encoding: base64
Content-Disposition: inline;
filename="et-online.jpg"

/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAsICAoIBwsKCQoNDAsNERwSEQ8PESIZGhQcKSQr
.
.
Y9FFFbCCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAC
iiigAooooAKKKKAHdqQ8dDRRSA//2Q==
--------------675454AB4619496261BDECC9
Content-Type: application/x-unknown-content-type-Winamp.File;
name="Elvis Presley - Jailhouse Rock.mp3"
Content-Transfer-Encoding: base64
Content-Disposition: inline;
filename="Elvis Presley - Jailhouse Rock.mp3"

//uQRAAAAAAASwAAAAAAAAlgAAAAAAABLAAAAAAAACWAAAAA//////////////////////
.
.
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAER
--------------675454AB4619496261BDECC9--
```

This is an email sent with an other email client as the ones before. One can see this by looking at the *X-Mailer* header, now presenting Netscape (which identifies itself as Mozilla) as the email program, which was used to compose the email.

The keyword `boundary=` following the multipart content type declaration in the header fields defines the string used to separate parts of the message. In the example the string has been selected by the email client in a such a way that the string is no fragment of one of the parts of the MIME message.

One can see two parts: one gif, presenting the ET-Online logo, and a song from a famous rock star.

7.4 Email clients and transport of email

Remember the analogy between paper mail end email depicted in Chapter 7.2. This picture goes further. The transport of email can be compared with traditional mail, too.

A letter is written at home or at the office, put into the post, transported via car, plane or train from town to town (or country to country) and then placed into the mailbox of the intended receiver.

In electronic mail, these components can be identified, too. One can see two subsystems, a **User Agent** (**UA**), offering the capabilities of writing and reading emails, and a number of **Message Transfer Agents** (**MTA**), transporting the email to the addressee.

7.4.1 Email Clients

User agents are also called email clients (in the following we will use this term because it is more familiar). These are software tools, available for every operating system. Examples are Netscape Composer, Pegasus Mail, or pine. Email clients allow users to do several actions and support different functions:

- Composition of electronic mails with text, graphics, multimedia and so on.
- Administration of received messages. Users can move emails into named folders, delete or print messages. This function is also called disposition.
- A list of received or recently sent messages are displayed to the user.
- Email clients report new email to the user in form of popping up of an alert window and/or playing a special sound. User can read and react immediately to incoming emails.
- Many email clients support the digital signing and encryption of emails to ensure integrity and confidentiality of emails. !!!
- Email clients are also responsible for the first step of the transfer, the transmission to the MTA.
- Email clients allow the user to give messages different priorities, to forward received mails, or to send receipts for emails to the sender.
- Additionally, convenient clients can be configured to send answers automatically (auto-reply), if a user is absent (on holidays, etc.).

The functions composition, disposition, displaying, reporting, and transfering are characteristical for email clients.

7.4.2 Simple Mail Tranfer Protocol

Email clients run on local computers (PCs, MACs) normally. After composition, a client sends the message to the next message transfer agent (MTA). MTAs are programs, which run on servers (mostly UNIX) and accept emails from authorised clients or forward emails from other authorised MTAs (**relaying**).

Fig. 7.4-1 shows a sample architecture: Several UAs (email clients), e.g. computers of employees in a company, can connect to a central MTA, also called **mail server**. This MTA collects emails and regularly sends them to corresponding MTA.

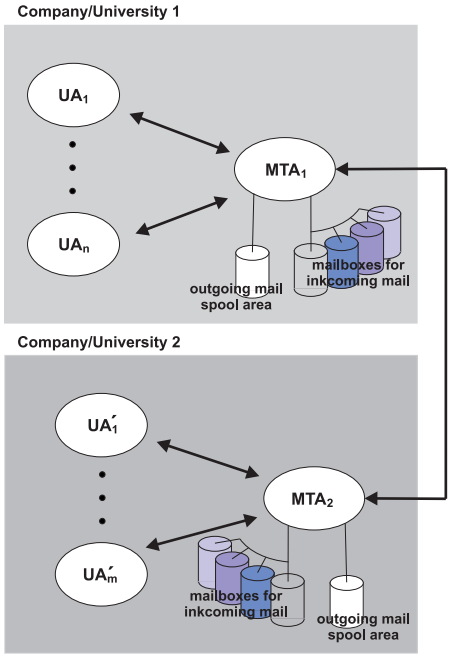


Fig. 7.4-1: Sample mail scenario

An MTA waits for emails from the local clients (or user agents) and queues them. In most cases, direct transfer from the start MTA to the destination MTA is not possible. As mentioned, many MTAs only accept emails from other authorised MTAs or email clients. In this case, emails have to be relayed (or forwarded) over one or more MTAs.

To promote understanding, in the following a direct connection between a source and a destination MTA is considered. MTA 1 opens a TCP connection on port 25 (the standard port for email transport, see Chapter 7.4) to the destination MTA. Once a connection is established, the two MTAs communicate via **SMTP** (**Simple Mail Transfer Protocol**). On the machines a so called daemon (server program) runs, waiting for communication. The daemon's task is to receive email, forward email to other computers, or distribute messages to mailboxes of users. If a message can not be forwarded, an error report is generated and delivered back to the sender.

SMTP uses straight 7 bit ASCII to communicate. After contacting the destination MTA the source MTA acts as a client, waiting for the server's response. The server, if able and willing, sends a line back, identifying itself and describing its abilities. Then, the client sends the mails successively to the server, every time telling the server the identifiers of sender and receiver. SMTP is well documented in RFC 821 [RFC821]. Although SMTP rigidly defines the command format, humans can easily read a transcript of interactions between client and server.

In the following the transcript of a transfer of a message to an MTA via SMTP is explained.

```
C: demuth@karlsberg:~ > telnet bonsai.fernuni-hagen.de 25
C: Trying 132.176.114.21...
C: Connected to bonsai.fernuni-hagen.de.
C: Escape character is '^]'.
```

```
S: 220 bonsai.fernuni-hagen.de ESMTP Sendmail 8.8.8+Sun/8.8.8; Tue, 25 Jul 2
C: HELO karlsberg.fernuni-hagen.de
S: 250 bonsai.fernuni-hagen.de Hello Karlsberg.fernuni-hagen.de [132.176.12.
C: MAIL FROM: demuth@karlsberg.fernuni-hagen.de
S: 250 demuth@karlsberg.fernuni-hagen.de... Sender ok
C: RCPT TO: kaderali@lgks1.fernuni-hagen.de
S: 250 demuth@lgks1.fernuni-hagen.de... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: From: demuth@karlsberg.fernuni-hagen.de
C: To: kaderali@lgks1.fernuni-hagen.de
C: Subject: My Research
C: Dear Prof. Kaderali,
C: thank you for the support in my research.
C: T. Demuth
C: .
S: 250 NAA11965 Message accepted for delivery
C: QUIT
S: 221 bonsai.fernuni-hagen.de closing connection
C: Connection closed by foreign host.

Received: from SpoolDir by LGKS1 (Mercury 1.31); 25 Jul 100 13:23:08 MET
Return-path: <demuth@karlsberg.fernuni-hagen.de>
Received: from elm.fernuni-hagen.de by lgks1.fernuni-hagen.de (Mercury 1.31)
with ESMTP;
25 Jul 100 13:23:03 MET
Received: from bonsai.fernuni-hagen.de by elm.fernuni-hagen.de
via local-channel with ESMTP; Tue, 25 Jul 2000 13:19:44 +0200
Received: from karlsberg.fernuni-hagen.de (Karlsberg.fernuni-hagen.de [132.1
by bonsai.fernuni-hagen.de (8.8.8+Sun/8.8.8) with SMTP id NAA11965
for kaderali@lgks1.fernuni-hagen.de;
Tue, 25 Jul 2000 13:18:30 +0200 (MET DST)
Date: Tue, 25 Jul 2000 13:18:30 +0200 (MET DST)
From: demuth@karlsberg.fernuni-hagen.de
Message-Id: <200007251118.NAA11965@bonsai.fernuni-hagen.de>
X-PMFLAGS: 33554560 0 1 Y06467.CNM

From: demuth@karlsberg.fernuni-hagen.de
To: kaderali@lgks1.fernuni-hagen.de
Subject: My Research
Dear Prof. Kaderali,
thank you for the support in my research.
T. Demuth
```

The first part is the dialog between client (lines marked with "C:") and the server ("S:"). Each server reaction starts with a reply code, signaling the successful transction or an error. The server responds to the connection to port 25 with the code "220" ("Service Ready") and the type of mailer (SMTP/ESMTP, see below), signaling, that he is ready to receive mails. The first client command is "HELO" with the name of the contacting computer as parameter.

Step by step the client now transmits the identifier of the sender ("MAIL FROM"), of the receiver ("RCPT TO") and the actual message, consisting of header and body, as described in Chapter 7.3. If the message has more than one receiver, the RCPT field would be used several times, each time with another receiver's address. Therefore only the addresses, but not the complete message has to be repeatedly transfered.

With this technique, a layer is wrapped around body and header of an email. This layer is called **envelope** and is only used for the transport from one MTA to the next. The end of a message is described by a single dot "." in a line. After the reply code "250", signaling the acceptance of the email by the server, the client can send the next message. In the example, it completes the communication with the command "QUIT".

The second part of the example is the email as it is delievered to the receiver. Here all header are displayed (in reality most email clients suppress header fields to avoid confusion of the user). The "*Received*" header fields describe the route the email has taken over different MTAs (over "bonsai" and "elm" to "lgks1"). Further, the unique message identifier, given at the first MTA can be identified in the email the client has received.

Table 7.4-1 shows a list of standard reply codes.

Tab. 7.4-1: *SMTP reply codes*

211	System status, or system help reply
214	Help message
220	(domain) Service ready
221	(domain) Service closing transmission channel
250	Requested mail action okay, completed
251	User not local; will forward to (forward-path)
354	Start mail input; end with (CRLF).(CRLF)
421	(domain) Service not available,
450	Requested mail action not taken: mailbox unavailable
451	Requested action aborted: local error in processing
452	Requested action not taken: insufficient system storage
500	Syntax error, command unrecognised
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command parameter not implemented
550	Requested action not taken: mailbox unavailable
551	User not local; please try "forward-path"
552	Requested mail action aborted: exceeded storage allocation
553	Requested action not taken: mailbox name not allowed
554	Transaction failed

If a user does not have a SMTP client application, or for the purposes of testing and troubleshooting, it is possible to use a Telnet client application to contact an SMTP server and send mail messages. This is done manually by telnetting to port 25 and entering the commands manualy.

SMTP was defined in 1982. Since then, communication via email has extended in quantity, size, and frequency. To handle these requirements and limitations of some implementations (restrictions of MTAs to handle emails greater than 64 KB, etc.), the standard has been enhanced, resulting in ESMTP (RFC 1425, "service extensions to SMTP") [RFC1425]. ESMTP's most important feature is the negotiation between MTAs about the length of transported emails. Therefore redundant transport can be avoided.

7.4.3 Relaying of email

In reality, emails normally pass more MTAs than the source and the destination MTA. SMTP offers a mechanism called "*relaying*". Each SMTP server is able to relay emails (if it is not disabled intentionally). The receiver MTA receives mail to be relayed to another MTA. the receiver MTA may accept or deny the task of relaying the mail in the same way it accepts or rejects mail for a local user. The receiver MTA adds its own identifier from to the beginning of the reverse path (see header fields) to enable an easy way to reply to a message. The receiving MTA then becomes a sending MTA, establishes a transmission channel to the next MTA and sends the email.

This mechanism is called **store-and-forward** because of its behaviour. While being forwarded, the header of the email is extended with entries in the received field. A copy of this field is added by each transport service that relays the message. The information in the field can be quite useful for tracing transport problems, e. g. if an intended receiver is not known at the destination MTA. Furthermore it can be used to respond to an email: In this case the MTA can use the already known route in the received field(s).

7.5 Accessing mailboxes

Normally, not every single user computer runs its own SMTP daemon. Firstly, this is not neccessary for economical reasons. Secondly, personal computers are not online all the time. Institutions like universities, companies or internet service providers offer access to an MTA to their users. Up to now, the mechanism of transfering emails from a user agent to the first MTA has been handled in an abstract way.

In fact there are several protocols, with which the two instances can handle an email access or delivery. These protocols are independent of the underlying kind of connections. It is not relevant, if the user agent is connected via a dial up connection or possesses a leased or dedicated line. Today two main protocols, the more simple *POP3* and the advanced *IMAP* are generally used.

7.5.1 POP3

On certain types of smaller nodes in the Internet it is often impractical to maintain a **message transport system** (**MTS**). For example, a workstation may not have sufficient resources (CPU cycles, disk space) in order to permit an SMTP server and associated local mail delivery system to be kept resident and continuously running. Similarly, it may be expensive (or impossible) to keep a personal computer interconnected to an IP-style network for a duration of time (the node is lacking the resource known as "connectivity").

Despite this, it is often very useful to be able to manage mail on these smaller nodes, and they often support a user agent (UA) to aid in the task of mail handling. To solve this problem, a node which can support an MTS (Message Transfer Service) entity offers a maildrop service to these less endowed nodes. The **Post Office Protocol - Version 3** (**POP3**) is intended to permit a workstation to dynamically access a maildrop on a server host in a useful fashion. Usually, this means that the POP3 protocol is used to allow a workstation to retrieve mail that the server is holding for it. POP3 is not intended to provide extensive manipulation operations of mail on the server; normally, mail is downloaded and then deleted.

The POP server provides an off-line mail system, whereby a client, using POP client software, can remotely access a mail server to retrieve electronic mail messages. The client can either download the mail messages and immediately delete the messages from the server, or download the messages and leave the messages resident on the POP server. After the mail is downloaded to the client machine, all mail processing is local to the client machine. The POP server allows access to a user’s mailbox by only one client at a time.

The Post Office Protocol, version 3 (POP3) is used to pick up email across a network. Not all computer systems that use email are connected to the Internet 24 hours a day, 7 days a week. Some users dial into a service provider on an "as needed" basis, and others may be connected to a LAN with a permanent connection but may not always be powered on. Other systems may simply not have the available resources to run a full email server, might be shielded from direct connection to the Internet by a firewall security system, or it may be against the organisation policy to have mail delivered directly to user systems. In cases such as these the email addressed to the users on these systems is sent to a central email system where it is held for the user until they can pick it up. POP3 allows a user to log onto an email post office system across the network, validates the user by ID and password, allows mail to be downloaded, and optionally allows the user to delete the mail from the server.

POP3 is defined in RFC 1939 [RFC1939]. Under TCP/IP the POP3 protocol is implemented as an application layer protocol that uses Transmission Control Protocol (TCP) (Section 4.4) to establish a reliable connection between the two systems. TCP uses a pair of ports numbers, one for source and one for destination, to identify each communications link. By default a POP3 client application will contact the remote server using TCP/IP application port number 110 as the destination port, and

will select at random a port from the dynamic or private range for the source port number. The remote server will normally respond by contacting the client system using the private port number as the destination and port number 110 as the source.

Commands in POP3 are a single keyword, possibly followed by one or more arguments. Keywords and arguments consist of text characters. Keywords and arguments are each separated by a single SPACE character. Keywords are three or four characters long. Each argument may be up to 40 characters long.

Responses in POP3 consist of a status indicator and a keyword, possibly followed by additional information. Responses may be up to 512 characters long. There are two status indicators: positive ("+OK") and negative ("-ERR"). Servers send the "+OK" and "-ERR" messages in upper case only.

Responses to certain commands are multi-line. In these cases, after sending the first line of the response, any additional lines are sent. When all lines of the response have been sent, a final line is sent, consisting of a dot character ("."). If any line of the multi-line response begins with a dot, an additional dot is added at the start of the line to prevent it from being confused with the termination sequence. This extra dot is to be stripped by the client program at the user end and is thus transparent to the user.

In a typical POP3 conversation, the user application connects to the post office server and logs on with a USER and PASS command. The user then issues a STAT command and the server responds with a message telling how many email messages are waiting. The user application then uses the RETR and DELE commands to retrieve each message from the server and, if successfully retrieved, to delete the message from the server. Then the user application issues a QUIT command to log off of the server. The user then is free to read the messages while "off-line”, i. e. not connected to the email post office server.

The standard POP3 commands are:

USER (User Name)

This is usually the first command transmitted after a link is established. The argument identifies the identity of the email Post Office user for access to the post office server system.

PASS (Password)

This command must follow immediately after a USER command and the argument completes the identification procedure.

STAT (Status)

This command requests the status of the user's post office box. The server will respond with a message telling how many email messages are in the user's mail box. The response message has a rigid format, it is "+OK mm bb" where "mm" is the number of messages and "bb" is the total number of bytes of storage taken by the messages.

LIST (Message)

The message argument is optional. If it is not given, the server will respond with a list of the messages in the user's mail box. The list numbers each message in the box, beginning with one and incrementing by one for each message. The list also gives the number of bytes in each message. If the message parameter is given, it is the number of a message in the user's mail box on the server. The response is one line describing the message number and number of bytes for that message. The syntax of the LIST command is:

LIST [Message]

The following is an example of two LIST commands and the possible output:

Example 7.5-1:

```
LIST
+OK 2 messages (320 octets)
1 120
2 200
.

LIST 200
+OK 2 200
```

RETR (Retreive)

This command instructs the server to send a message from the user's post office box. The argument is the number of the message to be sent. The syntax of the RETR command is:

`RETR (Message)`

DELE (Delete)

This command instructs the server to mark a message in the user's post office box as deleted. The argument is the number of the message to be deleted. The syntax of the DELE command is:

```
DELE (Message)
```

NOOP (No Operation)

This command instructs the server to take no action other than to respond with an "+OK” reply message.

RSET (Reset)

This command instructs the server to Reset the transactions. If any messages are currently marked for deletion, but have not yet been deleted, the delete flag on those messages is cleared.

QUIT (Quit)

This command instructs the server to delete any messages that are currently flagged for deletion, then send an "+OK" reply and close the communication link. If the POP3 email Post Office server does not receive a QUIT command, any messages marked for deletion will not be deleted.

TOP (List Top of Message)

This command instructs the server to open a selected message, send the header information (identifying the sender, the message topic, and possibly other information), and the first few lines of the message. The purpose of this line is to allow a user to preview a large message without having to retrieve the entire message. The first argument (Message) identifies the message by number. The second argument (Lines) tells how many lines from the beginning of the message to send. The syntax of the TOP command is:

```
TOP (Message) (Lines)
```

UIDL (Unique Identifier List)

This command instructs the server to reply with a unique identifier for messages in the user’s mail box. The argument is optional. If it is not present, then the server will send a multi-line list with a unique ID for each message in the user’s mail box. If the argument is present, it identifies a specific message by number, and the response gives the unique message ID for that message. The form of the unique ID is server dependent. The only requirement is that it be unique and persistent. This means that for the life of the user ID on that server, it will not generate the same value for more than one message. It also means that the unique ID for a given message will persist from POP3 session to session. If the user disconnects, then reconnects at a later time, a given message will have the same unique ID (this is not necessarily true of numeric message ID numbers). The syntax of the UIDL command is:

`UIDL [Message]`

`APOP (Authenticate Post Office Protocol)`

The APOP command can be used instead of the USER and PASS commands to identify and validate a user. When a user first connects to a POP3 server, the server responds with a single line greeting message. If the APOP command is supported by a server, there will be specific information in the greeting message. A client that supports the APOP command combines this information with the user password to generate a unique code. The user name and this code are transmitted to the server as the arguments of the APOP command so that the password for a user is not transmitted across the network in clear text. The syntax of the APOP command is:

`APOP (Name) (Code)`

If a user does not have a POP3 client application, or for the purposes of testing and troubleshooting, it is possible to use a Telnet client application to contact a POP3 server and retrieve email messages. This is done by Telnetting to port 110 and entering the commands manualy.

7.5.2 IMAP

IMAP (Internet Message Access Protocol [RFC2060], formerly known as the Interactive Mail Access Protocol) is a protocol for email clients to retrieve email messages from, and work with the mailboxes on, a mail server. IMAP is the protocol that IMAP clients use to communicate with the servers. SMTP is the protocol used to transport mail to an IMAP server.

IMAP4, the latest version, is similar to POP3 but offers additional and more complex features. For example, the IMAP4 protocol leaves your email messages on the server rather than downloading them to your computer. If you wish to remove your messages from the server, you must use your mail client to generate local folders, copy messages to your local hard drive, and then delete and expunge the messages from the server. IMAP was developed at Stanford University in 1986.

The IMAP server provides a superset of POP functionality but has a different interface. (Thus, there are IMAP-specific mail clients and POP-specific mail clients.) The IMAP server provides an off-line service, as well as an on-line service and a disconnected service. The IMAP protocol is designed to permit manipulation of remote mailboxes as if they were local. For example, clients can perform searches and mark messages with status flags such as "deleted" or "answered." In addition, messages can remain in the server’s database until explicitly removed. The IMAP server also allows simultaneous interactive access to user mailboxes by multiple clients.

An IMAP server knows four states. Normally, the server is in the non-authenticated state after being connected. Here only the login and logout of a connection can take place. After authentication by the client, the server goes into the authenticated state. Now a mailbox can be chosen, on which is to be worked in the current session. Then, in the selected state messages in the mail box can be processed. After working on the mailbox, and from every other state, too, the server can change to the logout state to end the session.

In each of the mentioned states the server accepts many possible commands. In the authenticated state these are commands to handle (insert, modify, delete) folders. In the selected state, the client can get and modify single messages. Each IMAP command (as defined in RFC 2060) starts with a so called tag, which is a number for identification of the command. IMAP is a very complex protocol. For details, the reference of RFC 2060 is recommended. To get an impression, take a look at the following IMAP session:

```
C: demuth@eibauer:~ > telnet bonsai.fernuni-hagen.de 143
C: Trying 132.176.114.21...
C: Connected to bonsai.fernuni-hagen.de.
C: Escape character is '^]'.
S: * OK bonsai.fernuni-hagen.de IMAP4rev1 v11.237 server ready
C: 0000 LOGIN demuth elvisforever
S: 0000 OK LOGIN completed
C: 0001 SELECT INBOX
S: * 11 EXISTS
S: * 0 RECENT
S: * OK [UIDVALIDITY 943440696] UID validity status
S: * OK [UIDNEXT 33069] Predicted next UID
S: * FLAGS (\Answered \Flagged \Deleted \Draft \Seen)
S: * OK [PERMANENTFLAGS (\* \Answered \Flagged \Deleted \Draft \Seen)] Perma
S: * OK [UNSEEN 10] 10 is first unseen message in /var/mail/demuth
S: 0001 OK [READ-WRITE] SELECT completed
C: 0002 FETCH 1:10 (FLAGS)
S: * 1 FETCH (FLAGS (\Seen))
S: ...
S: * 10 FETCH (FLAGS ())
S: 0002 OK FETCH completed
C: 0003 FETCH 1 (RFC822)
S: * 1 FETCH (RFC822 {15271}
S: Received: from elm.fernuni-hagen.de (elm.fernuni-hagen.de [132.176.114.24
S: by bonsai.fernuni-hagen.de (8.8.8+Sun/8.8.8) with ESMTP id LAA12589
S: for <demuth@bonsai.fernuni-hagen.de>; Mon, 17 Jul 2000 11:54:58 +0200 (ME
S: Resent-Message-Id: <200007170954.LAA12589@bonsai.fernuni-hagen.de>
S: Received: from lgks1.fernuni-hagen.de by elm.fernuni-hagen.de
S: via local-channel with ESMTP; Mon, 17 Jul 2000 11:54:30 +0200
...
S: To: Multiple recipients of list news <news@gnn.de>
S: Reply-To: GNN Newsletter Feedback <feedback@gnn.de>
S: From: GNN Redaktion <ownergnnnews@gnn.de>
S: X-Mailer: GNN-Newsletter V2.0
S: Date: Mon, 17 Jul 2000 11:16:23 +0100
S: Subject: Golem Network News: 17. Juli 2000
```

```
S: Message-ID: <20000717111618.181611.in@mail.gnn.de>
S: Content-Length: 13300
S:
S: GOLEM Network News                                                - GNN -
S: 17.07.2000, 11:07                                      http://www.gnn.de/
S: ...
S: 0003 OK FETCH completed
C: 0004 FETCH 1 (FLAGS)
S: * 1 FETCH (FLAGS (\Seen))
S: 0004 OK FETCH completed
C: 0005 FETCH 1 (RFC822.SIZE)
S: * 1 FETCH (RFC822.SIZE 15271)
S: 0005 OK FETCH completed
C: 0006 LOGOUT
S: * BYE bonsai.fernuni-hagen.de IMAP4rev1 server terminating connection
S: 0006 OK LOGOUT completed
C: Connection closed by foreign host.
```

Some of the most important commands are shown in the following tables:

Commands possible in any state:

CAPABILITY	prompts the server to send a list of his capabilietes to the client
NOOP	mainly nothing; can be used to reset the time-out timer
LOGOUT	disconnects

Commands possible in Non-Authenticated State. Both commands are used to change to the authenticated state:

AUTHENTICATE <authentification scheme>	starts authentification with the given method (see also RFC 1731: IMAP4 Authentification Mechanism [RFC1731])
LOGIN <name> <password>	authentification

Commands possible in Authenticated State:

SELECT <mailbox>	selects a mailbox and provokes the server to change into the selected state.
EXAMINE <mailbox>	gives information about a mailbox
CREATE <mailbox>	creates a new mailbox.
DELETE <mailbox>	deletes a a mailbox
RENAME <mailbox> <new mailbox name>	renames a mailbox
SUBSCRIBE <mailbox>	activates a mailbox or newsgroup
UNSUBSCRIBE <mailbox>	deactivates a mailbox or newsgroup
STATUS <mailbox> <data>	gives information about the prompted mailbox
APPEND <mailbox> [<flags>] [<date>] <message>	adds a mail to the mailbox

Commands possible in Selected State:

`CLOSE`	deletes marked messages following a change to authenticated state
`EXPUNGE`	deletes marked messages
`SEARCH [<character set>] <criteria>`	searches the mailbox with the given criteria
`FETCH <messages> <data>`	gets one or more messages

7.6 Mailing lists

Mailing lists can be used to address many receivers at once or to establish mail based discussions using a **mail exploder**. Mail exploders are part of an email delivery system that allow a message to be automatically and efficiently delivered to a list of addresses, thus implementing mailing lists. Users send messages to a single address (e.g., `bsc-internet-course@fernuni-hagen.de`) and the mail exploder takes care of delivery to the individual mailboxes in the list.

Actually, the mailing list is a database of email addresses, used by the mail exploder. Normally, a mailing list offers two special addresses:

- Subscription address e.g., `bsc-internet-course-request@fernuni-hagen.de` This is an email address to subscribe to the mailing list. Mostly all other administrative functions can be handled with this address, once a user has become a member of the list. These functions are, among other, changing of the own address, temporarily deactivating the forwarding of messages, unsubscribing, etc.
- List address (see above): To this address actual messages can be sent to by a subscribed user. These messages are then transported to each list member.

Mails addressed to a mailing list address are sent to the mail exploder. The mail exploder obtains the mailing list data from the accounts database, expands (explodes) list traffic to the subscribed user addresses, and rewrites the messages so that they contain the appropriate list information (prologue text, epilogue text, etc.).

Tab. 7.6-1: *Database of a mail exploder*

list name	description
tutors	gerd.steinkamp@fernuni-hagen.de, dagmar.sommer@fernuni-hagen.de, michael.stepping@fernuni-hagen.de, thomas.demuth@fernuni-hagen.de
friends	john.doe@everybodies-darling.de, anne.nonymous@nowhere.us
other	bill@microsoft.com, hk@gangster.de

Table 7.6-1 shows a database with three mailing lists. To address a mailing list correctly, a user has to add the name of the server the mail exploder is running on. If the mail exploder runs at the FernUniversität, the correct mail address to send an email to all tutors is: `tutors@fernuni-hagen.de`. The email will then be forwarded to all four list members. With this mechanism a big group of users can communicate without the need to exchange their email addresses explicitely.

7.7 Netiquette

It is essential for each user on the network to recognise his responsibility in having access to the internet. The user is ultimately responsible for his actions in accessing network services.

The "Internet" is not a single network; rather, it is a group of thousands of individual networks which have chosen to allow traffic to pass among them. The traffic sent out to the Internet may actually traverse several different networks before it reaches its destination. Therefore, users involved in this internetworking must be aware of the load placed on other participating networks.

As a user of the network, you may be allowed to access other networks (and/or the computer systems attached to those networks). Each network or system has its own set of policies and procedures. It is the users responsibility to abide by the policies and procedures of these other networks/systems. Remember, the fact that a user can perform a particular action does not imply that they should take that action.

The use of the network is a privilege, not a right, which may temporarily be revoked at any time for abusive conduct. Such conduct would include the placing of unlawful information on a system, the use of abusive or otherwise objectionable language in either public or private messages, the sending of messages that are likely to result in the loss of recipients' work or systems, the sending of "Chain letters," or "broadcast" messages to lists or individuals, and any other type of use which would cause congestion of the networks or otherwise interfere with the work of others.

7.7.1 Some simple rules for behaviour on the internet:

- Typing mail messages all in upper case is considered SHOUTING! and rude.
- When quoting someone else, remove what isn't directly applicable to your reply. Don't automatically quote the entire body of messages you are replying to when it's not necessary. Leave only the minimum necessary to provide context for your reply.
- Be professional and careful what you say about others. Email is easily forwarded and often archived or stored, so whatever you say may come back to haunt you.
- Never send **chain letters** through the Internet. A chain letter is simply sending the same email someone sends to you to other people, just for the sake of sending the letter.

- Be careful when using sarcasm and humor. Without face to face communication the other person may take your words as criticism. When being humorous, use emoticons to express humor. (tilt your head to the left to see the **emoticon** smile) :-) means happy face.
- It is extremely rude to forward personal email to mailing lists (list-serv) or Usenet without the original author's permission.
- When you join a list serv or newgroup, monitor the messages for a few days to get a feel for what common questions are asked, and what topics are deemed off-limits. This is commonly referred to as lurking. When you feel comfortable with the group, then you can start posting.
- See if there is a **FAQ** (Frequently Asked Questions) for a group that you are interested in joining. Veteran members get annoyed when they see the same questions every few weeks.
- When signing up for a group, save your subscription confirmation letter for reference. That way if you go on vacation you will have the subscription address for suspending mail.
- Sending a subscription or unsubscription notice directly to the list instead of to a listserv is annoying to others. Only messages meant to be read by the entire group should go to the list.
- Send a personal mail message aimed at one person to that person - not to a publicly distributed news group or list-serv. Otherwise, be prepared to get email messages teasing you or expressing people's annoyance.
- Follow any and all guidelines that the list owner has posted; the list owner establishes the local "netiquette" standards for his list.
- When posting a question to a discussion group, request that responses be directed to you personally. Post a summary or answer to your question to the group.
- Posting an **advertisement in news groups**, unless it is specially chartered for that purpose like the forsale newsgroup or sending unsolicited advertisements with email is considered rude and in violation of the spirit of the Internet.
- If you must **cross-post** messages to multiple news groups, include the name of the groups at the top of the mail message with an apology for any duplication.
- Be courteous and respect other people. If you are abusive, you run the risk of the Internet community using its own form to chastise you. The Internet community can use their individual or collective "voices" to inform and sometimes even ostracise you with "flame" (see below) messages. If your boorish behavior persists, they can contact your Network Administrator to report your abuses.

7.7.2 Further remarks on Netiquettes

- Many beginners try to use all features their email clients offer. This leads to annoying things like **HTML in emails** or **Vcards** (electronic business cards) in emails. This is bad behaviour (s. exercises for explanation).

- Don’t "**flame**." A flame is an inflammatory or critical message. Avoid sending junk emails, emails with insufficient information or any other email that might trigger an upsetting response from the recipient. If you do get flamed, the best thing to do is to just ignore the message. Responses to flames can escalate into "flame wars."
- Don’t "**spam**". Spam, used in reference to email, means electronic garbage. Sending junk email (such as an advertisement) to a newsgroup or a list serve, or to anyone you don’t know, is considered "spamming." "Spamming" often causes "flames."

And:

- Be patient!

8 World Wide Web

8.1 Goal of the Chapter

This chapter gives an overview over technical foundations of the World Wide Web.

The chapter starts with an explanation of the basic concepts of the World Wide Web (WWW). The Standard Generalized Markup Language (SGML) is shortly introduced which forms the basis for other markup languages like HTML and the Extended Markup Language (XML). In the following section basic elements of HTML are exlpained. After reading Section 8.4, the reader will be able to write simple HTML documents.

With increasing complexity of the tasks concerned with Web authoring it is important to follow generally approved guidelines. One such guideline is to separate content from presentation. In case of HTML authoring this is achieved by employing Cascaded Style Sheets (CSS) which will be introduced in Section 8.5.

The Extended Markup Languages (XML) aims at further improving the concept of separation of the content from the presentation. XML fascilitates the creation of content that must be displayed on a variety of devices like browsers, PDAs, Web TVs, WAP mobile phones, etc. The basic concepts of XML will be introduced in Section 8.6.

The transport protocol that is employed for the transmission of web documents is the Hypertext Transport Protocol (HTTP). It will be introduced in Section 8.7.

8.2 Fundamentals of the World Wide Web

The main concepts behind the Web's way of structuring documents are multimedia, hyperdocuments, and distributed documents.

Multimedia is the integration of different media types into one document model, e.g. a video sequence in combination with an audio sequence.

Hyperdocuments can be divided into **Hypertext** and **Hypermedia.**

Hypertext is non-sequential writing. The term was introduced by Theodor Nelson in 1965 as a body of written or pictoral material interconnected in such a complex way that it could not conveniently be presented or represented on paper [Len97]. Nelson described hypertext systems as text chunks connected by links which offer the reader different pathways.

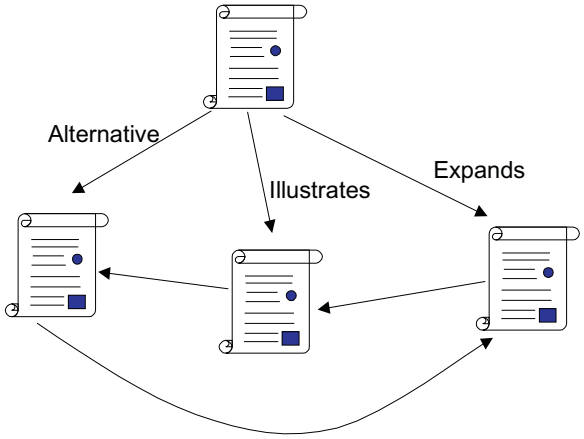


Fig. 8.2-1: Hypertext

Hypermedia (see Fig. 8.2-2) extends the Hypertext concept in that also multimedia objects can be incorporated in addition to the text chunks. The resulting hypermedia document is a media mix consisting of text and multimedia objects which can be arbritrarily linked. In practice, hypermedia and hypertext are very often treated as synonyms. Table 8.2-1 shows some examples of hypermedia systems.

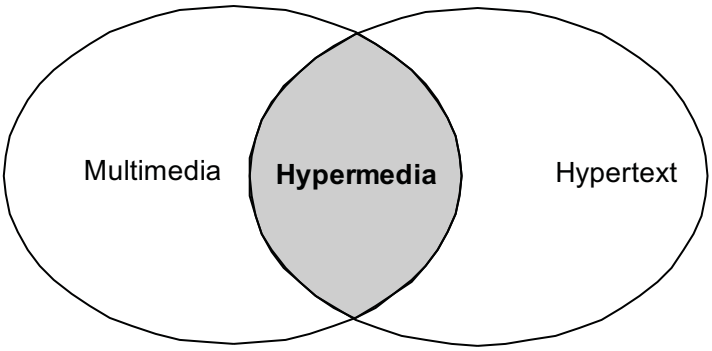


Fig. 8.2-2: Definition of hypermedia

Distributed documents are scattered across a network. This means that complete documents or parts of them can reside on different computers.

The Web can be said to to implement a **distributed hypermedia document model**.

Tab. 8.2-1: *Examples of Hypermedia systems*

System	Description
Xanadu	1965, Theodor Nelson, "Inventor" of hypertext
Hyper Card	1987, Predecessor of many following hypermedia systems, Apple Macintosh
WWW	1990, global distributed hypermedia system based on Hypertext Markup Language (HTML) on top of the Internet infrastructure
HM-Card	1993, Freeware Hypermedia System
MPEG-4	1999, Next generation hypermedia system standardized by ISO

The basic idea of a globally interconnected hypermedia system was developed at the European Laboratory for Particle Physics (CERN) in 1990. By 1992, 26 reliable web servers were available. The first important development which boosted the popularity of the Web was Marc Andresseen's **Mosaic** in 1993. This was the first comfortable and powerful browser, and soon after its release the general public began to recognize the Web for the first time.

In 1994, the **World Wide Web Consortium (W3C)** was founded as an international organization for standardization of the Web. In the beginning, the key areas for development were the Hypertext Transport Protocol (HTTP) and the Hypertext Markup Language (HTML). Nevertheless, the standards released by the W3C had little influence because Netscape, which was founded by Marc Andresseen in 1994, dictated the HTML syntax by inventing proprietary tags, thus producing de-facto standards which had to be followed by all HTML applications.

In 1995, the entrance of Microsoft with the Internet Explorer marked the beginning of the so-called **browser war**. In the following, the two contenders Microsoft and Netscape tried to achieve market dominance by frequently implementing new features in their browser products.

Finally, by the end of the last decade, the leadership of the W3C regarding standardization was accepted by both companies.

The Web relies on three mechanisms for the dissemination of information. These are:

1. A uniform naming scheme for locating resources on the Web, e.g. **Uniform Resource Identifier (URI)**.
2. Protocols, for access to named resources over the Web, e.g. **Hypertext Transport Protocol (HTTP)** and **Wireless Access Protocol (WAP)**.
3. Hypermedia document format, for easy navigation among resources, e.g. **Hypertext Markup Language (HTML)**.

One significant difference of the Web in comparison with other Hypermedia systems is, that links are not locally restricted. As long as a URI can be defined, the resource can be accessed all across the Internet.

8.2.1 Universal Resource Identifier (URI)

Since the Web is a distributed information system there must be a way to uniquely specify a resource. Most people who already have worked with the Web know the meaning of the term **link**. A link is a kind of pointer which connects the document in the current context with an information resource in another context. In the Web, this linkage is established with a **Universe Resource Locator (URL)**. So what is the difference between a URI and a URL?

This questions can be answered by carefully looking at the different meanings of *identifier* and *locator*. An identifier is more than just a link. It provides a unique identification of a resource indepent of its current location. A locator on the other hand, exactly specifies where a resource with a certain identifier can be found.

An URI allows the ditinction between a **Universal Resource Name (URN)** and the **Universal Resource Locator (URL)** (see Fig. 8.2-3). URNs are used to specify the names of information resources, while URLs are used to specify their addresses.

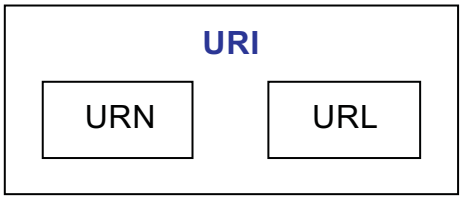


Fig. 8.2-3: URI

Example 8.2-1:
The current HTML 4.0 standard can be arbitrarily described by the following identifier:

`W3C/HTML401/revDec1999`.

The actual document can be in different places all across the Web. It will surely be somewhere on the W3C web server, e. g.

http://www.w3.org/TR/html4/,

but can also be mirrored by multiple other web sites, e. g.

`http://www.webdesign.com/standards/html401.html` or

`http://www.academy.com/courses/web/html/standard.html`

The syntax of a URI is the following:

```
uri = scheme ":" scheme-specific-part
```

The **scheme** of a URI identifies the naming scheme which is used for this particular URI. This part of the URI is seperated from the rest of the URI by a colon. Currently, a number of schemes are well defined, and the **Internet Assigned Numbers Authority (IANA)** maintains a list of these schemes and references to their definitions.

The **scheme specific part** of a URI contains the actual identification of the particular object in a scheme specific way. The interpretation of this part of the URI depends entirely on the scheme being used.

Example 8.2-2:
The URI `http://www.fernuni-hagen.de/` can be seperated into the scheme `http` which defines that the object addressed can be accessed by the HTTP protocol (see Section 8.7), and the scheme specific part
`//www.fernuni-hagen.de/`,
which defines the actual object being referenced.

It is not possible to differentiate between URLs and URNs syntactically. Hence, it is always necessary to interprete the URI scheme part in order to decide wether it is a URN or URL.

8.2.1.1 Uniform Resource Locator (URL)

Currently, a number of schemes are defined by the IANA in `RFC 1738`Some popular URL schemes are listet below:

- http: Access to the desired object via the HTTP protocol
- https: Secure Access via HTTP over SSL (secure socket layer)
- ftp: Access to the object via the FTP protocol
- file: Access to a host-specific file (e.g. on the local hard drive)
- mailto: Specifies an email address
- news: This scheme refers to newsgroups

The scheme specific part has the following general syntax (parts in brakkets are optional):
`"//"[user[":"password]"@"]host[":"port]"/"] url-path`

Example 8.2-3:
`http://wallace:gromit@www.bbc.com:8080/members.html`

Here, the user `wallace` with the password `gromit` connects to a restricted web site which is placed on a web server that runs on port `8080` (the default port for web servers is 80). Note: There are two security issues in this example. Firstly, in most cases it is not advisable to include sensitive information like user name and password in a URL. Secondly, do not use pet names as passwords. They can be guessed very easily.

8.2.1.2 Uniform Resource Name (URN)

URNs are persistent identifiers for information resources. URNs are still under development and at the moment there is no infrastructure that supports URNs. This infrastructure is necessary to resolve the URL for a given URN (similar to DNS). The syntax of URNs is defined by `RFC 2141`. At the moment, it does not seem likely that URNs will be employed in the Web in the near future. In practice, URI and URL can be used as synonyms. The official standards of the W3C use the term URI.

8.3 Standard Generalized Markup Language

8.3.1 SGML concepts

The introduction of computers in many facets of human activities also reflects on the field of publishing. Since the early 1980s a constant transition from paper based publishing to computer assisted publishing has occured. Today it is common to use computer based word processing systems to write letters, newspapers, books, WWW pages, etc. Some important advantages of computers in publishing applications are

- High volumes of data can easily be stored and managed.
- It is much more comfortable to edit large documents in computers than on paper.
- Electronic documents can be exchanged and distributed very easily.
- It is possible to derive multiple "views" from the same source document. For instance, an address list can be turned into a directory on paper, put on CD-ROM, made available as a database to allow interactive or email access on the Internet, or used to print a series of labels.

To allow such applications with different views on a document it is not sufficient to employ word processing systems that emphasize only **WYSIWYG** (What you see is what you get) oriented display and printing. These word processors are suitable for office and private applications but are not able to provide satisfactory semantical and structural functions.

For this reason, the Standard Generalized Markup Language (SGML) was defined and standardized by ISO in 1986. The basic idea of SGML is the separation of content and presentation. SGML only deals with content and the structure of content, and leaves the presentation of this structured content to other mechanisms. SGML is employed in large documentation projects for the aerospace, automotive, semiconductor, defense, telecommunication and other industries. In these kind of industries information and documents have to be stored over decades and must be accessible any time. Given the case that today a document is stored in a format of a popular word processor there is no guarantee that this piece of software will exist in ten years from now. Even if the software would still exist, the probability that it could read the old format is very low.

SGML's seperation of content and presentation allows that authors (e. g. scientists) can concentrate on content and publishing specialists can concentrate on the presentation. Although SGML was successfully employed in the publishing industry its popularity stems from the fact that it is the basis of HTML. In fact, HTML is completely defined in SGML and is said to be a **SGML application**.

An important concept of SGML is that the structure of SGML documents is defined by **document classes**. Although the notion of document classes is not important for HTML (there is only one document class for HTML) it is of increasing importance for the Web because of the advent of the **Extensible Markup Language (XML)** which allows arbitrary document classes. In contrast to HTML, XML is not a SGML application but a subset of SGML.

8.3.2 The SGML Document Instance

Markup is a general concept of mixing content and structural information in the same document. Structural information is defined in terms of **elements** which have unique names like `chapter` or `section`. To distinguish markup information from regular content, element names are placed inside **markup delimiters** ("<" and ">"). An element name with markup delimiters is also called **tag**. Tags are employed to visually mark start and end points of an element. An example of a simple SGML document may is displayed below:

```
<book>
<heading>Computer Networks</heading>
<author>Andrew S. Tanenbaum</author>
<chapter><heading>The Physical Layer</heading>
<section><heading>Transmission Media</heading>
<paragraph>
...
</paragraph>
</section>
<section><heading>Transmission and Switching</heading>
<paragraph>
...
<paragraph>
...
</section>
</chapter>
</book>
```

It can be seen that the document is hierarchically structured with meaningful tags[3] (semantic information). It can be observed that there are two types of tags:

3 The tags have a meaning for human users, not computers

- `<...>` tags mark the beginning of a structural component (element), e. g. the `<chapter>` tag identifies the beginning of a new chapter.
- `</...>` tags mark the end of a structural component, e. g. the `</section>` tag identifies the end of a document section.

In the example above, there are some `<paragraph>` tags which have no associated ending tag. This is possible in SGML and HTML when it is clear, where the ending tag can be expected (in XML, it is not allowed to omit ending tags).

The structure of the SGML document can be displayed graphically. Fig. 8.3-1 shows the structure of the example document and it can be seen that the document exhibits a tree structure with the `book` element as root element. Other elements follow as branches of the root element. The leaves of the tree are the text elements inside the tags. Hence, the graphical representation of the document is called the **document tree**. In this example the `book` element is referred to as being the **parent** of the `chapter` and the `author` element. The chapter element itself has two **children**: the `heading` and the `section` elements. The `heading` and the `section` elements are **siblings**.

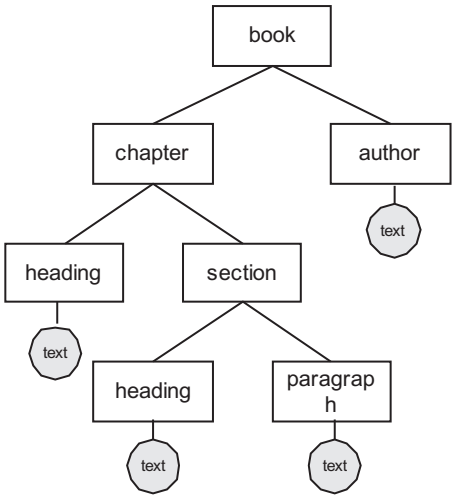


Fig. 8.3-1: Structure of an example SGML document

Besides the element concept SGML defines **attributes** which allow additional information inside an element:

```
<chapter id="infra">Internet Infrastructure</chapter>
<heading>The Physical Layer</heading>
<section id="sct-transmedia">
<heading>Transmission Media</heading>
<paragraph>
```

In this example the `chapter` and `section` element are augmented with an `id` attribute. In this example, the `id` attribute holds additional information which can be used to refer to the chapter and section. A reference from within a paragraph to the chapter "Internet Infrastructure" may be constructed as follows:

```
<paragraph>In<reference style="link" id="infra">it was
explained that ...
```

Here, the `style` attribute is used to specify that the reference should include the a hyperlink to the referred chapter. Depending on the type of formatting this can have different effects. If the document is formatted as a book a hyperlink cannot be employed and the reference could be resolved into

```
In chapter 3 on page 34 it was explained that ...
```

This type of reference does not make sense on web pages where there is no page mechanism. For a web page the reference can be resolved as follows:

In chapter 3 it was explained that ...

Here, a hypertext link is created which points to the referred chapter.

8.3.3 Presentation of Contents

It is obvious that the examples above do not include any information about the presentation of the text (e. g. no font type or size). It only consists of logical elements which are arranged in a certain structure. This means that the author can write the contents of the document without knowing how it will be displayed in the finished product. Whether the contents of the document are displayed on a computer screen, represented aurally or printed as a book has nothing to do with the document itself. This is the concept which makes SGML so popular in the publishing industry. For the presentation of the structural elements SGML specifies the **Document Style Semantics and Specification Language (DSSSL)**. Since DSSSL is very powerful and complex it will not be described in detail here. There are two other similar style languages which are of greater importance for the Web and will be described later in the chapter. These are **Cascading Style Sheets (CSS)**and the **Extensible Style Language (XSL)**.

8.3.4 The Document Type Definition (DTD)

To fully exploit the advantages of structured documents, the markup scheme must adopt a clear set of rules. In SGML, these rules are defined in a **document type definition (DTD)**. The DTD specifies which elements and attributes are allowed and also defines the structural relations between the elements. By defining clear rules the exchange of documents between authors and publishers is fascilitated. For the book document in Section 8.3.2 the DTD may look as follows:

```
<!ELEMENT book        (author,heading,chapter+) >
<!ELEMENT chapter     (heading,paragraph*,section*) >
<!ELEMENT section     (heading,paragraph*) >
<!ELEMENT author      (#PCDATA) >
<!ELEMENT heading     (#PCDATA) >
<!ELEMENT paragraph   ((#PCDATA|reference)*) >
<!ELEMENT reference   EMPTY >

<!ATTLIST (chapter|section)
   id CDATA #IMPLIED >
<!ATTLIST reference
   id CDATA #REQUIRED
   style (link|page) link >
```

The DTD in this example consists of two parts. The first part contains **element declarations** and the second part consists of **attribute declarations**. The syntax rules applied in this DTD are listed in Table 8.3-1.

The DTD specifies that a book element contains an `author` and a `heading`, and must contain at least one `chapter` (indicated by the "+"). A `chapter` contains a `heading` and may contain several `paragraph`s and `section`s (the "*" indicating zero or more occurences of a `paragraph` or `section`). A section also contains a `heading` and several `paragraph`s. The `author` and `heading` element do not accomodate any further elements and only contain actual content in form of characters. Such elements are identified by the **keyword**`#PCDATA`. A `paragraph` may contain textual content and references to other parts of the text (the "|" indicates an "or" relation between `#PCDATA` and `reference`). The last element in the list is the `reference` which neither contains any elements nor textual content (it only contains attributes). This is indicated by the keyword `EMPTY`.

Tab. 8.3-1: *Basic SGML syntax rules*

Syntax rule	Meaning
`(...)`	Delimits a group
`A`	A must occur, one time only
`A+`	A must occur one or more times
`A?`	A must occur zero or one time
`A*`	A may occur zero or more times
`+(A)`	A may occur
`-(A)`	A must not occur
`A | B`	Either A or B must occur, but not both
`A , B`	Both A and B must occur, in that order
`A & B`	Both A and B must occur, in any order

In the attribute declaration the first attribute to be defined is the `id` attribute which can be used inside `chapter` and `section` elements (`chapter` and `section` elements share the same definition). The keyword `CDATA` indicates that the `id` attribute takes character data as values. This attribute can be used optionally inside the elements which is expressed by the `#IMPLIED` keyword. The `reference` element contains two attributes. Since a reference without an identification of what is referred to does not make any sense the `id` attribute is mandatory for `reference` elements. This is indicated by the keyword `#REQUIRED`. The second attribute is the `style` attribute which specifies if the reference should be made in form of a hyperlink or page reference. If no `style` attribute is given the default value is `link`.

SGML also contains techniques for adding standard (boilerplate) text to a document and for handling characters that are outside the standard character set, but which are available on certain output devices.

Commonly used text can be declared within the DTD as a text **entity**. A typical text entity declaration could take the form:

```
<!ENTITY fernuni "FernUniversitaet Hagen">
```

Once such a declaration has been made in the DTD users can use an entity reference of the form `&fernuni;` in place of the full sequence. An advantage of using this technique is that, should the name of the university referred to by the mnemonic change later (e. g. into "Universität Hagen"), only the entry in the DTD needs to be changed as the entity reference will automatically call in the latest definition. The "`&`" is called **entity reference open delimiter (ERO)** and the "`;`" is called **entity reference close delimiter (REFC)**. This type of entity reference is also called **general entity reference**.

Another type of entity reference is the **character entity reference** which allows the specification of characters which are not available using a keyboard. For example, the german Umlaut ö (which is not available on an american keyboard) can be adressed as `&#214;`. Here the character sequence "`&#`" is called **character reference open delimiter (CRO)**.

In HTML, entity references are made for special characters.

```
<!ENTITY Ouml CDATA "&#214;" >
```

This general enitity declaration declares the `Ouml` entity to be replaced by the character entity reference "`&#214;`", which references the german upper case letter Ö within the `ISO 10646` character set.

8.3.5 The SGML declaration

The SGML declaration specifies basic facts about the dialect of SGML being used such as the character set, the codes used for SGML delimiters (we have used "<" and ">" so far), the length of identifiers, etc.. Furthermore, SGML defines a number

of options which can be used in different ways. These are all things which have to be defined prior to developing an actual DTD.

8.3.6 SGML applications

An SGML application consists of all the parts which were described in the preceding sections:

1. An SGML declaration that specifies which characters and delimiters my appear in the application
2. A DTD which defines the syntax of markup constructs.
3. The document instance containing content and markup. Each document instance contains a reference to the DTD to be used to interpret it.

The processing system that analyses the SGML document instance and checks if it conforms to the DTD is called **Parser**.

8.4 Hypertext Markup Language (HTML)

8.4.1 Brief history of HTML

HTML was originally developed by Tim Berners-Lee while at CERN, and popularized by the Mosaic browser developed at NCSA. Table 8.4-1 gives an overview of HTML's version history.

Tab. 8.4-1: *HTML/XML version history*

Standard	Year
HTML 2.0	1994
HTML 3.2	1997
HTML 4.0	1997
XML 1.0	1998
HTML 4.01	2000
XHTML 1.0	2000
XHTML 1.1	2000

Originally, HTML is a markup language which was defined in SGML. In the course of the last few years it was recognized, that HTML is not flexible enough for the new Web applications. Furthermore, it was realized, that the Web needed a format which was more about content than about how to present the content. SGML was too complicated to be successfully employed in the Web environment, and thus a new markup language was defined by the W3C, the **Extensible Markup Language (XML)**. XML was conceived to overcome the shortcomings of SGML for web applications and is a subset of SGML. XML is said to provide 80% of SGML's functionality while only showing 20% of its complexity. XML is not supposed to replace HTML but provide means to create content without having to deal with the

presentation of that content. It is expected that XML will have a great impact on the future of the Web.

Year 2000 marks an important change in the history of HTML. Due to the growing importance of XML the W3C decided to newly define HTML in XML (thus leaving SGML). This new HTML standard was named **Extensible HypertText Markup Language (XHTML)** and is about to completely replace HTML 4.0 in the near future. The most significant properties of XHTML are that it is extensible and is more strict in regard to syntax conformance than the original HTML, while still providing a certain compatibilty to older HTML standards.

Nevertheless, in this course we will refer to the HTML 4.01 standard which is the most popular version at the moment and will remain so in the near future.

8.4.2 Basic HTML

HTML was originally conceived to be a language for the exchange of scientific and other technical documents, suitable for use by non-document specialists. HTML addressed the problem of SGML complexity by specifiying a small set of structural and semantic elements suitable for authoring relatively simple documents. In addition to simplifying the document structure, HTML added support for hypermedia.

There are three DTDs which can be used for the interpretation and generation of HTML documents. (see Table 8.4-2)

Tab. 8.4-2: *HTML DTDs*

DTD	Description
Transitional DTD	The transitional DTD should only be used for interpreting HTML documents because it includes deprecated elements and attributes. These elements and attibutes are still valid but will not be supported by future HTML standards. User agents should still be able to display these tags correctly but they should be avoided during the generation of HTML documents. Examples of tags which should be avoided are: <BASEFONT>, <CENTER>, <FONT>, <S>, <STRIKE>, and <U>. The formatting functionality of these tags should be achieved by using Cascaded Style Sheets.
Strict DTD	The HTML 4.0 DTD includes all elements and attributes that have not been deprecated or do not appear in framesets. This DTD should be emloyed for the generation of HTML 4.0 documents.
Frameset DTD	Since HTML 4.0 frames are officially supported. This DTD should be used in HTML documents that define a frameset.

An HTML 4.0 complient browser contains a parser that is able to validate documents according to all three DTDs. The parser in an HTML browser uses this fixed set of three DTDs and the HTML SGML declaration (see Fig. 8.4-1).

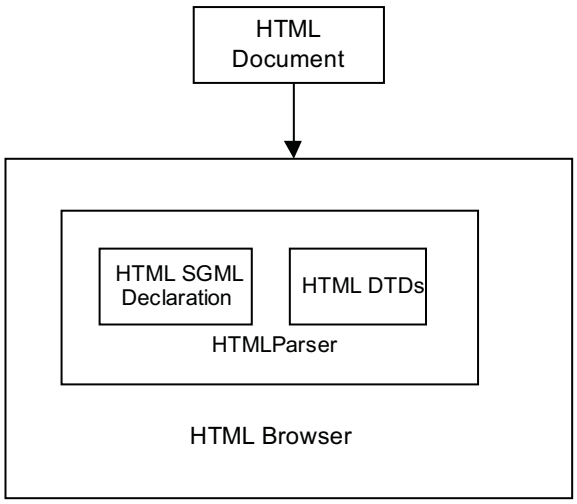


Fig. 8.4-1: HTML Browser

8.4.3 Basic Structure of an HTML Document

The basic structure of an HTML document looks as follows:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
                "http://www.w3.org/TR/REC-html40/strict.dtd">
<html>
<head>
<title>Title of the document</title>
</head>
<body>
Content goes here ...
</body>
</html>
```

In every HTML document the HTML version must be indicated by specifying the HTML DTD. In this case the HTML strict DTD is employed which means that deprecated elements are not allowed.

An HTML document consists of two parts, a document head which is specified inside the `<head>` tags and a document body which is specified inside the `<body>` tags. The document head contains information about the document, e. g. the title of the document. The document body contains the actual content that is displayed when the document is viewed with a visual user agent (browser).

8.4.4 Document Head

8.4.4.1 Title element

Every HTML document *must* have a `title` element in the `head` section. Authors should use the `title` element to identify the contents of a document.

8.4.4.2 Meta data

Meta data is information about a document rather than document content. Meta data can be used by Web search engines to classify web pages according to their content.

```
<head>
<meta name="author" content="John Rambo">
<meta name="keywords" content="slashing and burning, Vietnam">
</head>
```

Here the `meta` element is used to identify the author of the HTML document and to specify keywords that a search engine may use to improve the quality of search results.

8.4.5 Document body

The content of the HTML document resides in the `<body>` section. The content must not necessarily be presented visually on a computer screen. Special user agents can also speak the content of a HTML document (this can be helpful for disabled people).

HTML groups content elements into two catagories, which are **block-level elements** and **inline elements**. Block-level elements create larger structures than inline elements. Generally, block-level elements begin on new lines, inline elements do not. Furthermore, block-level elements can contain inline elements, inline elements may not contain block-level elements.

```
<body>
<p>This is an example how a block-level
element can contain an <em>inline-element</em>
</p>
<body>
```

In the example above the `body` and `p` elements are block-level elements and `em` is an inline element. The block-level element `body` contains the block-level-element `p`, the block-level-element `p` contains the inline-element `em`.

8.4.6 Text

In most cases the textual information is the most important part of an HTML document. HTML provides several elements to structure textual content in form of paragraphs, headings and phrases.

8.4.6.1 Lines and paragraphs

Usually, a long text document is logically divided into a sequence of paragraphs. HTML defines the `p` element for the representation of paragraphs.

```
<body>
<p>This is a first paragraph</p>
<p>This is another paragraph</p>
</body>
```

If it is necessary to specify line breaks within paragraphs, this can be done using the `br` element.

```
<body>
<p>This paragraph breaks <br>
into two lines (at least)</p>
<p>This is another paragraph</p>
</body>
```

It must be noted, that in the HTML 4.0 standard the `br` element only consists of a start tag , which means that constructs like `<br>...</br>` are not allowed! This will be different in the new XHTML standard.

8.4.6.2 Headings

A heading element briefly describes the topic of the section it introduces. Since documents can be structured in chapters, section, subsection, etc. HTML defines six levels of headings numbered 1-6 (Level one is the largest) that can be used to mark up the content.

```
<body>
<h1>Chapter 1</h1>
<p>content goes here</p>
<h2>Section 1</h2>
<p> .... </p>
<h6>subsection</h6>
<p> .... </p>
</body>
```

8.4.6.3 Phrases

Phrases allow the semantical markup of text. HTML defines the following phrase elements:

- `em` indicates that a piece of text should be emphasized.
- `strong`indicates that a piece of text should be strongly emphasized.
- `dfn` indicates that its content is a definition.
- `code` designates a fragment of computer code.
- `samp` designates sample output from programmes, skripts, etc..
- `kbd` indicates text to be entered by the user.
- `var` indicates an instance of a variable or programme argument.
- `cite` contains a citation or a reference to other sources.
- `abbr` indicates an abbreviated form (WWW, URI, etc.).
- `acronym` indicates an acronym (sonar, radar, etc.).

Example 8.4-1: HTML Acronyms

```
<acronym title="Radio Association Defending Airwave Rights">
   Radar
</acronym>
```

In this example the `title` attribute is used to provide the full meaning of the acronym. The title attribute is part of the core attribute set (`id`, `class`, `style`, `title`) which is available for most HTML elements. The presentation of the content of the title attribute depends on the user agent. For instance, Netscape 4.72 does not display the title information, IE 5 displays it as a "tool-tip" when the mouse pointer moves over the term.

8.4.7 Quotations

There are two elements for the inclusion of quotations. The `blockquote` element is used for long quotations and the `q` element is used for short (inline) quotations. The `q` element should be rendered with language specific quotation marks by the browser.

Example 8.4-2: HTML Quotations

```
<blockquote cite="http://www.stephenking.com/files/theplant/html/Plant2.htm
<p>JOHN KENTON, who attended Brown University, majored
in English, and was president of the Literary Society, has
had a rude awakening in the real world: he is one of four
editors at Zenith House, a down-at-the-heels paperback
```

```
publisher in New York.</p>
</blockquote>


Caesar said <q>Veni, Vidi, Vici</q>
```

8.4.8 Subscripts and superscripts

The elements `sub` and `sup` can be used to lift or lower text elements.

Example: HTML Subscripts/Superscripts

```
H<sub>2</sub>O
E = mc<sup>2</sup>
```

8.4.9 Lists

HTML offers three different mechanisms to create lists:

- Unordered lists (e. g. bulleted lists)
- Ordered Lists (e. g. numbered lists)
- Definitions (e. g. a term with its description)

8.4.9.1 Unordered lists

An unordered list is created with the `ul` element. List items can be specified by the `li` element. The item label (bullets, hyphens, etc.) can be selected by using style sheets (see Section 8.5.3.5).

Example: HTML Unordered Lists

```
<ul>
<li>item</li>
<li>another item</li>
<li>and another item</li>
</ul>
```

8.4.9.2 Ordered lists

An ordered list is created with the `ol` element. The numbering scheme (arabian, roman, etc.) can be effected by style sheets (see Section 8.5.3.5).

Example: HTML Ordered Lists

```
<ol>
<li>first item</li>
<li>second item</li>
<li>third item</li>
<ol>
<li>second level item</li>
<li>another second level item</li>
</ol>
</ol>
```

This example shows that lists can also be nested.

8.4.9.3 Definition lists

A definition list does not use fixed labels or numbers but allows the author to define the label. A typical application is the definition of a term, where first the term itself is displayed followed by its description.

Example: HTML Definition Lists

```
<dl>
<dt>Domain Name System</dt>
<dd>A distributed, replicated, data query service
mainly used on the Internet for translating host names
to IP addresses.</dd>

<dt>Data Encryption Standard</dt>
<dd>DES is a symmetric cryptosystem, when used for
communications, both sender and receiver must know the
same secret key, which is used both to encrypt and
decrypt the message.</dd>
</dl>
```

As already mentioned, the appearance of lists, e. g. the definition of the list-item marker or the numbering format (roman, decimal, etc.), can be influenced by style sheets. See Section 8.5.3.5 for properties that can be used to change the presentation of lists.

8.4.10 Tables

With HTML tables data can be arranged into rows and columns of cells. A table is represented by the `table` element which consists of an optional caption, represented by the `caption` element, and the table specification itself. The caption is followed by optional `col` elements and `colgroup` elements which specify column widths and groupings. Also optional are `thead` and `tfoot` elements, which contain header and footer rows. The only mandatory part of the `table` element is the `tbody` element, which constitutes the table body. It must be noted that the `tbody` start and end tags are both optional, which means that even if there are no `<tbody></tbody>` tags, the `tbody` element is still there. A minimal HTML table consists of a table body, which is a sequence of one or more rows specified by `tr` elements, which in turn are sequences of table header (`th` elements) or data cells (`td` elements).

Example: HTML Table

```
<table>
<caption>HTML DTDs</caption>
<tr>                        <!-- start of first row -->
<th>HTML DTD</th>  <!-- 1. header cell -->
<th>Description</th>
</tr>                       <!-- end of first row -->
<tr>
<td>Strict DTD</td>
<td>Includes all elements and attributes that have
not been deprecated or do not appear in framesets</td>
</tr>
<tr>
<td>Transitional DTD</td>
<td>Should only be used for <em>interpreting</em>
HTML documents because it includes deprecated elements
and attributes</td>
</tr>
</table>
```

The rows of the table are defined by `tr` elements. A `tr` element contains header or data cells, represented by `th` and `td` elements. Although table data and header cells are very similar, table header cells are formatted a little bit different than table data cells.

The HTML table in this example will be displayed without any rules and borders. The attribute `border` allows a border width to be specified by the author.

Example 8.4-3:

```
<table border="1">
...
```

This defines a table which will be displayed with rules and borders with a width of one pixel.

This is the first attribute so far that only controls visual formatting of an HTML element. The table elements are the only set of elements that still have formatting attributes associated with them wich are not deprecated. The reason for this is that the style sheet language which is targeted at HTML, Cascading Style Sheets version 1 (CSS1, see Section 8.5), does not properly support tables. Hence, style information is still specified inside elements of the table set.

8.4.11 Links

One of the most powerful features in HTML is the ability to create links that point to other HTML documents. Generally, a link consists of two ends called anchors with a directed connection between them (see Fig. 8.4-2). The source anchor normally is a piece of text or a graphic that can be selected by clicking on it. The destination anchor can be any kind of web resource like another web page, a graphic, a multimedia file, etc. The activation of the link initiates the retrieval of the destination resource over the Web.

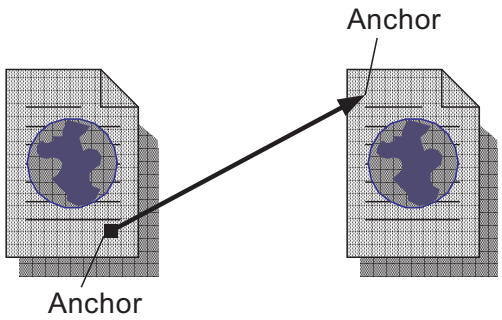


Fig. 8.4-2: Link

Anchors for a link can be defined with the a element.

```
This <a href="target.html">link</a> points to a target document.
```

This example defines a link with the word *link* as source anchor and the web page *target.html* as destination anchor. The attribute `href` specifies the address (URI) of the destination anchor. Most browsers will render the text inside the a element in a special way, such as using a different colour or underlining it.

The next example shows how a link can point inside a destination document by specifying a destination anchor. (see Table 8.4-3)

Tab. 8.4-3: *Linking into a HTML document*

Source document	`This <a href="target.html#target">link</a> points to the word anchor in the destination document (target.html).`
Destination document (version 1)	`This destination document with the name "target.html" includes an <a name="target">anchor</a> inside the text.`
Destination document (version 2)	`This destination document with the name "target.html" includes an <em id="target">anchor</em> inside the text.`

The source anchor includes a href attribute as before, but this time the anchor in the destination text is specified in a URI including its name as a **fragment identifier**. There are two possibilities to create destination anchors:

1. The destination anchor can be defined by the `a` element which includes a name attribute specifying the name of the anchor.
2. The destination anchor can be defined by any regular HTML element and this element includes an `id` attribute specifying the name of the anchor.

If source and destination anchor reside in the same document, only the fragment identifier portion of the URI can be used to create a link:

```
This <a href="#target">link</a> points to the word
anchor in the same document.
...
...
...
This part of the document includes an
<em id="target">anchor</em> inside the text.
```

8.4.12 Images and Multimedia Objects

HTML allows the inclusion of all kinds of multimedia objects like images, video clips and Java applets inside an HTML document.

Images are the most important multimedia objects in the World Wide Web. There are two methods to include images inside of web pages. The traditional way is to use the `img` element. The other way which is advocated by HTML 4.0 is to use the generic `object` element. Nevertheless, since most user agents today do not correctly display the `object` element, the `img` element should still be used.

It must be noted that HTML does not specify any special image formats. Nevertheless, there are three important image formats which are widely supported by browsers:

- The **Graphics Interchange Format (GIF)** format is well suited for images which only consist of 256 colors (8-bit colour resolution). An important feature is, that a transparent background colour can be defined which fascilitates the seamless inclusion of images in web pages. A disadvantage of this format is, that applications which make use of the GIF format must obtain a liscence for the patented compression algorithm. For this reason many experts suggest that the GIF format should be abandoned in favour of the new PNG format (as described below).

- The image format specified by the **Joint Photographic Experts Group (JPEG)** is ideally suited to include true colour (24-bit) photographic material inside web pages. In contrast to GIF JPEG does not support transparency. Furthermore, JPEG employs lossy compression, which is partially achieved by discarding frequency components which are not percieved by the human eye.

- The **Portable Network Graphics (PNG)** format was specified by the W3C to replace the patented GIF format. PNG posesses nearly all features of GIF (except animation) and supports additional features like increased colour resolution (32-bit) and improved transparency handling (8-bit alpha channel)[4].

8.4.12.1 Including an image with the `img` element

```
<p>This example shows how an image can be included
in HTML documents
   <img src="photo.jpg"
        alt="The photo shows me and my family">
</p>
```

The `img` element is an empty element and thus only has a start tag. The URI of the image is specified by the `src` attribute. In this example the URI is `photo.jpg` which specifies a file including a JPEG graphic. The `alt` attribute is used to specify a textual description of the image. This can be useful if the browser is not able to

4 This allows smooth transitions between background and image

display the image (e.g. because the user deactivated the rendering of images). Both, the `src` and `alt` attributes are required and may not be omitted.

The image can be scaled by employing the two attributes `height` and `width`:

```
<img src="photo.jpg" alt="description" height="100%" width="50">
```

In this example, the height is specified in proportion to the browser window height and the width is defined in terms of pixels. Hence, the image in this example fills the browser window in vertical direction and has a size of 50 pixels in horizontal direction. If the `height` and `width` attributes contain the actual size of the image, the user agent is able reserve space for it and continue rendering the document while waiting for the image data.

Height and width of an image can also be selected by applying style sheets to the `img` element. The respective CSS1 properties are displayed in Section 8.5.3.4.

8.4.12.2 Including multimedia objects with the `object` element

HTML 4.0 specifies the `object` element that is employed to hold any kind of multimedia material like images, animations, video clips, and Java applets. The object element is not well supported by the popular browsers at the moment and other elements like `applet` (depricated), `img` and `embed` (proprietary Netscape element and not HTML 4.0 compliant) are utilized to include multimedia objects. Nevertheless, the `object` element is the only official HTML 4.0 element for the inclusion of generic multimedia objects and will gain popularity in the future.

An image can be included by utilizing the `object` element as follows:

```
<p>This example shows how an image can be included
in HTML documents by utilizing the object element
   <object data="photo.jpg" type="image/jpg">
      The photo shows me and my family
   </object>
</p>
```

The data attribute specifies the URI of the object. Since an arbitrary object can be of any kind a `type` attribute specifies the content type for the data specified by the `data` attribute. In this case the content type is a JPEG image. If the browser is not able to display the specified content type, the text inside the `object` tags will be displayed.

8.4.12.3 Grouping HTML elements with `div` and `span`

In many cases it is desirable to group elements which form a semantical unit. This can be achieved with the elements `div` (block-level element) and `span` (inline element). By grouping semantically related elements the structure of the HTML document also reflects this relation. This can be useful if these structures shall be assigned a uniform presentation, e. g. by appliying style sheets (see Exercise 8.5-1)

Example 8.4-4:

```
<div class="GlossaryEntry">
  <h5>Data Encryption Standard (DES)</h5>
  <p>DES is a symmetric cryptosystem, when used for
   communications, both sender and receiver must know
   the same secret key, which is used both to encrypt
   and decrypt the message.</p>
</div>
```

In this example a glossary entry consisting of a heading and a paragraph is grouped by the `div` element. The class attribute will be further explained in Section 8.5.1.5.

Exercise 8.4-1:

The following Listing shows a glossary in SGML format

```
<glossary>
<entry id="modem">
<term>
Modem
</term>
<def>
A device that converts the digital signals produced by
terminals and computers into the analog signals that
telephone circuits are designed to carry.
</def>
</entry>

<entry id="phasemod">
<term>
Phase modulation
</term>
<def>
Phase is the position of a waveform of a signal with
respect to the origination of the carrier cycle.  Thus,
phase modulation is the process of varying the carrier
signal with respect to the origination of its cycle.
Several forms of phase modulation are used in modems,
including single- and multiple-bit phase-shift keying
(PSK) and the combination of amplitude and multiple-bit
phase-shift keying.
```

```
</def>
</entry>

<entry id="internet">
<term>
Internet
</term>
<def>
The Internet is a large data network of networks. It grew
out of the ARPAnet, which was original operated by the
U.S. Defense Advanced Research Projects Agency, and was
based on TCP/IP. The Internet still supports TCP/IP but
encompasses additional networking protocols as well.
</def>
</entry>
</glossary>
```

Write the corresponding DTD for this glossary.

Exercise 8.4-2:

Download the HTML 4.01 standard from the W3c web site:

http://www.w3.org/TR/html4/

It is available in different formats. We recommend the HTML version (the PDF version has 400 pages).

Have a look at section 3 of the HTML 4.01 specification. This section explains the SGML constructs which are relevant for understanding the HTML DTDs.

Exercise 8.4-3:

Create an HTML file with the following content locally on your computer. Display it in your browser.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN" "http://www.w3.org/TR/REC-html40/strict.d
<html>
<head>
<title>
Title of the document
</title>
</head>
<body>
Content goes here ...
</body>
</html>
```

Try out all HTML elements which are explained in this section. Use different browsers (e. g. Internet Explorer, Opera, Mozilla, Amaya) to display the document. Look up the elements in the HTML 4.01 standard.

Exercise 8.4-4:

Transform the SGML glossary into an HTML presentation by using a definition list.

- *Create a heading with the title "Glossary".*
- *Display a table of contents of all glossary entries at the top of the HTML document. Assign links to each entry in the table of contents that points to the Glossary entry.*
- *Group entries (a combination of* `dt` *and* `dd` *elements) with the* `div` *element.*

8.5 Cascading Style Sheets (CSS)

In the preceeding section basic HTML elements were presented. Most of these elements are employed to specify contents and structural information. So far, the visual formatting of contents has been neglected (except the visual formatting of tables in Section 8.4.10). It is regarded as good style to seperate the visual formatting of HTML documents from the contents. The mechanism that allows this seperation for HTML documents is called **Cascading Style Sheets (CSS)** and will be briefly introduced in this section. CSS as presented in this section is standardized by the W3C under the name **Cascading Style Sheets level 1 (CSS1)**.

8.5.1 Basic CSS

We will introduce basic CSS in a short example. Assume that a `h1` heading shall be displayed in the colour blue. This can be achieved with the following style sheet:

```
h1 { color: blue }
```

This style information can either be included in the same HTML document or in an external style sheet file (normally having the ending `*.css`).

Basically, a CSS1 style sheet is a set of rules which apply to an HTML document. Each rule consists of two parts, a **selector** and a **declaration**. In the example above which only contains a single rule `h1` is the selector and `color:blue` is the declaration.

8.5.1.1 Containment of style information inside an HTML document

The following example shows how style information can be specified inside an HTML document.

```
<html>
   <head>
      <title>Title</title>
      <style type="text/css">
         h1 { color: blue;
              border: solid }
         p { font-weight: bold}
      </style>
   </head>
   <body>
      <h1> Headline is blue </h1>
      <p>This text is bold</p>
   </body>
</html>
```

In this example style information is specified in the head of the HTML document by the `style` element. The attribute `type` indicates that the style information conforms to the CSS standard[5]. In this example all `h1` headings are blue and surrounded by a solid border and all paragraphs use a bold font.

8.5.1.2 Linking to an external style sheet

All style sheet information may be specified in an external CSS document as shown in the next example:

```
HTML dcoument:

<html>
   <head>
      <title>Title</title>
      <link rel="stylesheet" type="text/css">
            href="style.css"
   </head>
   <body>
      <h1>Headline is blue</h1>
      <p>This text is bold</p>
   </body>
</html>
```

5 Generally, any style sheet language may be used with HTML

```
CSS document with the filename "style.css":

h1 { color: blue }
p { font-weight: bold}
```

It can be seen that the external CSS file is specified utilizing the `link` element in the HTML document. The `href` attribute points to the location of the stylesheet file and the `rel` attribute indicates that the style sheet should be persistent.

Specifying style information in external files has the following benifits:

- Authors and web site managers may share style sheets across a number of documents (and sites).
- Authors may change the style sheet without requiring modifications to the HTML document.

8.5.1.3 Grouping

To reduce the size of style sheets selectors can be grouped in comma seperated lists:

```
h1, h2, h3 { font-family: helvetica }
```

In this example the three heading elements are rendered with the font `helvetica`.

Similarly, declarations can be grouped in semicolon seperated lists:

```
h1 {
   font-weight: bold;
   font-size: 18pt;
   font-family: helvetica;
}
```

This style sheet specifies that each h1 element is rendered in a bold `helvetica` font with point size 18.

8.5.1.4 Inheritance

In the first example the colour of the `h1` element was set to blue. If there is an element contained inside the `h1` element, this element will inherit the colour blue from the `h1` element:

```
<h1>This is a <em>blue</em> heading</h1>
```

In this example the `em` element will also be rendered in the colour blue because it inherits the style sheet information from its parent `h1`.

8.5.1.5 Class as selector

To allow a finer control over elements the `class` attribute can be assigned to HTML elements:

```
<h1 class="blue">This is a blue heading<h1>
```

It can be seen that the h1 element was augmented by a `class` attribute. The `class` attribute `blue` can be specified in a style sheet as follows:

```
h1.blue { color: blue }
```

By omitting the element name in the selector one can address arbitrary elements:

```
.blue { color: blue }
```

```
<h1 class="blue">This is a blue heading<h1>
<p class="blue">And this text is also blue</p>
```

8.5.1.6 Anchor pseudo-classes

Normally, hyperlinks (defined by the `a` element) in HTML documents are specially marked to make them recognizable as links. In most cases this is achieved by underlining the appropriate text and assigning the colour blue. If authors whish to change the standard way of marking hyperlinks they can do so by employing style sheets. CSS defines three states for hyperlinks:

- The *link*pseudo-class is used for a link which has not yet been visited
- Most browsers remember the links the user has visited before. These links are represented by the *visited* pseudo-class.
- The third class represents links which are currently selected, e. g. the user has clicked on the link but has not yet released the mouse button. This state is represented by the *active* pseudo-class.

Authors can assign new properties to the three pseudo-classes as shown in the following example:

Example 8.5-1:

```
a:link { color: green }
a:active { color: red }
a:visited { color: blue }
```

After assigning this style sheet to an HTML document unvisited links are rendered green, visited links are rendered blue, and links the user currently clicks on are rendered red.

8.5.1.7 Assigning style information with the `style` attribute

In case, where indvidual HTML elements must be formatted the style attribute can be employed for single HTML elements:

```
<h1 style="color: blue">This is a blue heading</h1>
```

Nevertheless, this approach should be avoided if possible, because it leads to a mix of content and presentation. The seperation of content and presentation was the reason, why style sheets were introduced in the frist place.

8.5.2 Units

This subsection gives a short overview over units which are commonly used for CSS1 properties

8.5.2.1 Colours

The easiest way to specify colours is to refer to them by their name. The following sixteen colours can be directly referred to:

```
aqua, black, blue, fuchsia, gray, green, lime, maroon,
navy, olive, purple, red, silver, teal, white, yellow
```

Other colours can be specified by using **RGB triplets**, consisting of three values which define the amount of red, green, and blue. This can be done in three different ways:

- In form of *percentages*: rgb(0%,10%,100%)
- In form of *decimal numbers*: rgb(0,127,255)
- In form of *hexadecimal numbers*: #00A1FF

8.5.2.2 Lengths

Length units can be devided into two categories:

- absolute lengths
- relative lengths

Absolute length units are:

- in - inches (1 inch = 2.54 cm)
- cm - centimeters
- mm - millimeters
- pt - points (1 pt = 1/72 in)
- pc - picas (1 pc = 12 pt = 1/6 in)

Example 8.5-3:

```
h4 { word-spacing: 4mm }
```

Relative length units are:

- em - font specific, the width of the capital ’M’
- ex- font specific, the height of the letter ’x’
- px - display specific, size of one pixel

Example 8.5-4:

`h1 { margin: 0.5em }` defines a margin having half the width of the letter ’M’

`h1 { margin: 1ex }` defines a margin having the same width as the letter ’x’

`p { font-size: 12px }` defines a font-size relative to the canvas pixels (the browser’s drawing surface)

Another form of relative length units are *percentages:*

Example 8.5-5:

```
p { line-height: 120% }
```

It is advisable to use relative lengths because they allow scalabilty of the document independent of the medium it is rendered on (e. g. screen or printer). In some cases, absolute lengths may be useful, but the general guideline is that absolute values should be avoided and relative lengths should be used instead.

8.5.3 Declarations

The following subsection gives a systematic overview over important CSS1 declarations. Due to the limited space of this lesson we are only able to provide some examples. We suggest that you look up further information on style sheet declaration yourself in the official CSS1 standard document [CSS1].

8.5.3.1 Font properties

The following table gives an overview over font properties with typical examples:

Declaration	Example
font-family	`body { font-family: Times, serif }`
font-style	`h2, h3 { font-style: italic }`
font-variant	`em { font-variant: small-caps }`
font-weight	`strong { font-weight: bolder }`
font-size	`p { font-size: 12pt }`
font	`p { font: 80% sans-serif}`

The font size can be controlled by the length properties presented in Section 8.5.2.2. There are two additional methods:

Absolut size: The size can be one of the following keywords:
`[ xx-small | x-small | small | medium | large | x-large | xx-large ]`

Relative size: `[ larger | smaller]`

For a detailed description of font properties please refer to Section 5.2 of the CSS1 standard [CSS1].

8.5.3.2 Color and background properties

CSS defines a number of properties to influence colour and background-colour of elements. Furthermore background images can be defined which can be used instead of a background colour. Table lists the different properties with examples

Declaration	Example
color	`em {color: red }`
background-color	`h2, h3 { background-color: rgb(10%,50%,20%)}`
background-image	`body { background-image: url(background.gif)}`
background-repeat	`body { background-image: url(background.gif); background-repeat: repeat-x; }`
background-attachment	`body { background-image: url(background.gif); background-repeat: repeat-x; background-attachment: fixed; }`
background-position	`body { background-image: url(background.gif); background-attachment: fixed; background-position: 100% 100%; }`
background	`p { background: url(background.gif) gray 50% repeat fixed }`

The `background` property can be used to set all properties at once. It has the following syntax:

```
<background-color>||<background-image>||<background-repeat>||
<background-attachment>||<background-position>
```

For a detailed description of colour and background properties please refer to Section 5.3 of the CSS1 standard [CSS1].

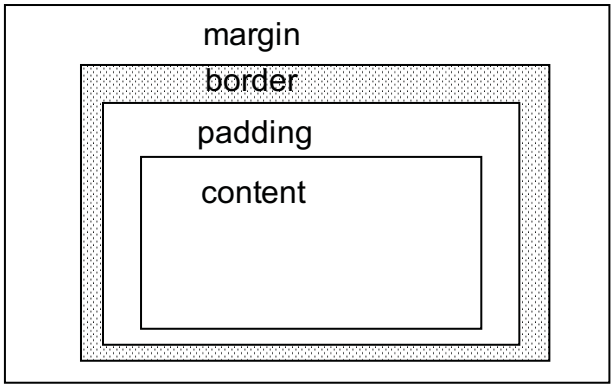
8.5.3.3 Text properties

Text properties can be used to influence parameters like word spacing, letter spacing and alignment of text.

Declaration	Example
word-spacing	`h1 {word-spacing: 0.4em }`
letter-spacing	`blockquote { letter-spacing: 0.1em }`
text-decoration	`a:link, a:visited, a:active { text-decoration: none }`
vertical-align	`img.center { vertical-align: middle }`
text-transform	`h1 { text-transform: uppercase }`
text-align	`p { text-align: center }`
text-indent	`p { text-indent: 3em }`
line-height	`p { line-height: 1.2em }`

Text properties are described in Section 5.4 of the CSS1 standard.

8.5.3.4 Box properties



Animation 8.5-1: CSS1 formatting model

CSS1 assumes a simple box-oriented formatting model where each element results in one or more rectengular boxes (see Animation 8.5-1). The box properties set the size, circumference and position of the boxes that represent elements. All boxes have a core content area with optional surrounding padding, border and margin areas.

- The padding area uses the same background as the element itself (set with the background properties)
- The colour and style for the border is set with the border properties
- The margins are always transparent, so the parent element will shine through.

declaration	example
margin-top	`h1 { margin-top: 2em }`
margin-right	`h1 { margin-right: 12% }`
margin-bottom	`h1 { margin-bottom: 2px }`
margin-left	`h1 { margin-left: 2ex }`
margin	`body { margin: 2 em }`

The `margin` property is shorthand for setting `margin-top`, `margin-right`, `margin-bottom`, and `margin-left` at the same place in the style sheet.

If four values are specified they apply to top, right, bottom and left respectively. If there is only one value, it applies to all sides, if there are two or three, the missing values are taken from the opposite side.

declaration	example
padding-top	`blockquote { padding-top: 2em }`
padding-right	`blockquote { padding-right: 12% }`
padding-bottom	`blockquote { padding-bottom: 2px }`
padding-left	`blockquote { padding-left: 2ex }`
padding	`body { padding: 1em 2 em }`

Again, the single padding property is shorthand for setting `padding-top`, `padding-right`, `padding-bottom`, and `padding-left`at the same place in the style sheet. If values are missing the same rules apply as for `margin`.

declaration	example
border-top-width	`p { border-top-width: 1em; border-style: solid; }`
border-right-width	
border-bottom-width	
border-left-width	
border-width	`p { border-width: thick; }`
border-color	`p { border-color: blue, border-width: 1em; }`
border-style	`p { border-style: none }`
border-top	`p { border-top: thick solid red }`
border-right	`p { border-right: thin double yellow }`
border-bottom	`p { border-bottom: thin dotted yellow }`
border-left	`p { border-left: 2em groove yellow }`
border	`p { border: medium dashed red }`

The `border` property is shorthand for setting the same width , colour and style on all four borders of an element.

The following properties normally apply to images but can also be used for other elements, e. g. the `div` element:

declaration	example
width	`img.icon {width: 100px }`
height	`img.icon { height: 100px }`
float	`img.icon { float: left }`
clear	`img.icon { clear: left }`

The `float` property allows to place an element at the left or right edge of the parent element. The image is taken out of the normal flow of text and is placed at one of the edges.

The `clear` property specifies if an element allows floating elements on its sides. The value of this property lists the sides where floating elements are not accepted.

8.5.3.5 Classification properties

This catagory of properties includes one property which can be used to change an element's classification according to HTML's differentiation between block-level and inline elements as described in Section 8.4.5. Furthermore, the treatment of lists and white-space (e. g. space and line feed) characters in element content can also be changed by using CSS1 properties.

declaration	example
display	`p.inline {display: inline}`
white-space	`pre { white-space: pre}`
list-style-type	`ol { list-style-type: lower-roman}`
list-style-image	`ul { list-style-image: url(bullet.png) }`
list-style-position	`ul { list-style-position: outside }`
list-style	`ul { list-style: circle inside}`

Exercise 8.5-1:

Download the CSS1 standard document from the W3C web site:

http://www.w3.org/TR/REC-CSS1

This is a single HTML page which can be saved from within the browser.

Look up detailed information in the CSS1 standard on every property you use in the following example.

Now, we will work on the presentation of the previously created HTML glossary document. Please try to implement the following features:

1. *Create a space of 4em between the left window border and all content. The document has a white background.*
2. *The links are rendered in blue and are not underlined. Visited links keep their blue color*
3. *Each glossary entry is put in a silver box with black solid border (width = 2px). The box has a width of 20em. The space between border and content is set to 1em. Between the boxes for each term there is a vertical space of 1em.*
4. *The glossary term is rendered in a bold font.*
5. *The definition of the term is rendered in the standard browser font.*
6. *Term and definition are aligned at the left side.*

8.6 Extended Markup Language (XML)

Although HTML has evolved into a a rich markup language for structuring many kinds of document types its main disadvantage is that it is not flexible enough. HTML only allows one particular document class which is simply too restricting for many of the currently emerging web applications. The benefit of XML is that it allows arbritary data structures that can be adjusted to the requirements posed by different kinds of applications.

XML is a subset of SGML (see Section 8.3). As already mentioned, SGML is too complex to be succesfully employed in the Web. For this reason, XML was designed by the W3C for ease of implementation and for interoperability with both SGML and HTML [W3C-1].

The W3C specifies the following design goals for XML [W3C-1] (excerpt):

1. XML shall be straightforwardly usable over the Internet.
2. XML shall support a wide variety of applications.
3. XML shall be compatible with SGML.
4. It shall be easy to write programmes which process XML documents.
5. XML documents should be human-legible and reasonably clear.
6. XML documents shall be easy to create.

8.6.1 Differences between XML/XHTML and SGML/HTML

Most of the differences between SGML and XML are beyond the scope of this lecture. One major difference between SGML (including HTML) and XML is that rules regarding markup are more strict. XML does not allow the minimization of markup. For example, XML does not permit empty elements like the following HTML 4.0 construct:

```
<img href=... alt=...>
```

Instead, empty XML elements must be closed as follows:

```
<img href=... alt=.../>
```

This would be the correct XHTML [W3C-2] (see also Section 8.4.1) syntax of the `img` element. Furthermore, in XML ending tags may not be omitted like in the following HTML example:

```
<p>This is a paragraph.
<p>This is another paragraph
```

The respective XHTML code must include the ending tags:

```
<p>This is a paragraph.</p>
<p>This is another paragraph</p>
```

Another difference between HTML and XHTML is that in XHTML all tag names must be written in lowercase. HTML allows both, uppercase and lowercase writing for tag names.

Although this course text uses HTML 4.0 as basis, we did not make use of the loose HTML 4.0 syntax to stay compatible with the new XHTML standard. For this reason we have employed lowercase writing and ending tags so far.

In contrast to SGML, XML documents may be written without an associated DTD. XML documents which comply to the markup rules but have no DTD associated with them are referred to as being **well-formed**. A well-formed XML document is one that is syntactically correct, whether or not is has been checked against a DTD.

A **valid** XML document is a well-formed XML document that also has been validated against a DTD. The validation process is carried out by an XML parser which reads the file containing the XML markup and checks if it complies to all rules specified in the associated DTD.

8.6.2 XML Application Areas

Typical applications for XML in the Internet environment are [Mar99]:

- Use of XML to describe *metacontent* regarding documents or online resources.
- Use of XML to *publish and exchange* different kinds of *content*.
- Use of XML as a *messaging format* for communication between application programmes.

8.6.2.1 Metacontent

Metacontent is information about a document's contents, such as its title, author, creation date, and so on. Metacontent can be used, for example, for searching, information filtering, and document management.

Although HTML specifies the `meta` element, this element is not powerful enough to support complex search requests. Furthermore, the `meta` element is specified inside the HTML document so that search engines cannot refer to the information without downloading the entire HTML file. It is, for example, not efficient to download every HTML file from a web server to check if it inlcudes a special keyword. If, on the other hand, all metacontent is collected in one single file, the search process becomes much more efficient.

XML is considered the best vehicle for such external metacontent because of its extensibility, flexibility, and readability.

A standardized XML application for the specification of metacontent is the **Resource Description Framework (RDF)**[W3C-4].

8.6.2.2 Publishing Content

The popularity of SGML in the publishing industry has already been mentioned in Section 8.3. XML inherits all the advantages of SGML with regard to separation of content and presentation, while beeing easier to use than SGML. The emerging diversity of web enabled devices like standard PCs, personal digital assistants, WAP mobile phones, and TV sets with Internet functionalties requires a new approach to the design of web content. Since all these different devices have dissimilar display properties (e. g. screen and colour resolution), there is a need for a mechanism that flexibly adapts the content for each device (see Fig. 8.6-1).

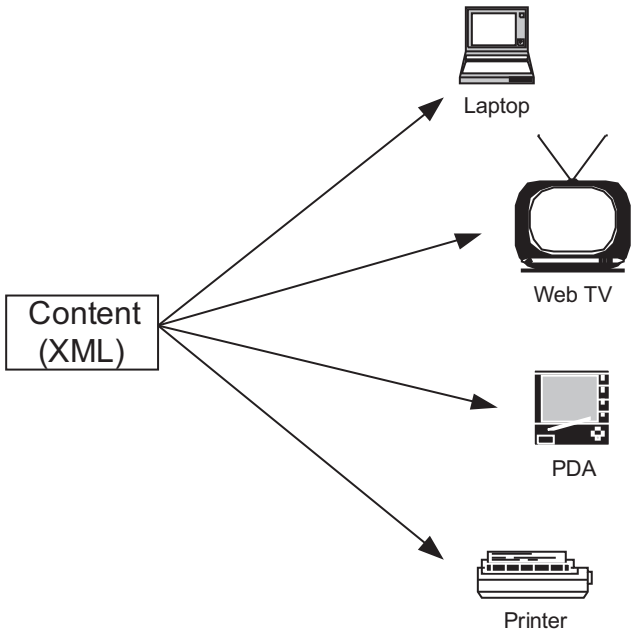


Fig. 8.6-1: XML publishing

This can be achieved by defining all content in XML and employing style sheets to adapt the content to the display device. In Section 8.5 we have already shown how CSS can be employed to customize the presentation of HTML pages. CSS can be used for XML content the same way.

In the following listing there is a short excerpt from a glossary in XML format .

```
<glossary>
<entry id="modem">
<term>
Modem
</term>
<def>
A device that converts the digital signals produced by
terminals and computers into the analog signals that
telephone circuits are designed to carry.
</def>
</entry>
</glossary>
```

You can see that it is identical with the SGML format from Exercise 8.4-1. In fact, XML was defined in a way that every XML document is also a valid SGML document. It is also obvious, that the semantics of the XML document are more clear than the semantics of the HTML document of Exercise 8.4-4.

A CSS style sheet can be directly employed as shown in the following.

```
glossary { display: block;
           margin-left: 4em;
           background-color: white;
         }
entry {  display: block;
          background-color: silver;
          margin: 1em 0em;
          border: black solid 2px;
          width: 20em;
          padding: 1em;
        }
term { display: block;
       font-weight: bold;
       margin: 0em;
      }
def { display: block;
      margin: 0em; }
```

The content can be adapted to different devices by creating a specific style sheet for each device. Nevertheless, in many cases CSS is not powerful enough. For this reason the W3C has defined the **Extensible Style Language (XSL)** for the use with XML.

In essence, XSL is two languages, not one. The first language is a transformation language, the second a formatting language. The transformation language **XSLT (XSL Transformations)** is useful to move XML data from one XML representation to another. XSLT provides elements that define rules for how one XML document is transformed into another. For example, XSLT can be employed to transform XML electronic commerce data specific to one corporation into an XML format specific to another corporation. XSLT can also be used to transform XML into (X)HTML.[6]

The second part of XSL is the **XSL formatting objects** which still is in draft status (autumn 2000). The formatting language describes how content should be rendered when presented to a reader. XSL formatting objects provide a more sophisticated visual layout model than HTML + CSS. Nevertheless, at the moment there are no browsers available that support XML + XSL formatting objects (Internet Explorer 5.0 supports an early version of XSLT).

6 The HTML and PDF versions of this course were created by transforming XML output from an authoring tool (Framemaker) with XSLT.

8.6.2.3 Messaging

Messaging is the exchange of messages between organizations or between application systems within an organization. The traditional format for messaging between enterprises is the **Electronic Data Interchange (EDI)** format. In it s long history, EDI has greatly contributed to automating business-to-business (B2B) transactions. Nevertheless, not all corporations can afford EDI systems because of the high costs for building and operating them. B2B messaging employing XML (Internet EDI) can be much more cost efficient and easier to implement. The required security mechanisms can be realized by employing standard cryptographic protocols like Secure Socket Layer (SSL) and formats like S/MIME (Secure MIME).

Exercise 8.6-1:

Write an XML representation of the EDI message below. It is a message for a book order (taken from [Mar99]).

```
UNH 000002+ORDERS;DD 96A UN:EAN008          Header
BGM+220_B00002-9'                           Order Number
DTM-137-19940202:102'                       Message Date
NAD+BY Stadt- und Universitaetsbibliothek :  Buyer name
   Frankfurt-Bockenheimer Landstr. 134-13    and address
   8+Frankfurt++60325'
RFF+API:DE1141110388'                       Buyer ID number
NAD+SU+++DREIER'                            Supplier name
CUX+2:DEM:9'                                 Order Currency
```

8.7 Hypertext Transfer Prototol (HTTP)

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems and has been in use by the World Wide Web since 1990. The first version HTTP/0.9, was a simple protocol for raw data transfer accross the Internet. HTTP/1.0, as defined by `RFC 1945`, improved the protocol by allowing messages to be in the format of MIME-like messages (see Section 7.3.2), containing meta-information about the data transfered and modifiers on the request/response semantics. The most recent version of HTTP is HTTP/1.1 and introduces important concepts like proxies, caching, persistant connections, and virtual hosts. HTTP/1.1 is defined in `RFC 2068`.

HTTP is a request/response protocol. A client sends a **request** to the server which replies with a **response**.

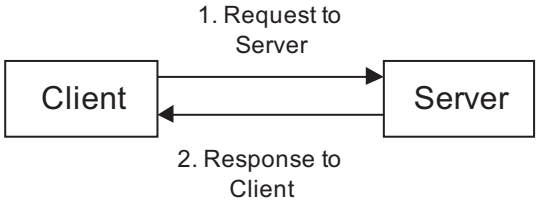


Fig. 8.7-1: HTTP request and response

HTTP communication usually takes place over TCP/IP connections. The default port is TCP 80, but other ports can be used. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used. A connection may be utilized for one or more request/response exchanges.

The following basic roles are important for HTTP interaction:

- **Client** - A client is programme that establishes connections for the purpose of sending requests. Typically, a client will be a WWW browser, but it may also be a search engine or some other sort of programme.
- **Server** - Any programme accepting connections in order to service requests by sending back responses is a server. A server has to interpret and understand the contents of a request. The typical HTTP server is a **web server** (WWW server) which stores web documents like web pages, graphics and multimedia files.
- **Proxy** - A proxy is an intermediary programme which acts as both a server and a client, receiving a request and then acting as a client and making requests on behalf of other clients. A popular application for a proxy server is to cache the most frequently demanded web pages at an ISP's POP. The requests from the user's browser are first directed to the proxy to look up if the demanded page resides in the proxy's **cache**(memory for temporary data). If the web page is not present the proxy redirects the request to the origin web server. This methods frees the Internet backbones from unnecessary traffic. For this method to work, the proxy's address must be explicitely configured in the user's browser.

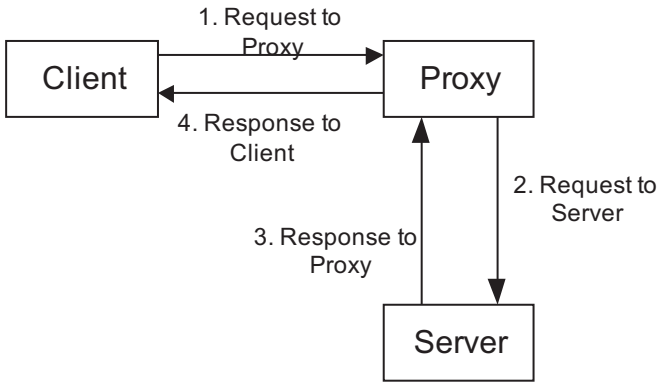


Fig. 8.7-2: HTTP Proxy

8.7.1 HTTP messages

HTTP messages consist of requests from client to server and responses from server to client. Request and response messages use the generic message format of RFC 822 for transferring entities (the payload of the message). Both types of messages consist of a start-line, zero or more header fields (headers), an empty line indicating the end of the header fields, and an optional message-body.

```
generic-message =
   start-line
   *message-header
   CRLF
   [ message-body ]



start-line =
   request-line | status-line
```

The start line of a message consists of a request line, if the message is a request message, or a status line, if the message is a response message.

The header fields can be grouped into four different catagories:

- **General headers** apply to both request and response messages, and they do not apply to the entity being transferred.
- If the entity being transferred by a request or response needs to be described by some meta information, this can be done by using **entity headers** in the message being sent. If no message body is present, the information given in the entity headers describes the resource identified by the request.
- The purpose of **request headers** is to allow the client to pass information about the request, and about the client itself to the server. They do not contain any information about the message body (i.e. the entity being transferred).
- **Response headers** are used by the server to pass any information which can not be given in the status line. They do not contain any information about the message body.

8.7.2 HTTP request

An HTTP request has the following format:

```
request =
   request-line
   *(general-header|request-header|entity-header)
   CRLF
   [ message-body ]


request-line =
   method SP request-URI SP HTTP-version CRLF
```

The HTTP request starts with a request line which contains the most important information of the request. The request line is followed by zero or more headers, which can be general headers, request headers, or entity headers. Seperated by a blank line (carriage return + line feed), an optional message-body can be part of the request message. The request-line contains three fields seperated by space characters:

- The method field specifies the method to be performed by the server on the resource identified by the request-URI
- The request-URI is a URI as described in Section 8.2.1 and identifies the resource upon which to apply the request. The absolute form -containing the host name - must be used when sending a request to a proxy. If the request is sent directly to the origin server, the path form (the url-path part) can be used, which uniquely identifies the resource on the server.
- The HTTP-version field indicates the version of an HTTP message. Any application sending a request message as defined by the HTTP/1.1 specififcation must include a string indicating an HTTP-version of HTTP/1.1.

In the following, some popular HTTP/1.1 request methods are presented:

- **GET** requests a resource. Normally this would be a document or any other kind of data stored on the server.
- A **HEAD** request looks very much like a GET, except the server should return only the HTTP headers, not the item. This method is often used for testing URIs for validity, accessibility, and recent modification.
- **POST** is used to send a block of data to the server. This might be form data or some other application-specific information. The URL identifies a function that will process the block of data. In web applications the function is often part of a CGI script or a java servlet.
- The **PUT** method can be used by a client to store an entity on a server under a particular URI. A typical application is the upload of web pages belonging to a web site which is hosted by an ISP.

- **DELETE** asks the server to delete the resource named in the URI.

A typical request from a web browser to a web server may look as follows:

```
GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Accept-Language: de
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98)
Host: licher.fernuni-hagen.de
Connection: Keep-Alive
```

The first line in this example consists of the request method sent by a Microsoft Internet Explorer. The browser uses the GET method to request the homepage (indicated by the URI /) from the WWW server. In the second part of the first line the browser indicates that it understands HTTP/1.1.

The client's **accept headers** indicate:

- The client is willing and able to receive a variety of image datatypes.
- The user's language is German.
- The client is willing to receive information that has been compressed using *gzip* or *deflate*.

The client identifies the browser product and version in the **User-Agent header**. In this example it is a Microsoft Internet Explorer 5.0 running on a Windows 98 machine.

The **Host-Header** identifies the server that was named in the request URL. Without this field, the server would not get to see the host name. This is important because a web server can run several **virtual hosts**. It is very common for Web sites to be outsourced to a service provider. The provider can load a company's web site onto a computer which is shared with many other companies. In such a configuration, each web site is represented by a virtual host.

Example 8.7-1:
An ISP runs three virtual hosts (web sites) on the same web server: www.CompanyA.com, www.CompanyB.com, and www.CompanyC.com. A DNS server resolves all three names into the same IP address, which is the IP address of the host running the web server. A browser connects to the web server (port 80) and uses `GET / HTTP/1.1` to request the homepage of Company A. The web server does not know from the GET method which company's home page the browser wants to access. So, it looks up the host header to find out that the browser wants to access the web site of Company A. Now the web server can load the requested HTML files from its hard disks and serve them to the browser.

The **Connection-Header** allows the sender (either client or server) to specify options which should be applied to a particular connection, which means that it must

not be communicated by proxies or further connections. In this example the client indicates that the connection (in most cases TCP/IP) should be kept alive after the response has been delivered. Hence the connection can be used for succeeding requests and responses. If the client wants close the connection after receiving the response it employs the following header:

```
Connection: close
```

The concept of utilizing a connection for multiple request/response pairs is referred to as **persistent connection** (see Fig. 8.7-3). In HTTP/1.1 persistent connections are the default.

Persistent connections have the following advantages:

- Saving of operating system resources like CPU time because less connection setup and tear-down are necessary.
- It is possible to send multiple requests on one connection without having to wait for the first response before sending the second request. This mechanism is referred to as **pipelining** (see Fig. 8.7-4).
- Since there is less connection setup and tear-down, fewer packets are sent over the network. Since the TCP connection setup is rather complicated using a three-way handshake mechanism (see Section 4.4.4), less connection setups lead to a considerable reduction of packets sent between a client and a server.

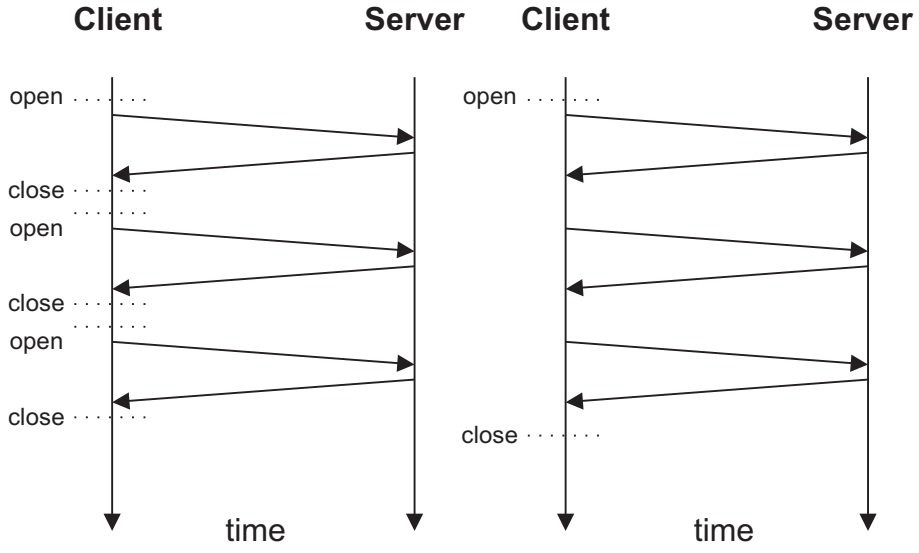


Fig. 8.7-3: HTTP persistent connection

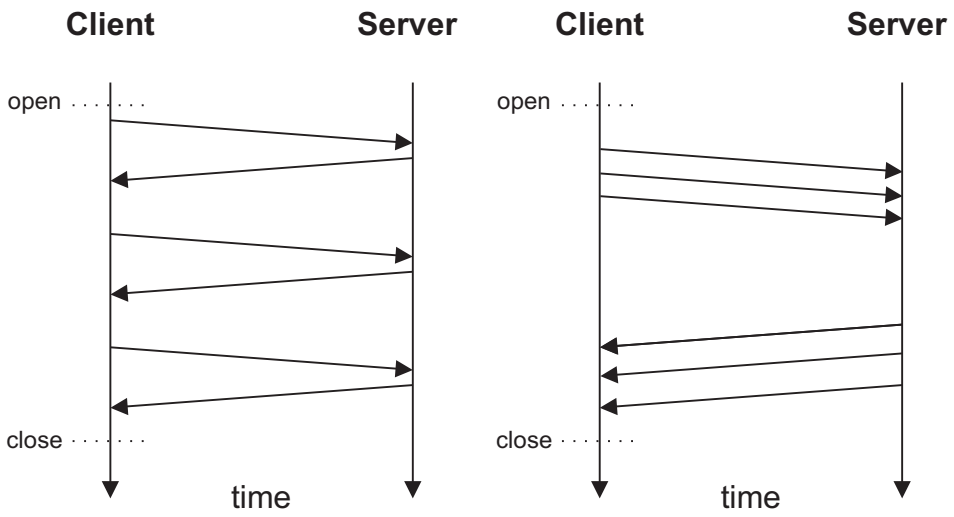


Fig. 8.7-4: HTTP pipelining

Exercise 8.7-1:

You can display a request from your browser by connecting to the following URI:
`http://www.rewebber.de/show_request/http://www.fernuni-hagen.de`

8.7.3 HTTP response

An HTTP response is always the second message in an HTTP interaction and has the following format:

```
request =
   status-line
   *(general-header|response-header|entity-header)
   CRLF
   [ message-body ]



status-line =
   HTTP-version SP satus-code SP reason-phrase CRLF
```

The status line contains the most important information of the response and indicates if the request was successful.

The status code is a 3-digit integer result code of the attempt to understand and satisfy the request. The general code assignments are displayed in Table 8.7-1. The reason phrase gives a short textual description of the status code and is aimed at human users.

Tab. 8.7-1: *Status Codes*

Code	Meaning
1xx	Informational
2xx	Success. The action was succesfully received, understood, and accepted
3xx	Redirection. Further action must be taken in order to complete the request.
4xx	Client Error. The request contains bad syntax or cannot be fulfilled.
5xx	Server Error. The server failed to fulfill an apparently valid request.

A typical response message from a web server:

```
HTTP/1.1 200 OK
Date: Fri, 09 Jun 2000 09:47:54 GMT
Server: Apache/1.2.0
Last-Modified: Wed, 07 Jun 2000 13:26:14 GMT
ETag: "3531b8-5d-39db2ff6"
Content-Length: 93
Accept-Ranges: bytes
Content-Type: text/html

<html>
<head>
<title>Hello World</title>
</head>
<body>
Hello World
</body>
</html>
```

The status line indicates that the server speaks HTTP/1.1 and informs that the request was understood and processed.

The **Date header** simply gives the date and exact time of the response message.

The **Server header** identifies the server product, just as the client did. In this case the product is an Apache web server version 1.2.0.

The **Last-Modified header** is a field which should be included in a response whenever possible. It indicates the date and time at which the server believes the entity was last modified. Depending on the type of the entity, this may be a modification date from a file system, a time stamp from a database, or the current date and time, if the entity is dynamically generated from data which changes constantly.

The **ETag header** contains a validator for the requested resource. It can be used in conjunction with the Last Modified header to identify if the resource has changed. This can be achieved by associating a unique ETag (validator) to each version of the

resource. The client can compare the ETag with a cached version from a previous response. If the ETag in the current response is different from the cached version, the resource has changed.

The **Content-Length header** signals the length of the message body in bytes.

The **Accept-Ranges header** indicates that the server is willing to accept requests that ask for parts of resources by specifying one or more ranges of bytes.

The **Content-Type header** specifies the media type of the resource in the message body. In this is example the resource is an item of type text and subtype html.

9 Usenet

9.1 Goal of the Chapter

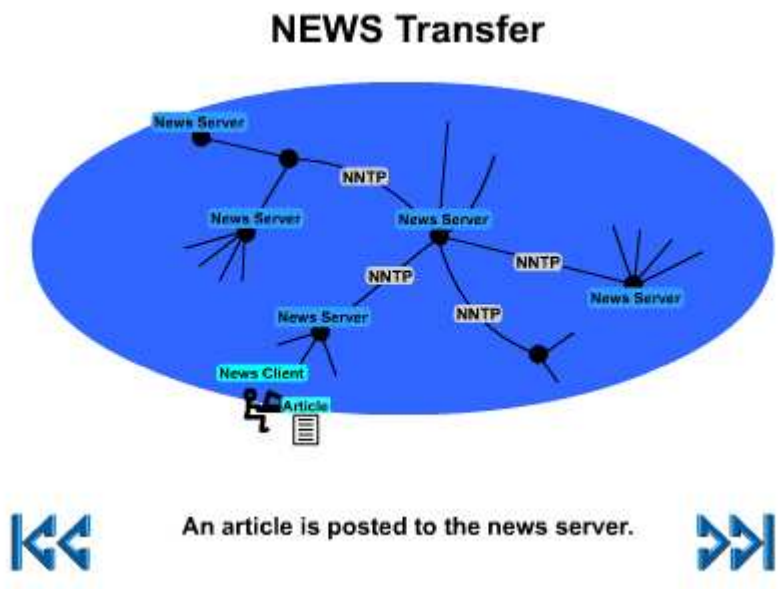
The goal of this chapter is to get an understanding of the mechanisms of **Usenet News**. Usenet News are considered with respect to the user's view, to technical aspects, and to the usage of Usenet News.

9.2 What is Usenet?

The Terms "Newsgroups", "Usenet News", and "Net News" are synonymous. "Net News" is the oldest denotation, , which was replaced by "Usenet (News)", resulting of the days, the traffic was transported by the uucp program from one Unix machine to another. Another explanation is "Unix USEr's NETwork".

The Difference between Usenet and Internet is on a content, not on a technical level.

Usenet is a world-wide distributed discussion system. It consists of a set of "newsgroups" with names that are classified hierarchically by subject (topic). "Articles" are "posted" to these newsgroups by people on computers with the appropriate software. Articles are similiar to email messages in structure and type of transport mechanisms, but handled by different instances on a computer. Articles are intended for public discussions rather than personal communication and are broadcast to other interconnected computer systems via a wide variety of networks. Some newsgroups are "moderated"; in these newsgroups, the articles are first sent to a moderator for approval before appearing in the newsgroup. Usenet is available on a wide variety of computer systems and networks, but the bulk of modern Usenet traffic is transported over either the Internet (NNTP) or UUCP.



Animation 9.2-1: Usenet

Formally, the Usenet is a logical network as it has a structure but physically its data is propagated from one location to another via various different networks, such as the Internet, BBSs (Bulletin Board Systems) and others.

Practically, the Internet adopted Usenet, and browsing Usenet news is made within the Internet via Internet protocols. Still, one may have access to Usenet without having access to the Internet.

9.3 Usenet History

In the late 1970s, Unix developers came up with a new feature: a system to allow Unix computers to exchange data over phone lines.

In 1979, two graduate students at Duke University in North Carolina, Tom Truscott and Jim Ellis, came up with the idea of using this system, known as **UUCP** (for **Unix-to-Unix CoPy**), to distribute information of interest to people in the Unix community. Along with Steve Bellovin, a graduate student at the University of North Carolina and Steve Daniel, they wrote conferencing software and linked together computers at their universities.

Word quickly spread and by 1981, a graduate student at Berkeley, Mark Horton and a nearby high school student, Matt Glickman, had released a new version that added more features and was able to handle larger volumes of postings - the original North Carolina program was meant for only a few articles in a newsgroup each day.

The software has been adapted to the increasing number of news articles.

Today, Usenet connects hundred of thousands of sites around the world offering more than 50.000 newsgroups.

9.4 Structure and Organisation of Usenet

Usenet is a forum for public network messages. The messages are sent by people who have posting access to Usenet, and they are divided into different newsgroups by a hierarchy, according to their subject. Newsgroup names generally contain two or more parts, separated by dots. As one can read from left to right, the various parts of the name progressively specialise the topic of the discussion. The first part of the name indicates the top-level hierarchy to which the newsgroup belongs. Based on the historical evolution of Usenet there is a seven standard top level hierarchy, called "**Big Seven**". These are the first seven top level that were established (see Table 9.4-1 and the next section for explanation).

Tab. 9.4-1: *"Big Seven" top level hierarchy*

name	topic
comp.	discussions about computer hardware and software
talk.	general discussions about controversial topics
news.	discussions about Usenet like administration and creation of new newsgroups
soc.	social and cultural discussions (ethnics, religous, etc.)
sci.	science-related topics
rec.	"recreational": sports, hobbies, food, music, etc.
misc.	discussions, which don't fit anywhere else

In addition there are many more national newsgroups. They start with the international **TLDcc** ("**Top Level Domain Country Code**"), e.g. "de." for Germany, "nl." for the Netherlands, or "eunet." for europe wide topics. These newsgroups are are intended mainly for distribution within limited geographical areas or within single institutions (like the "feu." newsgroups of the FernUniversität).

Furthermore mappings from other networks and bulletin boards like "z-netz." (Zerberus-Netz), "fido." (Fid net), or "maus." (Maus-Netz) can be found. Articles posted into these newsgroups are vice versa transported to the other networks.

Finally, there are various special-purpose hierarchies which are not distributed as widely as the "Big Seven" and alt. Some of these focus on specialised fields, for example, "bionet." for biology or "school." for discussions mainly about school releated topics and mostly used by pupils.

To clarify the diversity of newsgroups see Table 9.4-2 for a small selection of newsgroups available at the FernUniversität's news server:

Tab. 9.4-2: *Newsgroup selection*

Name	Explanation
comp.ai.fuzzy	Artifical intelligence
comp.compression	Compression algorithms
comp.protocols.misc	Here you can find more information about the protocols explained in this course
de.markt.buecher	German newsgroup about selling and buying books
de.org.mensa	Clever people
feu.kontakt, feu.cafe	To meet fellow students
misc.fitness.misc	Mens sana in corpore sano!
sci.crypt	Cryptology related discussions
soc.history.science	History of Science
alt.music.abba	About a no more existing swedish pop group

9.4.1 Structure of newsgroup names

As mentioned in the last section, the name of a newsgroup specifies the topic the discussions in it are about. Like an email address, the newsgroup name goes from general to specific (but in contradiction to email addressing from the the left to the right).

Example 9.4-1:
`de.comp.office-pakete.staroffice`

This newsgroup is German speaking ("de.). The main topic are computers ("comp"). On the same level of hierarchy are main topics like "lang." (languages) or "os." (operating systems like Unix, Mac OS, or OS/2).

Further, the fragment "office-pakete." describes the subgroup which handles with software for offices (like wordprocessors, spread sheets, etc.).

Last, "staroffice" denominates the actual topic StarOffice, an integrated office application.

In general, newsgroups are text oriented. This property results from the historical roots; in former Usenet times, the nodes were connected via small bandwidth data paths (analogue telephone connections typically). It was not possible to transport thousands of articles containing binary data (like graphics, software, etc.) every day using such connections.

This attitude has changed with upcoming high speed networks. There is now the possibility to fulfill the need of so called high traffic and high volume newsgroups. An indicator for these newsgroups is the word "binaries" somewhere in the name (like "de.comp.linux.binaries"). Because of the high volume, not every server is hosting these newsgroups.

9.4.2 Moderated Newsgroups

The fact, that every user subscribed to a newsgroup can post arcticles, suggestive or senseless, is a reason, that some newsgroups insist that the discussion remains focused on the proper topic. Therefore moderated newsgroups have been introduced.

Articles posted in such a moderated newsgroup are mailed to the group's moderator. He reviews the arcticles and decides whether to forward them to the actual newsgroup or not. Because of this approach articles appear delayed in the newsgroup.

9.5 User's View to Usenet

In general, every internet providers offers access to Usenet. The source of the newsgroups is a so called **news server**, wich holds the news. This server is also called **news feed** for his function of feeding Usenet articles to users and other news servers.

A user, who wants to access newsgroups needs permission to subscribe to newsgroups and to read at least. If he wants to post articles he also needs the right to write or to send the articles to the news server, respectiveley.

Not all newsgroups are provided by every internet service provider, the number of groups offered depends on the hard disk capacity and bandwith a service provider is willing to invest.

The exact number of all newsgroups is unknown and hard to define because of newsgroups only locally available or because of restriced distribution. Assumptions go from 30.000 to 100.000 newsgroups worldwide. The number of the Big Seven hierarchy newsgroups averages between 20.000 and 30.000 newsgroups.

9.5.1 Newsreader

Like using email, the user who wants to access newsgroups, needs a software tool, here a so called news reader. Like email clients a multitude of such clients are available. Independent of their user interface (textual or graphical), they offer a certain number of base functions and a more or less equal number of extented features.

The two main functions of news readers are to manage the subscription of newsgroups and to handle the reading and posting of articles (see below, Section 9.5.2 and Section 9.5.4).

The user needs to have an active internet connection to use the Usenet. Because most of the time using the Usenet is spent in reading and writing articles money is wasted if he has to pay for the internet connection. As a solution, offline newsreader only connect to the internet, while data is transmitted.

In a short online phase, the news reader contacts the news server, transfers the subject header (identical to email subject headers) of the articles of the newsgroups the user has subscribed. Further, articles the user has composed are posted to the corresponding newsgroups. Then the internet connection, which is often established via a telephone line, can be dropped.

The user can now select the arcticles he is interested in arbitrarily time.

In a second online phase the marked articles are downloaded from the news server.

For the communication details of the dialog between news server and news reader see Section 9.6.4.

9.5.2 Threading

One of the terms used when dealing with newsreaders that causes the most trouble with new Usenet users is the term **thread**. Newsreaders often give the option of "threading" articles and people often refer to "threads" within a newsgroup. A thread is simply a group of articles in a newsgroup that each have the same subject, in which one of the articles is the original post and the rest are **followups** (replies to the newsgroup). Sometimes a thread can last so long that the original posting is no longer on the news server and nothing but followups exist.

Keeping the definition of a thread in mind, the difference between threading articles or not within a newsreader is this: threading articles makes all articles with the same subject appear together, under the original post; leaving the articles unthreaded will result in articles being sorted strictly by the selected method (date, subject, etc.), regardless of whether or not they have the same subject. In most cases, threading is the way to go. This makes it easier to follow a thread.

9.5.3 Subscribing and Unsubscribing to Newsgroups and Reading of Usenet Articles

When using a news reader for the first time, the user prompts the news server for an inital download of all newsgroups available. Following to this, the user can select (mark) the newsgroups he is interested in.

If the user does not want to read the arcticles of a newsgroup further, he signals this to the news readber by unsubscribing the newsgroup.

After done so, the news reader downloads the subject headers of hte existing arcticles of the marked newsgroups and displays them. By selecting a subject header the news reader load the corresponding article and presents it to the user.

The availability of Usenet articles depends on the configuration of the news server (and therefore on the Usenet policy of the news server administrator). Depending on the size and number of daily messages and the available disc capacity of the news server, the articles are stored for some days until several weeks.

9.5.4 Posting of Usenet Articles

In this section the main user actions are described: Posting articles as a response to an existing article, starting a new sequence of articles and the filtering of articles of unwanted other users.

9.5.4.1 Beginning a new thread

There are two basic types of articles that can be posted to Usenet: an original post and a followup. An **original posting** is simply an article posted to one or more newsgroups that has an original subject and can therefore become a thread. A followup is a reply to an original posting that is send to the same newsgroup(s) as the original posting. A **followup** is usually a response to the original posting (as in an answer to a question), but Usenet threads sometimes last so long that the original posting is forgotten and nothing but followups exist. Some newsreaders call followups "replies," but a reply is usually considered as a personal response via email, not a public posting to Usenet. Sometimes, users post a followup message in addition to a personal email reply.

9.5.4.2 Crossposting

When posting an article to Usenet, you have the option of posting it to one or more newsgroups. If an article is posted to more than one newsgroup, it is called **crossposting**. A newsreader can be configured to crosspost by simply placing a comma between the names of each newsgroup the article shall be posted to.

Example 9.5-1:
Newsgroups:
`feu.ice-bachelor.kurs.20018.diskussion,feu.urz.beratung`

Crossposting can be useful if an article shall be send to more than one group that would be of interest to readers of each newsgroup (who might not necessarily read each group).

If a suitable group can not be found, one should never post to an arbitrary sample of groups; that would be very unproductive, and the message might even be considered as **spam**[URL], against which serious measures can be taken.

9.5.4.3 Killfile

Most Usenet newsgroups are not moderated; the Usenet appears as a medium filled with annoying advertisement or stupid questions. This can be easily fixed, however, with the use of a **kill file**. A kill file is simply a file (usually kept in the same directory as the newsreader) that contains strings of information that tell the newsreader what articles in which newsgroups to erase automatically. The newsreader, with the help of a kill file, can sort through all incoming Usenet articlesautomatically and choose not to display ones that look like "**junk mail**."

Most newsreaders allow the user to "program" their kill files using easy-to-use menus or dialogue boxes. The user can usually choose to delete messages with certain phrase(s) in the "Subject:" line or in the "From:" line. In other words, a kill file can be used to eliminate messages with subjects containing one or more words, specified by the user, or messages from a specific individual (whose e-mail address you enter in your kill file).

Many newsreaders allow these "kill strings" to be entered by newsgroup, or globally. In other words, if all articles from `nag@students.fernuni-hagen.de` shall be deleted from the `feu.ice-bachelor.kurs.20018.diskussion` newsgroup, but not from any other newsgroup.

9.6 Technical Aspects

9.6.1 Basics

Each computer acting as a news server (and belonging to organisations like universities or companies) stores incoming news articles locally. There are directories and subdirectories corresponding to each newsgroup name. At the news server hosting this course’s newsgroups, the articles are stored in a directory ending with the path "news/ice-bachelor/kurs/20018/diskussion". The news server responding to news reader request for one or more articles of a newsgroup accesses the corresponding directory and delivers the articles. The advantage of this kind of storing is that only one instance of an article has to be retained, regardless the number of the users of the corresponding newsgroup.

Articles are stored on the news server until their number, size, or age exceeds a limit given by the administrator.

Seen topologically, the Usenet is a directed graph of news server (nodes), connected by communication lines (edges). Each news server queries its news feed periodically. New articles are transported from the news feed to the news server and saved to the corresponding folder in the directory structure explained above. This kind of inquiry is also called "**pull method**". If the news feed is initiating the transfer of news articles, this is called "**pushing**". With "pulling" and "pushing" news articles are distributed across the Usenet.

9.6.2 Anatomy of an Usenet Article

In Section 7.3 we have analysed the structure of email messages as specified in RFC 822 [RFC822]. Usenet articles are built the same way, their structure and format is equivalent to a valid Internet message according to the RFC 822, but is extended by several additional header fields. This property simplifies the transmission and makes it possible to use most of the standard email software. The additional headers are explained in the RFC 1036, "Standard for Interchange of Usenet Messages" [RFC1036].

Required headers are "From", "Date", "Newsgroup", "Subject", "Message-ID", and "Path", see the RFC for the optional header fields. Any other, unrecognised headers are allowed, but will be passed through unchanged.

The following listing holds a sample article from the newsgroup "sci.crypt", a group which handles with cryprological topics:

```
Path: oak.fernuni-hagen.de!news-koe1.dfn.de!news-
fra1.dfn.de!news-fra.pop.de!news.csl-gmbh.net!
newsfeed01.sul.t-online.de!newsmm00.sul.t-online.com!
t-online.de!news.t-online.com!not-for-mail
From: Mok-Kong Shen <mok-kong.shen@t-online.de>
Newsgroups: sci.crypt
Subject: Re: Whole file encryption
Date: Wed, 08 Nov 2000 10:39:36 +0100
Organization: T-Online
Lines: 65
Message-ID: <3A091F58.FC2F6E59@t-online.de>
References: <3A08EBF9.73B8741E@earthlink.net>
Mime-Version: 1.0
X-Complaints-To: abuse@t-online.com
X-Sender: 320032439720-0001@t-dialin.net
X-Mailer: Mozilla 4.51 [de]C-CCK-MCD DT (Win98; I)
Xref: oak.fernuni-hagen.de sci.crypt:88680

Benjamin Goldberg wrote:
> The following is a simple idea for whole file encryption.
> sbox is actually a keyed sbox.
>

If I understand correctly, what you do in encrypt_r
is equivalent to doing a permutation of the elements
of the array data...
[deleted].

M. K. Shen
```

Like in email messages, the headers in this article come before the body of the article, and are identified by the colon following them. The required headers are:

1. Path: From right to left, this line shows how the article managed to get to the local machine. In this case, the article was created by sending it to the news server news.t-online.com and was passed from this machine through the machines news-koe1.dfn.de! news-fra1.dfn.de!news-fra.pop.de!news.csl-gmbh.net!newsfeed01.sul.t-online.de!newsmm00.sul.t-online.com!t-online.de, and finally to the news server oak.fernuni-hagen.de, from where it was requested.
2. From: Tells, who sent the message, consisting of the person's email address followed by the real name in parentheses (optional).
3. Newsgroups: The newsgroups header for this article, says that this article should be directed to the newsgroup "sci.crypt".
4. Subject: The subject of the article. The "Re:" Identifies this article as a follow-up to an earlier article.
5. Date: This is the date that the message was sent. It may take a few days for an article to propagate across Usenet, depending on how the poster's computer is connected to its neighbours.
6. Organization: The organisation (institution, company) the sender belongs to.
7. Lines: Number of lines of the article.
8. Message-ID: This is a line giving an article ID number to this particular article. Of the form: `unique_string@host.domain`. The combination of unique_string and hostname uniquely identifies this article throughout the Usenet.
9. References: Shows the unique Messsage-ID of the article to which the actual one is a follow-up to. The purpose of "References:" headers is to allow messages to be grouped into conversations by the news reader.

The other fields are already known by email headers (Section 7.3) or will be subject of an assignment, respectively.

Finally we come to the article text itself. Lines which begin with ">" are "quoted" from the article that this one is following up. There is usually an attribution line which indicates who wrote the quoted text. These are inserted automatically when you reply to an article. Notice that there can be also lines which begin with ">>". These were quoted in the article which is being quoted. Occasionally one can see four, five, or even more levels of quoting, but many people consider this to be bad form.

At the end of the article, separated from it by two dashes and a space, there is often a signature, which may contain the author's name, email address(es), mail address, phone number, etc. Many newsreader programs allow to create a "**signature file**" which is automatically appended to the end of each article.

9.6.3 Control Messages

A news administrator (newsmaster) can manipulate Usenet articles and newsgroups by means of control messages. Control messages are special articles posted to the control newsgroup containing commands that instruct news servers to perform specified tasks.

They are recognised by the presence of a "Control:" field in the article header, which contains the name of the control operation to be performed. Most of these messages perform their action automatically at the time the article is processed by the news server without notifying the news administrator.

1. The cancel Message

   The most widely known message is cancel, with which a user can cancel an article sent earlier. This effectively removes the article from the spool directories, if it exists. The cancel message is forwarded to all sites that receive news from the groups affected, regardless of whether the article has been seen already. This takes into account the possibility that the original article has been delayed over the cancellation message. Some news systems allow users to cancel other people's messages; this is, of course, definitively not allowed.

   The Subject: and Control: fields should appear as follows:

   ```
   Subject: cmsg cancel <message ID>
   ```

   ```
   Control: cancel <message ID>
   ```

2. newgroup and rmgroup

   Two messages dealing with creation or removal of newsgroups are the newgroup and rmgroup messages. Newsgroups below the "usual" hierarchies may be created only after a discussion and voting has been held among Usenet readers. The rules applying to the "alt." hierarchy allow for something close to anarchy. For more information, see the regular postings in news.announce.newusers and news.announce.newgroups. Never send a newgroup or rmgroup message yourself unless you definitely know that you are allowed to.

   Subject: and Control: fields should appear as follows:

   ```
   Subject: cmsg newgroup <newsgroup name>
   ```

   ```
   Control: newgroup <newsgroup name>
   ```

   In the body of the message, a brief description of the group's purpose should be included. For example:

   ```
   For your newsgroups file:
   ```

   ```
   feu.students.brave A group for our brave students.
   ```

3. The checkgroups Message

Checkgroups messages are sent by news administrators to make all sites within a network synchronise their active files with the realities of Usenet. For example, commercial Internet Service Providers might send out such a message to their customers' sites. Once a month, the "official" checkgroups message for the major hierarchies is posted to comp.announce.newgroups by its moderator. However, it is posted as an ordinary article, not as a control message.

4. sendsys, version, and senduuname

   Finally, there are three messages that can be used to find out about the network's topology. These are "sendsys", "version", and "senduuname". They cause the news server to return the system description to the sender, as well as a software version string. Again, one should never issue such a message unless it can made sure that it cannot leave your (regional) network. Replies to sendsys messages can quickly bring down a UUCP network.

9.6.4 Transport of Usenet Articles

In the beginning of the Usenet, the protocol **UUCP** ("**Unix to Unix CoPy** ") was the base of communication for Usenet articles. As the name says, news articles were copied from one (Unix) system to the next in a batched mode.

Network News Transfer Protocol (**NNTP**) [RFC977] provides a different approach to news exchange rather than rely on a batch, it allows articles to be exchanged via an interactive network connection very similiar to SMTP (s. Section 7.4.2). NNTP is based on a stream-oriented connection, usually over TCP, between a client and a server. The only requirements for the client hosts are that they be able to open up a TCP/IP connection over the network and have client software that understands NNTP. Most common UNIX-based newsreaders and news-posting programs have built-in NNTP support.

An NNTP daemon runs continuously on the news server host listening on a well-known port, just as the Simple Mail Transfer Protocol (SMTP) server listens on a well-known port for incoming email connections. NNTP client programs connect to the NNTP server and issue commands for reading and posting news articles.

Several commands allow clients to retrieve, send, and post articles. NNTP also provides for an active and a passive way to transfer news, colloquially called "pushing" and "pulling." Pushing means that the news client initiates the transfer of new articles by offering them (see "ihave/sendme" commands in the RFC 977). The client offers an article to the server through the IHAVE msgid command, and the server returns a response code that indicates whether it already has the article or if it wants it. If the server wants the article, the client sends the article, terminated by a single dot on a separate line.

The opposite technique is pulling news, in which the client requests a list of all (available) articles from a group that have arrived after a specified date. This query

is performed by the news reader. From the returned list of message IDs, the client selects those articles it does not yet have, using the "article" command for each of them in turn.

9.6.4.1 The NNTP Protocol

Two NNTP commands are the key to how news articles are pushed or pulled between servers.

Connecting to the News Server

Connecting to the news server is as simple as opening a TCP connection to its **NNTP port** (119). One of the commands available in every situation is "help". The response generally depends upon whether the server believes that the connecting computer is a remote NNTP server or a newsreader, as there are different command sets required. The following listing shows a short session contacting the NNTP port of the university's news server and issueing the commands "help" and "list active".

Listing: NTTP Session

```
C: demuth@eibauer:~ > telnet news.fernuni-hagen.de 119
Trying 132.176.114.41...
S: Connected to oak.FernUni-Hagen.de.
S: Escape character is '^]'.S: 200 oak.fernuni-hagen.de InterNetNews NNRP se
INN 2.2 21-Jan-1999 ready (posting ok).
C: help
S: 100 Legal commands
  authinfo user Name|pass Password|generic <prog> <args>
  article [MessageID|Number]
  body [MessageID|Number]
  date
  group newsgroup
  head [MessageID|Number]
  help
  ihave
  last
  list [active|active.times|newsgroups|distributions|
        distrib.pats|overview.fmt|subscriptions|motd]
  listgroup newsgroup
  mode reader
  newgroups yymmdd hhmmss ["GMT"] [<distributions>]
  newnews newsgroups yymmdd hhmmss ["GMT"] [<distributions>]
  next
  post
  slave
  stat [MessageID|Number]
  xgtitle [group_pattern]
  xhdr header [range|MessageID]
  xover [range]
  xpat header range|MessageID pat [morepat...]
  xpath MessageID
Report problems to <usenet@oak.fernuni-hagen.de>
.
```

```
C: list active
S: 215 Newsgroups in form "group high low flags".
alt.best.of.internet 0000018063 0000017994 y
alt.comp.editors.batch 0000004229 0000004199 y
alt.folklore.suburban 0000002727 0000002727 y
alt.games.vga-planets 0000027087 0000026902 y
alt.music.depeche-mode 0000018880 0000018795 y
.
.
.
de.comp.security.firewall 0000002127 0000001533 y
de.comp.security.misc 0000001123 0000000430 y
de.soc.kultur.japan 0000000662 0000000532 y
.
C: quit
205 .
Connection closed by foreign host.
demuth@eibauer:~ >
```

"list active" shows each supported group and provides information about them. The two numbers in each line of the output are the highest numbered article and lowest numbered article in each group. The newsreader is able to form an idea of the number of articles in the group from these. The last field in the output displays flags that control whether posting is allowed to the group, whether the group is moderated, and whether articles posted are actually stored or just passed on.

The responses to NNTP commands always end with a period (.) on a line by itself. The numbers you see in the output listing are response codes and are used by the server to indicate success or failure of a command. The response codes are described in RFC 977.

Posting an Article

There is a difference between **pushing** an article and **posting** an article. Pushing an article means, that there is an implicit assumption that the article already exists, that it has a message identifier that has been uniquely assigned to it by the server to which it was originally posted, and that it has a complete set of headers. When an article is posted, it is creates for the first time and the only headers supplied are those that are basically meaningful such as the "Subject:" and the "Newgroups:" to which you are posting the article. The news server will add all the other headers for you and create a message ID that it will use when pushing the article onto other servers.

This means, that posting an article is very simple; that simple, that it will be part of an assignment.

The NNTP Reader Mode

Newsreaders use their own set of commands when talking to a news server. To activate these commands, the news server has to be operating in the so called "**reader mode**". Most news servers default to the reader mode, unless the IP address of the connecting host is listed as a news-forwarding peer. In any case, NNTP provides a command to explicitly switch into reader mode: mode reader.

NNTP reader mode has a lot of commands. Many of these are designed to simplify the development of a newsreader. Therefore there are commands that instruct the server to send the head and the body of articles separately. There are also commands that list the available groups and articles, and others that allow posting, an alternate means of sending news articles to the server.

Listing New Articles

When a newsreader first connects to a new server and the user chooses a newsgroup to browse, the newsreader will want to retrieve a list of new articles, those posted or received since the last login by the user. The newnews command is used for this purpose. Three mandatory arguments must be supplied:

The name of the group or groups to query, the start date, and the start time from which to list.

The date and time are each specified as six-digit numbers, with the most significant information first; `yymmdd and hhmmss`, respectively:

Example: `newnews feu.urz.beratung 001101 000000` lists all new articles in the newsgroup feu.urz.beratung from November 11th 2000 midnight.

Selecting a Newsgroup

When the user selects a newsgroup to browse, the newsreader may tell the news server that the group was selected. This simplifies the interaction between newsreader and news server; it removes the need to constantly send the name of the newsgroup with each command. The group command simply takes the name of the selected group as an argument. Many following commands use the group selected as the default, unless another newsgroup is specified explicitly:

Example 9.6-1:
`group feu.urz.beratung`

Response from the news server: `211 50 2754 2803 feu.urz.beratung`

The group command returns a message indicating the number of active messages, the low-water mark, the high-water mark, and the name of the group, respectively. Note that while the number of active messages and the high-water mark are the same in our example, this is not often the case; in an active news server, some articles may

have expired or been deleted, lowering the number of active messages but leaving the high-water mark untouched.

Listing Articles in a Group

To address newsgroup articles, the newsreader must know which article numbers represent active articles. The listgroup command offers a list of the active article numbers in the current group, or an explicit group if the group name is supplied:

Example 9.6-2:
`listgroup feu.urz.beratung`

Response from the news server: A number list of active articles.

Retrieving an Article Header

The user must have some information about an article before he can know whether he wishes to read it. We mentioned earlier that some commands allow the article header and body to be transferred separately. The head command is used to request that the server transmit just the header of the specified article to the newsreader. If the user doesn't want to read this article, time and network bandwidth is not wasted transferring a potentially large article body unnecessarily.

Articles may be referenced using either their number (from the listgroup command) or their message identifier:

```
head 2802
221 2802 <8uui4r$ins$1@oak.fernuni-hagen.de> head
Path: oak.fernuni-hagen.de!not-for-mail
From: "Martina Preuss" <Martina.Preuss@FernUni-Hagen.de>
Newsgroups: feu.urz.beratung
Subject: Re: Logfiles
Date: Wed, 15 Nov 2000 18:37:50 +0100
Organization: FernUni Hagen
Lines: 68
Message-ID: <8uui4r$ins$1@oak.fernuni-hagen.de>
References: <8um8ik$bl5$1@oak.fernuni-hagen.de>
<3A0EEAF9.E5D86D4D@gmx.de> <8uogkb$8hp$1@oak.fernuni-hagen.de>
<9u4qu8.3h4.ln@slarty.dhaller.de>
NNTP-Posting-Host: shiva-hgw-197.fernuni-hagen.de
Mime-Version: 1.0
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: 8bit
X-Trace: oak.fernuni-hagen.de 974310363 19196 132.176.120.197
(15 Nov 2000 17:46:03 GMT)
X-Complaints-To: usenet@oak.fernuni-hagen.de
NNTP-Posting-Date: 15 Nov 2000 17:46:03 GMT
X-Priority: 3
X-MSMail-Priority: Normal
X-Newsreader: Mickeymouse Outlook Express 5.00.2314.1300
```

```
Xref: oak.fernuni-hagen.de feu.urz.beratung:2802
.
```

Retrieving an Article Body

If, on the other hand, the user decides, he does want to read the article, her newsreader needs a way of requesting that the message body be transmitted. The body command is used for this purpose. It operates in much the same way as the head command, except that only the message body is returned:

```
body 2802
222 2802 <8uui4r$ins$1@oak.fernuni-hagen.de> body
Hello,

> David Haller <David@dhaller.de> wrote
>
.
.
```

Reading an Article from a Group

The command "*article*" also accepts an article number or message ID as an argument, but returns the whole article including its header.

Example 9.6-3:
`article 2802`

Attempting to retrieve an unknown article, the server will return a message with an appropriately coded response code and perhaps a readable text message:

Example 9.6-4:
`article 4711`

Response: `423 Bad article number`

10 FTP - File Transfer Protocol

10.1 Goal of the Chapter

The goal of this chapter is to understand the mechanism of the **File Transfer Protocol**. Special emphasis is put on the client-server-interaction while user commands will be considered as well.

10.2 The File Transfer Protocol

The **FTP** is a simple client server protocol for exchanging text and binary files [RFC959]. It has been defined in 1985 and is still used as standard protocol in the Internet.

Up until 1994, when the World Wide Web took over the Internet, FTP was the most widely used Internet client application besides email. It is used as a remote shell for file access on an Internet host. Using an FTP application, one can connect to an FTP server, navigate through the available directories, and transfer files.

An FTP site can be public, private, or both. With a private account, a user can be given access to the entire network's directory structure, or just specific areas.

The Internet is also home to thousands of **public access FTP** servers that allow anyone to connect and transfer files to and from specific directories regardless of whether they have an account on the host. This is called anonymous FTP. When connecting to an **anonymous FTP** site, a user usually specify "anonymous" as the user name and "guest" or the email address as a password.

Anonymous FTP sites are used, for example, to publish a large listing of public domain and/or shareware files. One public FTP site e. g. is ftp.fernuni-hagen.de. FTP was designed mainly for use by programs, but the FTP application itself has turned out to be a critical part of any TCP/IP. In fact, FTP is built into World Wide Web browsers so a user can browse FTP servers with the same program that he uses to browse the Web without recognising that he ist using FTP.

As stated in RFC 959, there were four objectives in the design of the FTP protocol:

1. To promote sharing of files (computer programs and/or data).
2. To encourage indirect or implicit (via programs) use of remote computers.
3. To shield a user from variations in file storage systems among hosts.
4. To transfer data reliably and efficiently.

10.3 FTP Clients

FTP refers to both the **FTP protocol** and an **FTP application**. The FTP protocol defines a series of commands that the client sends the to server, and how the client and server transfer data. An FTP application is usually a character-based terminal-type application in which the user connects to an FTP server. The purpose for the FTP application is to provide natural language commands and help/error messages, as well as higher-level functionality than just a terminal could provide.

Using Unix the command is simple "ftp" or "ftp <hostname>". This is an FTP client application.

10.3.1 Connecting and Logging In

Running FTP for the first time it presents the following prompt:

```
ftp> _
```

The user can connect to any FTP site with the "OPEN" command and has to type "open" followed by the name of the FTP server. For example, to connect to the FernUniversität's public FTP site the following must be typed:

```
ftp> open ftp.fernuni-hagen.de <enter>
```

At this time the program attempts to connect to the server on port 21. Once connected, the client program receives a 220 reply (for explanation see Section 10.4.1) followed by a welcome message. Here is the welcome message at ftp.fernuni-hagen.de:

```
Connected to ftp.fernuni-hagen.de.
220-********** ********** ********** ********** ********** **********
220-*
220-*     Kurze Anleitung fuer nicht geuebte ftp-Benutzer/innen:
220-*
220-*   1. als login-Namen   ftp   oder   anonymous   und nicht etwa
220-*          den irgendwo vorhandenen login-Namen angeben
220-*
220-*   2. als Password bitte die eigene Mail-Adresse
220-*
220-*******


220-******* Achtung: falls der ls-Befehl nicht funktioniert:
220-*******          ls -l                     oder
220-*******          dir                       eingeben
220-*******
220-********** ********** ********** ********** ********** **********
220-
220 ftp FTP server (Version wu-2.6.0(2) Sat Jul 15 12:06:41 MET DST 2000) re
Name (ftp.fernuni-hagen.de:demuth):
```

Next, the user gets a prompt to enter his username. Having an account on the server he can enter his username, but for public access (anonymous FTP) "anonymous" is typed:

```
Name (ftp.fernuni-hagen.de:demuth): anonymous
331 Guest login ok, send your complete e-mail address as password.
```

Next the servers asks for a password. Again, having an account, the password is entered here, but connecting for public access, the email address is entered. The password is not echoed to the screen.

```
Password:
```

The server then grants access with another welcome message, and finally the FTP prompt is waiting again. There are a fixed set of commands that can be used to navigate through the directories on the server and download files.

10.3.2 Listing Directories

One of the commands to navigate through directories is "DIR". In reality there is no DIR command in the FTP protocol specification. However, the standard user interface for accessing FTP servers has brought this command forward from the operating system because it is more user friendly.

The following listing shows the response to a "dir" command at an FTP prompt:

```
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 27
-r--------   1 0        10              0 Jul 13 07:01 .forward
-rwxr-xr-x   1 900      100             0 Apr 26  1996 .notar
-r--------   1 0        10              0 Jul 13 07:01 .rhosts
drwxr-xr-x   2 900      100          1024 Nov 27 23:00 Info
-r--r--r--   1 900      30403        1419 Nov 25  1999 README
-rwxr--r--   1 900      100           200 Nov  8  1996 REFERENZ
drwx------   2 900      100           512 Jul 22  1997 Tmp
d--x--x--x   2 0        30403         512 Jul 19 06:42 bin
d--x--x--x   2 0        30000         512 Jul 14 10:16 dev
-rwxr-xr-x   1 900      100          3247 Dec 11  1996 e_welcome.html
d--x--x--x   2 0        30000         512 Jul 17 10:00 etc
-rw-rw-r--   1 900      100          1946 Apr 29  1998 ftpwelcome.html
d--x--x--x   2 0        30000        8192 Aug  4  1995 lost+found
drwxrwxr-x   2 0        30403         512 Apr 26  1996 msgs
dr-xr-xr-x  24 900      60001         512 Feb 22  1999 pub
d--x--x--x   4 0        30000         512 Jul 12 06:14 usr
-rwxr-xr-x   1 900      100          1956 Apr 29  1998 welcome.html
226 Transfer complete.
ftp>
```

Note that there is a lot more information here than probably expected. In the right-most column is the file or directory name. To the right of that is the file date and size. All the way to the left is a field of 10 bits. These are attributes. One can tell which is a file and which is a directory by the "d" attribute, which is displayed in the far-left attribute field. If there is a "d" then it is a directory.

10.3.3 Changing Directories

Directories are changed with the CD command. Here is the command to change to the /pub directory:

```
ftp> cd pub
250 CWD command successful.
```

10.3.4 Downloading

Downloading a file is simple and straight ahead. Before downloading, it must be made sure, that binary mode is active. There are two modes, ASCII and binary. To change to binary mode, the command "BIN" is used.

```
ftp> bin
200 Type set to I.
```

To change back to ASCII mode, the command "ASC" is used.

The "GET" command is used to retrieve a file. To download with its original filename in the default directory, the following has to be enterd:

```
GET <filename> <enter>
```

Example: FTP Download

```
ftp> get readme.txt
200 PORT command successful.
150 Opening BINARY mode data connection for readme.txt
(1571 bytes).
226 Transfer complete.
1571 bytes received in 3.46 seconds (0.45 Kbytes/sec)
```

Only typing "GET" results in a prompt for the file to download, and then again for the name of the file (and path) on the local system.

10.3.5 Uploading

Uploadinga file is done in much the same way with the "PUT" command. The user has to be in a public area that allows uploads, of course.

Example: FTP Upload

```
ftp> send
(local-file) myfile.zip
(remote-file) myfile.zip
200 PORT command successful.
150 Opening BINARY mode data connection for myfile.zip.
226 Transfer complete.
3018 bytes sent in 0.06 seconds (50.30 Kbytes/sec)
```

10.4 FTP Process Model

The commands an FTP application accepts are a bit different from the commands that an FTP application gives to an FTP server. For example, to get a directory listing with an FTP application a user would use the "ls" command. However, the FTP application uses the "LIST" command.

If there is any one big difference between FTP and the protocols that were covered up to this point, it is that FTP uses more than one port. Port 21, the **control connection**, is used for transferring commands, and another port, the **data connection**, is used for transferring data. The default port for the data connection is 20, but any other port can be used also. This makes the explanation of FTP a bit more difficult to code than SMTP or POP3.

A client opens a connection to the FTP control port (**port 21**) of an FTP server. So that the server will be later able to send data back to the client machine, a second (data) connection must be opened between the server and the client. To make this second connection, the client sends a PORT command to the server machine. This command includes parameters that tell the server which IP address to connect to and which port to open at that address - in most cases this is intended to be a high numbered port on the client machine.

The server then opens that connection, with the source of the connection being **port 20** on the server and the destination being the port identified in the PORT command parameters.

The server then sends the data, and closes the connection. The reason this method is used and not the familiar send data ending with a period on a line by itself method, is because FTP sends binary data. There is no practical way to interpret an end-of-line character when every possible character could be interpreted as data.

Another option is for the client to tell the server to listen to a particular port with the **PASV** command (indicating passive mode), and then connect to that port for the data connection. Going to use the PORT method, it is best to open either port 20 or the next available port over 1024 and then send a PORT command to the server. That way, if the user is already transferring a file with a standalone FTP application, there is no chance of interfering with it (ports 1-1024 are reserved for TCP/IP and standard protocols.)

The PORT command is usually used only in the "active mode" of FTP, which is the default. It is not usually used in passive (also known as PASV) mode. Note that FTP servers usually implement both modes, and the client specifies which method to use.

10.4.1 Reply Code Categories

Each FTP command must result at least into one reply. Replies ensure synchronisation between client and server. Typcial replies are:

125 Data connection already open; transfer starting. 150 File status okay; about to open data connection. 200 Command okay. 202 Command not implemented, superfluous at this site. 220 Service ready for new user. 221 Service closing control connection. 225 Data connection open; no transfer in progress. 226 Closing data connection. 230 User logged in, proceed. 250 Requested file action okay, completed. 421 Service not available, closing control connection. 425 Can’t open data connection. 426 Connection closed; transfer aborted.

Each digit of the reply code has a specific meaning. There are five values for the first digit of the reply code: "1" indicates a positive preliminary reply (the command was accepted, and this is the first of more than one positive reply from the server); "2" indicates a permanent positive reply; "3" indicates a positive intermediate reply, in which case the server is waiting for more information; "4" indicates that the command was not accepted and the requested action did not occur, yet the condition may be temporary; "5" indicates absolute failure.

The second digit indicates the category of the reply: 0 indicates a syntax error; 1 indicates informational content; 2 indicates a message concerning the transmission channel; 3 refers to authentication or accounting messages; 4 is not used; and 5 indicates a message regarding the file system status. The third digit merely specifies the level of granularity of messages in a particular category.

The following listing shows a summary of how to interpret FTP reply codes. Consult RFC 959 for a complete discussion.

```
1xx Positive Preliminary Reply
2xx Positive Reply
3xx Positive Intermediate Reply
4xx Transient Negative Completion Reply
5xx Permanent Negative Completion Reply
x0x Syntax error
```

```
x1x Information
x2x Connections
x3x Authentication and accounting
x4x Unspecified as yet
x5x File system
```

11 IPv6

11.1 From IPv4 to IPv6

This chapter provides information about the **Next Generation Internet Protocol (IPng)**. The formal name of the IPng protocol is **IPv6**. The current version of the Internet Protocol is version 4 (referred to as IPv4).

IPng is a new version of IP which is an evolutionary step from IPv4. It can be installed as a normal software upgrade and is interoperable with the current IPv4. IPng is designed to run on high performance networks (e.g. Gigabit Ethernet, OC-12, ATM, etc.) and at the same time still be efficient for low bandwidth networks (e.g. wireless or modem). In addition, many new features will be included that will satisfy the needs of a future Internet.

IPng, of course, includes a transition mechanism to provide direct interoperability between IPv4 and IPng hosts. The IPng transition allows the users to upgrade their hosts to IPng, and the network operators to deploy IPng in routers, with little coordination between the two.

11.2 Reasons for the development of a new Protocol

Network growth is the basic reason for a next generation IP. The experience with IPv4 shows that addressing and routing must be capable of handling reasonable scenarios of future growth. Currently IPv4 serves what could be called the computer market. The computer market has been the driver of the growth of the Internet. It comprises the current Internet and countless other smaller intranets which are not connected to the Internet. Its focus is to connect computers together in the large business, government, and university education markets. This market has been growing at an exponential rate. One measure of this is that the number of networks in current Internet is doubling approximately every 12 months. The computers which are used at the endpoints of internet communications range from PC’s to Supercomputers. Most are attached to Local Area Networks (LANs) and the vast majority are not mobile.

Nomadic personal computing devices seem certain to become ubiquitous as their prices drop and their capabilities increase. A key capability is that they will be networked. Unlike the majority of today’s networked computers they will support a variety of types of network attachments. When disconnected they will use wireless networks, when used in networked facilities they will use infrared attachment, and when docked they will use physical wires. This makes them a candidate for internetworking technology as they will need a protocol which can work over a variety of physical networks. These types of devices will become consumer devices and will replace the current generation of cellular phones, pagers, and personal digital assistants. In addition to these requirements there is a need of supporting large scale routing and addressing. Therefore, the new internet protocol must introducing little

overhead and support auto configuration and mobility as basic elements. The requirement for little overhead is due to the wireless media. Unlike LAN's which are high speed, the wireless media is much slower.

Another issue is networked entertainment. Proposals are being discussed for 500 channels of television, video on demand, etc. Plans are that every television set will become an Internet host. As digital high definition television (**HD-TV**) approaches, the differences between a computer and a television will diminish. These devices require an Internet protocol which supports large scale routing and addressing as well as auto configuration.

Another use for the next generation IP would be device control. This means control of everyday devices such as lighting equipment, heating and cooling equipment, motors, and other types of equipment which are currently controlled via analog switches and in aggregate consume considerable amounts of electrical power.

The challenge in the selection of an IPng is to pick a protocol which meets today's requirements and also matches the requirements of new markets. The alternative is a world of disjoint networks with protocols controlled by individual vendors which is unacceptable.

IPng was recommended by the Internet Engineering Task Force (IETF) on July 25, 1994 in RFC 1752, "The Recommendation for the IP Next Generation Protocol" [RFC1752].

11.3 Features of IPv6

11.3.1 Expanded Addressing Capabilities

IPv6 increases the IP address size from 32 bits to 128 bits, to support more levels of addressing hierarchy, a greater number of addressable nodes and simpler auto-configuration of addresses. The scalability of multicast routing is improved by adding a "scope" field to multicast addresses. A new type of address called an "**anycast address**" is also defined, used to send a packet to any one of a group of nodes.

11.3.2 Header Format Simplification

Some IPv4 header fields have been dropped or made optional, to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header.

11.3.3 Improved Support for Extensions and Options

IP header options encoding is changed witch allows for more efficient forwarding, less limits on the length of options, and flexibility for introducing new options.

11.3.4 Flow Labeling Capability

A new capability is added to enable the labeling of packets belonging to particular traffic "flows" for which the sender requests special handling, such as a non-default quality of service or "real-time" service.

11.3.5 Authentication and Privacy Capabilities

Extensions to support authentication, data integrity, and (optional) data confidentiality are specified for IPv6.

11.4 Transition to IPng

Two factors are driving the transition to IPv6: routing and addressing. Global internet routing based on the 32-bit addresses of IPv4 is becoming increasingly strained. IPv4 addresses do not provide flexibility to construct hierarchies which can be aggregated. The deployment of Classless Inter-Domain Routing (CIDR) [18] is extending the lifetime of IPv4 routing by a number of years, but the effort to manage the routing will continue to increase. Even if the IPv4 routing can be scaled to support a full IPv4 Internet, the Internet will eventually run out of network numbers.

The challenge for an IPng is for its transition to be complete before IPv4 routing and addressing break. The transition will be easier if IPv4 addresses are still globally unique. The two transition requirements which are the most important are flexibility of deployment and the ability for IPv4 hosts to communicate with IPng hosts. The capability must exist for IPng-only hosts to communicate with IPv4-only hosts globally while IPv4 addresses are globally unique. Features need to be designed into an IPng to make the transition as easy as possible.

11.5 The Protocol: IPv6

IPv6 is standardised in the standard track of IETF in RFC 2460 [DH95].

11.5.1 The basic header format

The IPng protocol consists of two parts, the basic IPng header and IPng extension headers. (see Table 11.5-1)

Fig. 11.5-1: Format of the IPv6 Header. The Header comprises fewer fields than the IPv4 Header.

Tab. 11.5-1: *Explanation of the IPv6 Header fields*

Field	Description
VERSION	Internet Protocol version number = 6
PRIORITY	Priority Value
FLOW LABEL	Quality of Service
PAYLOAD LEN	16-bit unsigned integer. Length of payload, i.e., the rest of the packet following the IPng header, in octets.
NEXT HDR	8-bit selector. Identifies the type of header immediately following the IPng header. Uses the same values as the IPv4 Protocol field, explicit frame referencing
HOP LIMIT	8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.
SOURCE ADDRESS	128 bits. The address of the initial sender of the packet.
DESTINATION ADDRESS	128 bits. The address of the intended recipient of the packet (possibly not the ultimate recipient, if an optional Routing Header is present).

11.5.2 Extension Headers

IPng options are placed in separate extension headers that are located between the IPng header and the payload, respective the transport-layer header in a packet. Most extension headers are not examined or processed by any router along a packet's delivery path until it arrives at its final destination. This facilitates an improvement in router performance for packets containing options. The cause is that a router has to retrieve the destination address from its lookup-table. This has to be done in every processed packet. An additional statement in the router software causes extra processed code which decreases the performance of a router. In IPv4, the presence of options requires the router to examine all options.

The other improvement is that IPng extension headers can be of arbitrary length and the total amount of options carried in a packet is not limited to 40 bytes (as in IPv4). An example of these features is the IPng Authentication and Security Encapsulation options. In order to improve the performance when handling subsequent option headers and the transport protocol which follows, IPng options are always an integer multiple of 8 octets long, so called 8-byte-alignment. The IPng extension headers which are currently defined are:

Ext. Header	Explanation
ROUTING	Extended Routing (like IPv4 loose source route).
FRAGMENTATION	Fragmentation and reassembly
AUTHENTICATION	Integrity and authentication. Security
ENCAPSULATION	Confidentiality
HOP-BY-HOP OPTION	Special options which require hop by hop processing
DESTINATION OPTIONS	Optional information to be examined by the destination node.

11.5.3 Adressing

IPng addresses are 128-bits long and are identifiers for individual interfaces and sets of interfaces. IPng Addresses of all types are assigned to interfaces, not nodes. Since each interface belongs to a single node, any of that node's interfaces' unicast addresses may be used as an identifier for the node. A single interface may be assigned multiple IPv6 addresses of any type.

There are three types of IPng addresses:

1. Unicast addresses identify a single interface.
2. Anycast addresses identify a set of interfaces such that a packet sent to an anycast address will be delivered to one member of the set.
3. Multicast addresses identify a group of interfaces, such that a packet sent to a multicast address is delivered to all of the interfaces in the group. There are

no broadcast addresses in IPv6, their function being superseded by multicast addresses.

The assignment and routing of addresses requires the creation of hierarchies which reduces the efficiency of the usage of the address space. Christian Huitema performed an analysis in [HUI94] which evaluated the efficiency of other addressing architecture's (including the French telephone system, USA telephone systems, current internet using IPv4, and IEEE 802 nodes). He concluded that 128bit IPng addresses could accommodate between $8x10^{17}$ to $2x10^{33}$ nodes assuming efficiency in the same ranges as the other addressing architecture's.

The specific type of IPng address is indicated by the leading bits in the address. The variable-length field comprising these leading bits is called the **Format Prefix (FP)**. The initial allocation of these prefixes is shown in Table 11.5-2.

Tab. 11.5-2: *Initial allocation of prefixes*

Allocation	Prefix(binary)	Fraction of Address Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP Allocation	0000 001	1/128
Reserved for IPX Allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Unassigned	001	1/8
Provider-Based Unicast Address	010	1/8
Unassigned	011	1/8
Reserved for Neutral-Interconnect-Based Unicast Addresses	100	1/8
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link Local Use Addresses	1111 1110 10	1/1024
Site Local Use Addresses	1111 1110 11	1/1024
Multicast Addresses	1111 1111	1/256

There are several forms of unicast address assignments in IPv6. These are the global provider based unicast address, the neutral-interconnect unicast address, the NSAP

address, the IPX hierarchical address, the site-local-use address, the link-local-use address, and the IPv4-capable host address. Additional address types can be defined in the future.

The remainder of the address space is unassigned for future use. Note that Anycast addresses are not shown here because they are allocated out of the unicast address space. Approximately fifteen percent of the address space is initially allocated. The remaining 85% is reserved for future use.

11.5.4 Provider-based Unicast Addresses

Provider-based unicast addresses are used for global communication. They are similar in function to IPv4 addresses under CIDR. The assignment plan for unicast addresses is described in [RLi95] and [RLo95]. Their format is:

The first 3 bits identify the address as a provider-oriented unicast address. The next field (REGISTRY ID) identifies the internet address registry which assigns provider identifiers (PROVIDER ID) to internet service providers, which then assign portions of the address space to subscribers. This usage is similar to assignment of IP addresses under CIDR [FLYV93].

The SUBSCRIBER ID distinguishes between multiple subscribers attached to the internet service provider identified by the PROVIDER ID. The SUBNET ID identifies a specific physical link. There can be multiple subnets on the same physical link. A specific subnet can not span multiple physical links.

The INTERFACE ID identifies a single interface among the group of interfaces identified by the subnet prefix.

11.5.5 Local-Use Addresses

A local-use address is a unicast address that has only local routability scope (within the subnet or within a subscriber network), and may have local or global uniqueness scope. They are intended for use inside of a site for local communication and for bootstrapping up to the use of global addresses [Tho95].

There are two types of local-use unicast addresses defined. These are "Link-Local" and "Site-Local". The Link-Local-Use is for use on a single link and the Site-Local-Use is for use in a single site.

Link-Local-Use addresses is displayed in Fig. 11.5-2.

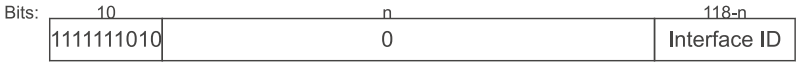


Fig. 11.5-2: Link-local-Use address format

Link-Local-Use addresses are designed to be used for addressing on a single link for purposes such as auto-address configuration.

The format of Site-Local-Use addresses is displayed in Fig. 11.5-3.

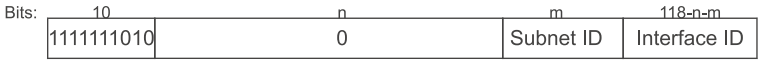


Fig. 11.5-3: Site-local-Use address format

For both types of local use addresses the INTERFACE ID is an identifier which must be unique in the domain in which it is being used. In most cases these will use a node's IEEE-802 48bit address. The SUBNET ID identifies a specific subnet in a site. The combination of the SUBNET ID and the INTERFACE ID to form a local use address allows a large private internet to be constructed without any other address allocation.

Local-use addresses allow organisations that are not connected to the global Internet to operate without the need to request an address prefix from the global Internet address space. Local-use addresses can be used instead. If the organisation later connects to the global Internet, it can use its SUBNET ID and INTERFACE ID in combination with a global prefix (e.g., REGISTRY ID + PROVIDER ID + SUBSCRIBER ID) to create a global address. This is a significant improvement over IPv4 which requires sites which use private (non-global) IPv4 address to manually renumber when they connect to the Internet. IPng does the renumbering automatically.

11.5.6 IPv6 Addresses with Embedded IPV4 Addresses

The IPv6 transition mechanisms include a technique for hosts and routers to dynamically tunnel IPv6 packets over IPv4 routing infrastructure. IPv6 nodes that utilize this technique are assigned special IPv6 unicast addresses that carry an IPv4 address in the low-order 32-bits. This type of address is termed an "IPv4-compatible IPv6 address" and has the format:

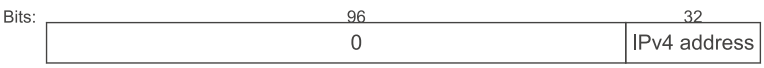


Fig. 11.5-4: IPv6 Address compatible with IPv4

A second type of IPv6 address which holds an embedded IPv4 address is also defined. This address is used to represent the addresses of IPv4-only nodes (those that do not support IPv6) as IPv6 addresses. This type of address is termed an "IPv4-mapped IPv6 address" and has the format:

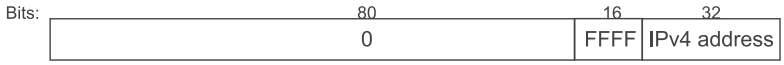


Fig. 11.5-5: IPv6 Address compatible with IPv4 (II)

11.5.7 Anycast Addresses

An IPv6 anycast address is an address that is assigned to more than one interfaces (typically belonging to different nodes), with the property that a packet sent to an anycast address is routed to the "nearest" interface having that address, according to the routing protocols' measure of distance.

Anycast addresses, when used as part of an route sequence, permits a node to select which of several internet service providers it wants to carry its traffic. This capability is sometimes called "source selected policies". This would be implemented by configuring anycast addresses to identify the set of routers belonging to internet service providers (e.g., one anycast address per internet service provider). These anycast addresses can be used as intermediate addresses in an IPv6 routing header, to cause a packet to be delivered via a particular provider or sequence of providers. Other possible uses of anycast addresses are to identify the set of routers attached to a particular subnet, or the set of routers providing entry into a particular routing domain.

Anycast addresses are allocated from the unicast address space, using any of the defined unicast address formats. Thus, anycast addresses are syntactically indistinguishable from unicast addresses. When a unicast address is assigned to more than one interface, thus turning it into an anycast address, the nodes to which the address is assigned must be explicitly configured to know that it is an anycast address.

11.5.8 Multicast Addresses

A multicast address is an identifier for a group of interfaces. An interface may belong to any number of multicast groups. IPng Multicast addresses have the following format:

Fig. 11.5-6: IPv6 Multicast Address

(see Table 11.5-3)

Tab. 11.5-3:

11111111	at the start of the address identifies the address as being a multicast address			
FLGS	is a set of 4 flags |0|0|0|T|. The high-order 3 flags are reserved, and must be initialized to 0.			
	T=0	indicates a permanently assigned ("well-known") multicast address, assigned by the global internet numbering authority		
	T=1	indicates a non-permanently assigned ("transient") multicast address.		
SCOP	is a 4-Bit multicast scope value used to limit the scope of the multicast group. The values are:			
	1	reserved	8	organization-local scope
	2	node-local scope	9	(unassigned)
	3	link-local scope	A	(unassigned)
	4	(unassigned)	B	(unassigned)
	5	(unassigned)	C	(unassigned)
	6	site-local scope	D	(unassigned)
	7	(unassigned)	E	global scope
	8	(unassigned)	F	reserved
GROUP ID	identifies the multicast group, either permanent or transient, within the given scope.			

11.5.9 Routing

Routing is almost identical to IPv4 routing under CIDR except that the addresses are 128-bit addresses instead of 32-bit IPv4 addresses. With extensions, all of IPv4's routing algorithms (OSPF, DVRP, RIP, IDRP, ISIS, etc.) can be used to route IPng.

IPng also includes simple routing extensions. These capabilities include:

1. Provider Selection (based on policy, performance, cost, etc.)
2. Host Mobility (route to current location)
3. Auto-Readdressing (route to new address)

The new routing functionality is provided by creating sequences of IPng addresses using the IPng Routing option. The routing option is used by an IPng source to list one or more intermediate nodes (or topological group) to be "visited" on the way to a packet's destination. This function is similar to IPv4's Loose Source and Record Route option.

In order to make address sequences a general function, IPng hosts are required in most cases to reverse routes in a packet it receives (if the packet was successfully authenticated using the IPng Authentication Header) containing address sequences

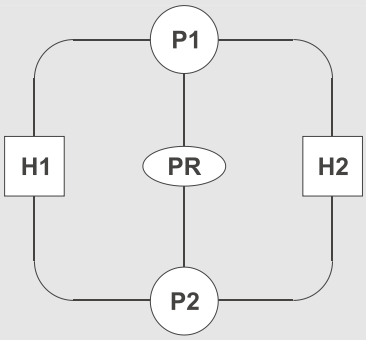
in order to return the packet to its originator. This approach is taken to make IPng host implementations from the start support the handling and reversal of source routes. This is for allowing them to work with hosts which implement the features such as provider selection or extended addresses.

Example 11.5-1:
Address sequences are shown by a list of individual addresses separated by commas. For example:

```
SRC, I1, I2, I3, DST
```

Where SRC is the source address, DST is the destination address, and the middle addresses are intermediate addresses.

For examples assume that two hosts, H1 and H2 wish to communicate. Assume that H1 and H2's sites are both connected to providers P1 and P2. A third wireless provider, PR, is connected to both providers P1 and P2.



The simplest case (no use of address sequences) is when H1 wants to send a packet to H2 containing the addresses:

H1, H2

When H2 replies, it reverses the addresses and constructs a packet containing the addresses:

H2, H1

In this example either provider could be used, and H1 and H2 would not select which provider traffic would be sent to and received from.

If H1 wants that all communication to H2 should only use provider P1, it constructs a packet containing the address sequence:

H1, P1, H2

This ensures that when H2 replies to H1, it will reverse the route and the reply would also travel over P1. The addresses in H2's reply look like:

H2, P1, H1

11.6 Stream Transport

11.6.1 IPng Quality-of-Service Capabilities

The Flow Label and the Priority fields in the IPng header may be used by a host to identify those packets for which it requests special handling by IPng routers, such as non-default quality of service or "real-time" service. This capability is important in order to support applications which require some degree of consistent throughput, delay, and/or jitter.

11.6.2 Flow Labels

A flow is a sequence of packets sent from a source to a (unicast or multicast) destination for which the source or destination desires special handling by the routers passed. The special handling might be conveyed to the routers by a control protocol, such as a resource reservation protocol, or by information within the flow's packets themselves, e.g., in a hop-by-hop option.

There may be multiple active flows from a source to a destination, as well as traffic that is not associated with any flow. A flow is uniquely identified by the combination of a source address and a non-zero flow label. Packets that do not belong to a flow carry a flow label of zero.

The 24-bit Flow Label field in the IPv6 header may be used by a source to label those packets for which it requests special handling by the IPv6 routers. Hosts or routers that do not support the functions of the Flow Label field are required to set the field to zero when originating a packet, pass the field on unchanged when forwarding a packet, and ignore the field when receiving a packet.

A flow label is assigned to a flow by the flow's source node. New flow labels must be chosen (pseudo-)randomly and uniformly. The random allocation is to make any set of bits within the Flow Label field suitable for use as a hash key by routers, for looking up the state associated with the flow. All packets belonging to the same flow must be sent with the same source address, same destination address, and same non-zero flow label. If any of those packets includes a Hop-by-Hop Options header, they must all be originated with the same Hop-by-Hop Options header contents. The routers or destinations are permitted to verify that these conditions are satisfied. If a violation is detected, it should be reported to the source (by an ICMP Parameter Problem message [CD95]).

Routers can set up flow-handling state for any flow, even when no explicit flow establishment information has been provided to them. As mentioned before, this can happen via a control protocol or a hop-by-hop header option. For example, receiving a packet from a source with an unknown, and therefore non-zero flow label, a router may process its IPv6 header and any necessary extension headers as if the flow label were zero.

That processing would include determining the next-hop interface, and updating a hop-by-hop option, or deciding on how to queue the packet based on its Priority

field. The router stores the results of those processing steps and caches that information (source address plus flow label). Subsequent packets with the same tupel (source address, flow label) are handled by referring to the cached information rather than examining all fields once again. According to the requirements of flow-labelling, it can be assumed that this tupel remains unchanged from the first packet in the flow.

11.6.3 Priority

The 4-bit Priority field in the IPv6 header enables a delivery priority of packets, relative to other packets from the same source. The Priority values are divided into two ranges:

Value	Traffic
0 - 7	source is providing congestion control (TCP-Traffic)
8 - 15	no congestion control, like constant rate streams (video) in real-time traffic

For congestion-controlled traffic, the following Priority values are recommended for particular application categories:

0	Uncharacterized traffic
1	"Filler" traffic (netnews)
2	Unattended data transfer (email)
3	(Reserved)
4	Attended bulk transfer (FTP, HTTP, NFS)
5	(Reserved)
6	Interactive traffic (telnet, X-Windows)
7	Internet control traffic (routing protocols, SNMP)
8	non-congestion-controlled traffic (high-fidelity video traffic)
...	
15	discarded (low-fidelity audio traffic)

For non-congestion-controlled traffic, the lowest Priority value (8) is used for packets the sender should discard under conditions of congestion, and the highest value (15) is used for packets that the sender is least willing to have discarded (e.g., low-fidelity audio traffic). There is no relative ordering implied between the congestion-controlled priorities and the non-congestion-controlled priorities.

11.6.4 IPng Security

IPv4 has security problems and lacks privacy and authentication mechanisms. IPng tries to avoid these lacks by introducing two integrated options that provide security services [Atk95a].

11.6.4.1 IPng Authentication Header

This is an extension header which provides authentication and integrity (without confidentiality) to IPng datagrams [Atk95b]. While the extension is algorithm-independent and will support many different authentication techniques, the use of keyed MD5 is proposed to help ensure interoperability within IPng. This should eliminate network attacks, including host masquerading attacks. IPv4 source routing does not support manipulated packet headers. IPng Authentication header for source routing placed at the internet layer can help provide host origin authentication to upper layer protocols and services that currently lack meaningful protections.

11.6.4.2 IPng Encapsulating Security Header

This extension header mechanism provides integrity and confidentiality to IPng datagrams. It is simpler than the enumerated security protocols (SP3D, ISO NLSP, etc.) but remains flexible and algorithm-independent. For interoperability, the use of DES CBC is being used as the standard algorithm for use with the IPng Encapsulating Security Header.

11.6.5 IPng Transition Mechanisms

The key transition objective is to allow IPv6 and IPv4 hosts to interoperate. A second objective is to allow IPv6 hosts and routers to be deployed in the Internet in a diffuse and incremental way, with few interdependencies. A third objective is that the transition should be as easy as possible for end-users, system administrators, and network operators to understand and carry out.

The IPng transition mechanisms provides the following features:

- Individual IPv4 hosts and routers may be upgraded to IPv6 one at a time without requiring any other hosts or routers to be upgraded at the same time. New IPv6 hosts and routers can be installed one by one. This can be called incremental upgrade and deployment.
- One prerequisite to upgrading hosts to IPv6 is that the DNS server must first be upgraded to handle IPv6 address records. There are no pre-requisites to upgrading routers. So there are minimal upgrade dependencies.
- When existing installed IPv4 hosts or routers are upgraded to IPv6, they may continue to use their existing address. They do not need to be assigned new addresses. This is a way of easy addressing. Administrators do not need to draft new addressing plans.
- An IPv6 addressing structure that embeds IPv4 addresses within IPv6 addresses, and encodes other information used by the transition mechanisms.
- All hosts and routers are upgraded to IPv6 in the early transition phase, and are capable of IPv4 and IPv6 protocol stacks.

- The technique of encapsulating IPv6 packets within IPv4 headers to carry them over segments of the end-to-end path where the routers have not yet been upgraded to IPv6.
- The header translation technique to allow the introduction of routing topologies exclusively for IPv6 traffic routing, and the deployment of hosts that support only IPv6. Use of this technique comes up in the later phase of transition.

The IPng transition mechanisms should ensure that IPv6 hosts can interoperate with IPv4 hosts. If IPv4 addresses run out, the transisiton mechanism allows IPv6 and IPv4 hosts within a limited scope to interoperate indefinitely after that. This feature protects the investments done in IPv4 and ensures that IPv6 does not render IPv4 obsolete. Hosts that need only a limited connectivity range (e.g., printers) need never be upgraded to IPv6.

11.7 Summary

This chapter briefly introduced the need of an evolution of IPv4 done with a state analysis, discussing where new internet devices will be introduced in the future. The new hierachical addressing scheme and the new IPv6 header format is explained as well as the new quality-of-service features provided by IPv6. Finally, transition strategies for moving from IPv4 to IPv6 are described.

Solutions for Exercises

Solution for Exercise 2.2-1:

a. The general method to represent an integer a in the binary system is to express a in the form

$$a = a_k * 2^k + a_{k-1} * 2^{k-1} + \cdots + a_2 * 2^2 + a_1 * 2 + a_0,$$

and to represent it by the symbol

$$a_k a_{k-1} a_{k-2} \cdots a_1 a_0.$$

The digits a_i are the remainders left after successive division of a by 2.

The ASCII code of the character F is 70 in decimal notation. The binary representation of 70 can be calculated as follows:

$$\begin{aligned} 70/2 &= 35 + 0 \\ 35/2 &= 17 + 1 \\ 17/2 &= 8 + 1 \\ 8/2 &= 4 + 0 \\ 4/2 &= 2 + 0 \\ 2/2 &= 1 + 0 \\ 1/2 &= 0 + 1 \end{aligned}$$

$$F : 70_{10} = 1000110_2 = 1 * 2^6 + 1 * 2^2 + 1 * 2^1$$

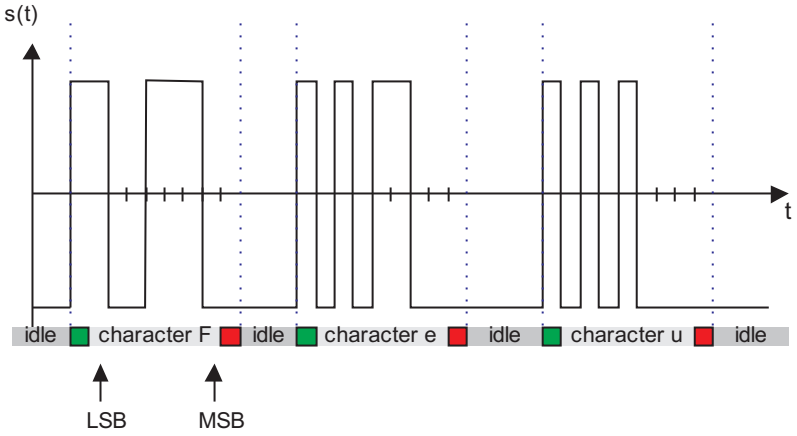
The ASCII code of the character e is 101 in decimal notation

$$\begin{aligned} 101/2 &= 50 + 1 \\ 50/2 &= 25 + 0 \\ 25/2 &= 12 + 1 \\ 12/2 &= 6 + 0 \\ 6/2 &= 3 + 0 \\ 3/2 &= 1 + 1 \\ 1/2 &= 0 + 1 \end{aligned}$$

e: $101_{10} = 1100101_2 = 1 * 2^6 + 1 * 2^5 + 1 * 2^2 + 1 * 2^0$

u : $117_{10} = 1110101_2$

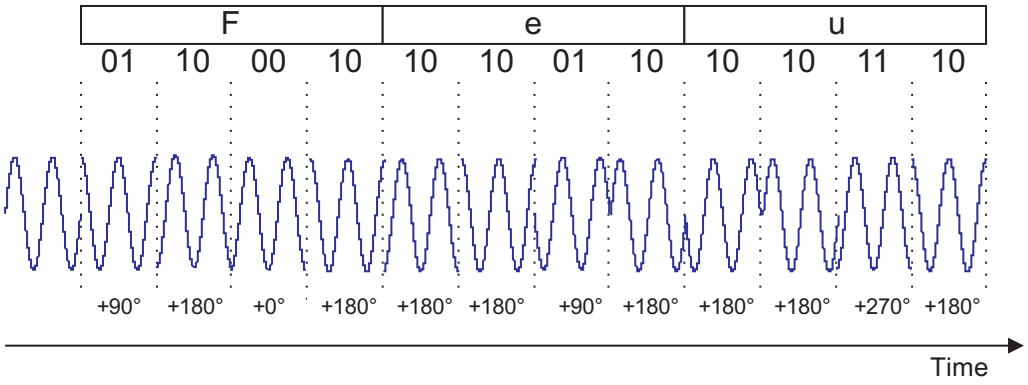
b) RS-232 signal diagramm



Solution for Exercise 2.2-2:

a. F: 01000110, e: 01100101, u: 01110101

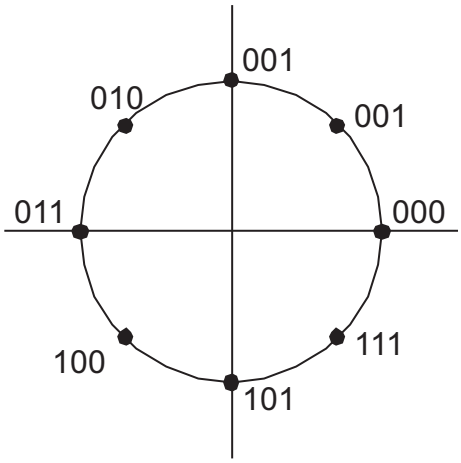
b. QPSK modulation:



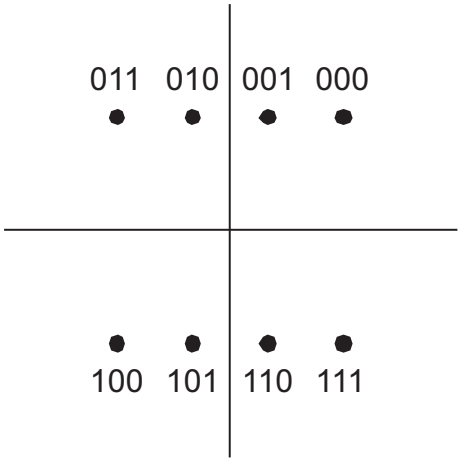
c. The baud rate is 2400/2 = 1200 baud.

Solution for Exercise 2.2-3:

a. 8-PSK (one of many possible solutions):



b. 8-QAM (one of many possible solutions):



Solution for Exercise 2.2-4:

$$30dB = 10 \log_{10} \frac{S}{N}$$

$$\frac{S}{N} = 10^3 = 1000$$

$$C = 3100 \times \log_2(1 + 1000)$$
$$= 30898 \text{ bit/s}$$

Solution for Exercise 2.3-1:

no solution

Solution for Exercise 2.4-1:

ISDN: efficiency = 192 kbit/s / 80 kHz = 2.4 bit/s/Hz

V.34: efficiency = 33.6 kbit/s / 3.1 kHz = 10.8 bit/s/Hz

The modulation scheme of V.34 is more efficient than the ISDN baseband transmission scheme. However, the high efficiency also leads to increased complexity of the modem circuitry.

Solution for Exercise 8.4-1:

```
<!ELEMENT glossary    (entry*) >
<!ELEMENT entry       (term, definition) >
<!ELEMENT term        (#PCDATA) >
<!ELEMENT def         (#PCDATA) >

<!ATTLIST entry
   id CDATA #REQUIRED >
```

Solution for Exercise 8.4-2:

No solution

Solution for Exercise 8.4-3:

Solution for Exercise 8.4-4:

Listing: HTML Glossary

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
                "http://www.w3.org/TR/REC-html40/strict.dtd">
<html>
<head>
<title>Glossary</title>
</head>
<body>
<h3>Glossary</h3>
<p><a href="#modem">Modem</a></p>
<p><a href="#phasemod">Phase modulation</a></p>
<p><a href="#internet">Internet</a></p>
<dl>
<div id="modem">
<dt>Modem</dt>
<dd>
A device that converts the digital signals produced by
terminals and computers into the analog signals that
telephone circuits are designed to carry.
</dd>
</div>
<div id="phasemod">
<dt>Phase modulation</dt>
<dd>
Phase is the position of a waveform of a signal with
respect to the origination of the carrier cycle.  Thus,
phase modulation is the process of varying the carrier
signal with respect to the origination of its cycle.
Several forms of phase modulation are used in modems,
including single- and multiple-bit phase-shift keying
(PSK) and the combination of amplitude and multiple-bit
phase-shift keying.
</dd>
</div>
<div id="internet">
<dt>Internet</dt>
<dd>
The Internet is a large data network of networks. It grew
out of the ARPAnet, which was original operated by the
U.S. Defense Advanced Research Projects Agency, and was
based on TCP/IP. The Internet still supports TCP/IP but
encompasses additional networking protocols as well.
</dd>
</div>
</dl>
</body>
</html>
```

Solution for Exercise 8.5-1:

Solution: see Fig. 8.5-1 for a screen shot. The result will depend on the browser. If possible use a current version of your favourite browser.

Step 1:
The body element is assigned the following style sheet:

```
body { left-margin: 4em;
       background-color: white;
}
```

All elements in the body section will inherit these settings.

Step 2:
The link behaviour is set with the `a:link` and `a:visited` pseudo classes.

```
a:link { text-decoration: none; color: blue;}
a:visited { text-decoration: none; color: blue;}
```

Step 3:
The box is created with the following style sheet assigned to the `div` element:

```
   div { background-color: silver;
         margin: 1em 0em;
         border: black solid 2px;
         width: 20em;
         padding: 1em;
       }
```

The vertical space is created by employing the margin property. In this case top and bottom margin are assigned a value of 1em, right and left margin are set to 0em. The width of the box is created by applying the `width` property. The padding is used to implement the space of 1em between border and content (the padding is simultaneously set for all four sides).

Step 4:
The term is rendered in a bold font:

```
   dt { font-weight: bold; }
```

Step 5:
No style sheet is applied to the `dd` element.

Step 6:
The margin of the `dt` and `dd` element is set to the same value to implement the alignment. The space between content and border is already controlled by the padding property of the `div` element. Hence, the margin of the dt and dd element can be set to zero:

```
   dt { font-weight: bold;
        margin: 0em;
      }
   dd { margin: 0em; }
```

The following listing shows the resulting style sheet code. It can be included in the head section of the HTML document created in Section 8.4.4.

```
...
<head>
...
<style type="text/css">
   a:link { text-decoration: none; color: blue;}
   a:visited { text-decoration: none; color: blue;}
   body { margin-left: 4em;
          background-color: white;
        }
   div { background-color: silver;
         margin: 1em 0em;
         border: black solid 2px;
         width: 20em;
         padding: 1em;
       }
   dt { font-weight: bold;
        margin: 0em;
      }
   dd { margin: 0em; }
</style>
...
</head>
...
```

Please make further experiments with the code.

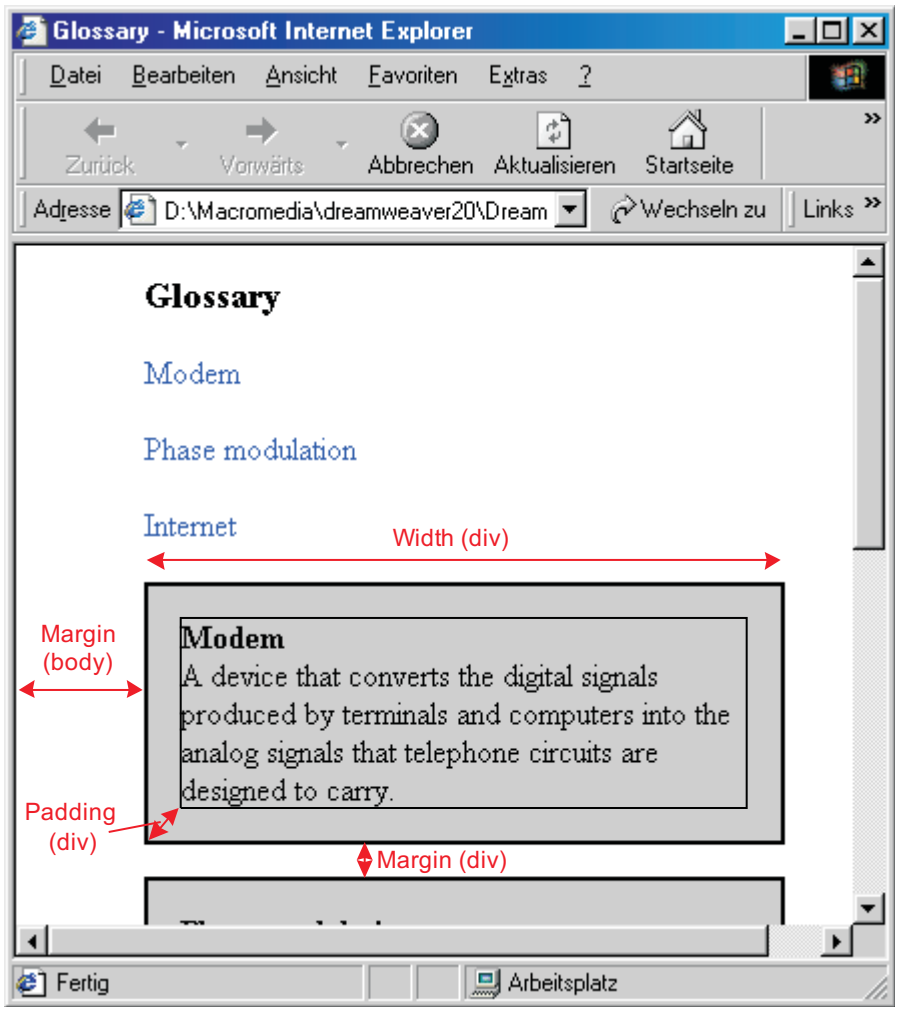


Fig. 8.5-1:

Solution for Exercise 8.6-1:

XML order:

```
<order>
  <order-no>B00002</order-no>
  <date>19940202</date>
  <buyer>
    <name>Stadt- und Universitaetsbibliothek</name>
    <address>
      <street>Bockenheimer Landstr. 134-138</street>
      <city>Frankfurt</city>
      <postcode>60325</postcode>
    </address>
    <Buyer-ID>DE1141110388</Buyer-ID>
  </buyer>
  <supplier>
    <name>DREIER</name>
  </supplier>
  <currency>DEM</currency>
</order>
```

It can be seen that the XML document is self-descriptive. It can be easily read humans and can still be processed by a computer. The original EDI message can only be read by a human who is an expert in the underlying EDI format (in this case EDIFACT).

Solution for Exercise 8.7-1:

no solution

Assignments

Assignments for Chapter "Basics of Communication Networks"

Assignment 1: *Send an e-mail to the tutor of this course!* **4 P.**

The e-mail address of the tutor is:

ea.k20018@ks.fernuni-hagen.de
Indicate in the message text of the e-mail your full name and your registration number. You will get an automatic receipt.

If you have technical problems sending this e-mail contact the computer centre of the university of Hagen:

Tel. No. +49 (0) 2331/987-2847

Fax No. +49 (0) 2331/987-19-2847

Assignment 2: The address of the newsgroup is: **4 P.**

`feu.ice-bachelor.kurs.20018.diskussion`

If you have technical problems writing a message in the course newsgroup contact the computer centre of the university of Hagen:

Tel. No. +49 (0) 2331/987-2847

Fax No. +49 (0) 2331/987-19-2847

Assignment 3: PCM **8 P.**

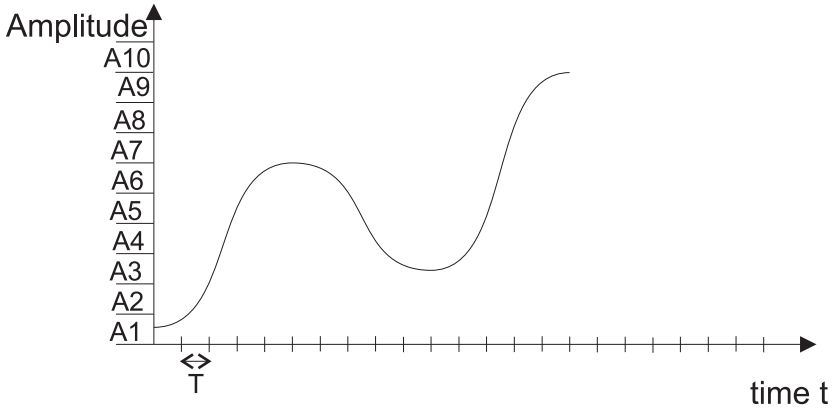


Fig. 1: Sound wave

In Fig. 1 we see an analogue sound wave which shall be digitized.

Solution hints: Copy the image above (HTML version: `http://www.ice-bachelor.fernuni-hagen.de/lehre/k20018.q1/ea/ea01/bild`) and edit it in an image editor of your choice. For each of the four tasks above create a separate image.

2 P. a) Sample the sound wave in time intervals of length T!

2 P. b) Classify the resulting discrete samples according to the 10 quantization intervals.

2 P. c) Outline the resulting sampled and quantized signal!

2 P. d) Mark the resulting quantization errors of the sampled and quantized signal in contrast to the original sound wave!

4 P. **Assignment 4**: *Switching*

Assign the concepts on the left side to the appropriate switching techniques.

	Circuit switching	Connection-less Packet switching	Connection-oriented packet switching
Most tolerant to network failures			
The physical links contained in a path are exclusively used by the communication partners for the whole duration of the communication process.			
Uses datagrams			
A virtual circuit is set up.			
Is employed in the Internet.			
No path is set up prior to transmission			

Solution hints: You can copy the table from the HTML-version of this document (`ea/ea01/ea01.html`) and paste it into the word processor of your choice (e. g. Word, StarOffice). The correct answer should be marked with an "x".

5 P. **Assignment 5**: *TCP/IP Model*

Assign the concepts on the left side to the appropriate TCP/IP layers.

	Network Access	Network	Transport	Application
Routing				
Email client				
Datagrams				
Signals				
Connection of hosts over the Internet				
Reliable connection of hosts over the Internet				
Display of data				
Transmission of frames				

Assignments for Chapter "The Network Access Layer"

Assignment 6: *RS-232 Transmission* **4 P.**

Which of the following statements are true?

a. RS-232 is a synchronous transmission protocol.

b. RS-232 is an asynchronous transmission protocol.

c. RS-232 employs parallel transmission.

d. RS-232 is a broadband transmission technique.

e. RS-232 uses only positive voltages to transmit bits.

f. Start and stop bit designate the begin and end of a transmission.

Assignment 7: *Channel Capacity* **5 P.**

What bandwidth in Hz must a transmission channel with a signal-to-noise ratio of 40dB have to allow transmission with a bit rate of 30 kbit/s (=30000 bits/s)?

Round the solution to the nearest integer (german: ganze Zahl).

Assignment 8: *Access* **4 P.**

Assign the concepts on the left side to the appropriate access technology.

	Basic Access ISDN	Primary Access ISDN	ADSL	V.34	RS-232
Which is the fastest access technology in the downstream?					
Which is the fastest access technology in the upstream?					
Which technology has the shortest range?					
Which technologies employ digital modulation?					

Solution hints: *Procedure*: You can copy the table from the HTML-version of this document (`ea02.html`) and paste it into the word processor of your choice (e. g. Word, StarOffice). The correct answer should be marked with an "x".

4 P. **Assignment 9**: *DSL*

Which of the following technologies includes DSL technology?

a. V.34

b. RS-232

c. ISDN

d. ADSL

e. VDSL

f. Powerline

g. CableTV

4 P. **Assignment 10**: *ISP*

Which of the following statements apply to an Internet Service Provider?

a. Every ISP is also a carrier.

b. Operates points-of-presence

c. Operates access routers

d. Owns physical lines in the local loop

e. Acts as content provider

f. Peers with other ISPs at Internet Exchange Points (IXP)

4 P. **Assignment 11**: *Carrier*

Which of the following statements apply to a telecommunication carrier?

a. Offers dark fiber services

b. Offers web hosting

c. Acts as regulator for the national telecom market

d. Leases capacity on physical transmission lines from an ISP

e. Owns physical transmission equipment

f. Operates web and email servers

Assignments for Chapter "Protocols at the Network Layer"

Assignment 12: *Internet address space* **3 P.**

Assuming you are the provider of a network. You receive the network address 196.125.2.0

a) To which address class belongs this network address? **1 P.**

b) How many different IP addresses are available in your network? **1 P.**

c) How do the IP addresses in your network look like? Give some examples! **1 P.**

Assignment 13: *Host name resolution* **6 P.**

A student from the United Kingdom wants to read the homepage of the departement of Electrical Engineering of the university of Hagen. It can be reached by the following internet address:

www.et-online.fernuni-hagen.de

The student has access to the Internet via the internet service provider "Blue Telecom", which has the domain name "bluetelecom".

a) Draw the structure of the involved domains (and DNS Servers) and make clear how they depend on each other. **3 P.**

b) Write down the complete process which takes place to resolve the host name as mentioned above so that the student's computer is able to contact the host. Describe the DNS system internal queries. (No caching takes place.) **3 P.**

Assignment 14: *IP Fragmentation* **6 P.**

A datagram has a length of 2300 bytes and includes a header of 20 bytes. During its transmission, the datagram reaches a link with a maximum transmission unit size of 1200 bytes.

a) Into how many fragments is the original datagram subdivided? **1 P.**

b) How long is each of these fragments? **2 P.**

2 P. c) Which fragment offset is assigned to the fragments?

1 P. d) How does the receiver recognize that the fragments belong together?

4 P. **Assignment 15**: *Ping*

Try to ping server bonsai at the university of Hagen which has the address 'bonsai.fernuni-hagen.de'

1 P. a) Which commands do you use to perform a ping to the server bonsai?

1 P. b) Which answer do you get?

2 P. c) Explain the elements of the answer!

Assignments for Chapter "Protocols at the Transport Layer"

3 P. **Assignment 16**: *TCP acknowledgement*

Host A and B have established a TCP connection. Host A now wants to transmit 50 bytes of data to host B. The hosts have negotiated a transmission window size of 20 bytes. Explain the steps in transmission and acknowledgement among the hosts.

Assignments for Chapter "Telnet and Remote Utilities"

2 P. **Assignment 17**: What are the main advantages of remote applications in contrast to telnet?

a. easier to use

b. higher level of interoperability

c. simpler to implement

d. supports more options

Assignments for Chapter "IRC"

Assignment 18: Why should one consider chatting with IRC instead of writing e-mails? **2 P.**

a. IRC enables near-real-time-communication

b. An IRC conversation is always secure, e-mail isn't

c. An IRC conversation is more personal than an e-mail

d. IRC provides one-to-many-communication

Assignment 19: An IRC channel is in "+tn" - mode. This means that **2 P.**

a. only normal users are allowed to join

b. no external messages are allowed

c. the server traces all actions

d. only channel operators may change the topic of the channel

Assignment 20: What is likely to happen when there a netsplit occurs? **2 P.**

a. a large number of users is suddenly unavailable

b. you get disconnected from your server

c. your dialup-connection hangs up

d. you lose your channel operator status

Assignment 21: What can be the cause of a so-called "lag" ? **3 P.**

a. a large number of users disconnects from a server at the same time

b. a large number of users carry a conversation

c. the network is congested with other traffic like http / ftp / e-mail

d. a server with a bad connection has a lot of users

e. someone launches a bot that floods the network

f. a lot of users carry DCC conversations and/or file transfers

Assignment 22: Can IRC packets become fragmented as they travel over the network? **2 P.**

a. Yes, since IRC uses TCP/IP

b. No, since IRC uses TCP/IP

c. Yes, since IRC uses UDP

d. No, since IRC uses UDP

e. Fragmentation does not apply to any IRC packets since they are already very small.

3 P. **Assignment 23**: What are the differences between channel operators and IRC operators?

a. there is no difference

b. an IRC operator maintains an IRC network, a channel operator maintains a channel

c. unlike a channel operator, an IRC operator cannot kick anyone from a specific channel

d. unlike an IRC operator, a channel operator cannot exclude anyone from a specific server

e. IRC operators are usually bots while channel operators are not.

f. Channel operators are usually bots while IRC operators are not.

3 P. **Assignment 24**: There was a netsplit and you were channel operator on #linux.ger. Now the netsplit is healed. What could happen?

a. Nothing. I keep my operator status on #linux.ger since statuses are merged upon reconnection.

b. There are suddenly a lot more users on #linux.ger than before

c. Everyone on #linux.ger gets operator status since merging of statusses would take up too much server load

d. A random user on every channel gets operator status, all the others are just regular users.

e. The server asks me whether I want to keep my operator status or not.

3 P. **Assignment 25**: A channel has one user with operator status and 5 without. Now the channel operator quits IRC. What happens?

a. the channel has no operator anymore

b. the first user in the channel list is assigned operator status

c. all users are assigned operator status

d. a random user is assigned operator status

e. the former channel operator has to re-join the channel so that it has an operator again.

f. in a "mode +tn”-channel, the topic cannot be changed anymore.

g. the channel ceases to exist since a channel must have at least one operator.

3 P. **Assignment 26**: A channel operator can

a. give other users operator status on the same channel

b. give other users operator status on any channel

c. kick users out of a channel

d. ban users from a channel preventing them from re-joining

e. kick a user out of IRC

f. change the channel's modes

g. change the "Message of the Day" of a server

Assignments for Chapter "Email"

Assignment 27: *Format of email messages* **8 P.**

a) Explain the purpose of "X-" headers with your own words. **2 P.**

b) Send a message with your favorite email client to the tutors' email address (with subject "email header assignment") and to yourself. Expand your own email that you can see all headers and explain all header fields as listed (include the original headers). **6 P.**

Assignment 28: *SMTP* **10 P.**

Connect to the university computer center's host "bonsai" (`bonsai.fernuni-hagen.de`). Access is only possible via Secure Shell (`ssh`). Unix distributions mostly come with a ssh client. Windows users please download the free client using

`ftp://ftp.cert.dfn.de/pub/tools/net/ssh/SSHSecureShellClient-3.2.3.exe`.

Install and start the `ssh` client. Klick "File", then "Quick Connect". "Host name" is "bonsai.fernuni-hagen.de", "User Name" your account name at bonsai.

For further information about UNIX see

`ftp://ftp.fernuni-hagen.de/pub/pdf/urz-broschueren/broschueren/a0379202.pdf`

Activate the session logging by selecting the menu "File->Log Session". Choose a file name and location for the log file.

The solution for this assignment is the log script created at the start of the `ssh` session. Please copy the relevant lines from the log file to your solution document. The log script should document all essential protocol steps.

a) *Sending email without an email client* **8 P.**
Connect to the SMTP port of `primus.fernuni-hagen.de` using a `telnet` session within the `ssh` session. Send an email to the tutors consisting of at least a "from" line, a "to" line, and a "subject" line; subject ist "email manually sent via the SMTP port". This email shall only contain the line:

`"This is an email manually sent via the SMTP port of bonsai. <name> <reg.no.>"`

in the body. Do not forget to state your name and registration number.

2 P. b) *Confirmation of a known user*
What is the correct email address of the user "sommer" at `bonsai`? Use SMTP to verify.

Solution hints: consult the literature references given in the course unit.

3 P. **Assignment 29**: *MIME*

2 P. a) Explain the "Content-Disposition" header! List three possible values and explain them.

1 P. b) In literature you can also find the term "S/MIME". Do some investigations and explain it shortly. (1 point)

Assignments for Chapter "World Wide Web"

4 P. **Assignment 30**: *Tag and Element*

Explain the difference between the terms HTML *tag* and *element.*

6 P. **Assignment 31**: *SGML DTD*

Listing 1 shows a Shakespearean Sonnet in SGML format.

Write the associated DTD for this Sonnet.

The `author` attribute may be used optionally. Each Sonnet must exactly contain one `title` and one `lines` element. The `lines` element must contain at least one `line` element.

Listing 1: SGML Sonnet

```
<sonnet author="Shakespeare">
 <title>Sonnet III</title>
 <lines>
  <line>Look in thy glass and tell the face thou viewest</line>
  <line>Now is the time that face should form another;</line>
  <line>Whose fresh repair if now thou not renewest,</line>
  <line>Thou dost beguile the world, unbless some mother.</line>
  <line>For where is she so fair whose unear'd womb</line>
  <line>Disdains the tillage of thy husbandry?</line>
  <line>Or who is he so fond will be the tomb,</line>
```

```
  <line>Of his self-love to stop posterity?</line>
  <line>Thou art thy mother's glass and she in thee</line>
  <line>Calls back the lovely April of her prime;</line>
  <line>So thou through windows of thine age shalt see,</line>
  <line>Despite of wrinkles this thy golden time.</line>
  <line>But if thou live, remember'd not to be,</line>
  <line>Die single and thine image dies with thee.</line>
 </lines>
</sonnet>
```

Assignment 32: *HTML* **7 P.**

Write an HTML document presenting the Sonnet in Listing 1.

- The document shall comply to the *HTML 4.0 strict DTD*.
- Use an arbitrary heading element for the title. The title of the Sonnet is also the title of the HTML document.
- Use a paragraph element for each line.
- Group the complete Sonnet with the appropriate HTML element. Apply a `class` attribute with the value "Sonnet" to this element.

Assignment 33: *Cascading Style Sheets* **8 P.**

Write a CSS1 conforming style sheet that leads to a presentation as shown in Fig. 2. The style sheet information is included in the HTML document. The solution is the resulting HTML document consisting of the document created in Section 8.4 plus the style sheet code.

Implement the following properties:

- The document's background color is black.
- The Sonnet is rendered in a blue box with a white solid border. The box has a width of 400px, the border has a width of 2px. The space between the left edge of the browser window and the box is 30px.
- The distance from the box border to the text is 20px on the top, bottom, and left side.
- The font for the title has the following properties: font: Helvetica (or sans-serif), color: white, size: 20px.
- The font for the lines has the following properties: font: Times (or serif), color: white, size: 14px.
- The text of the Sonnet is aligned at the left side.

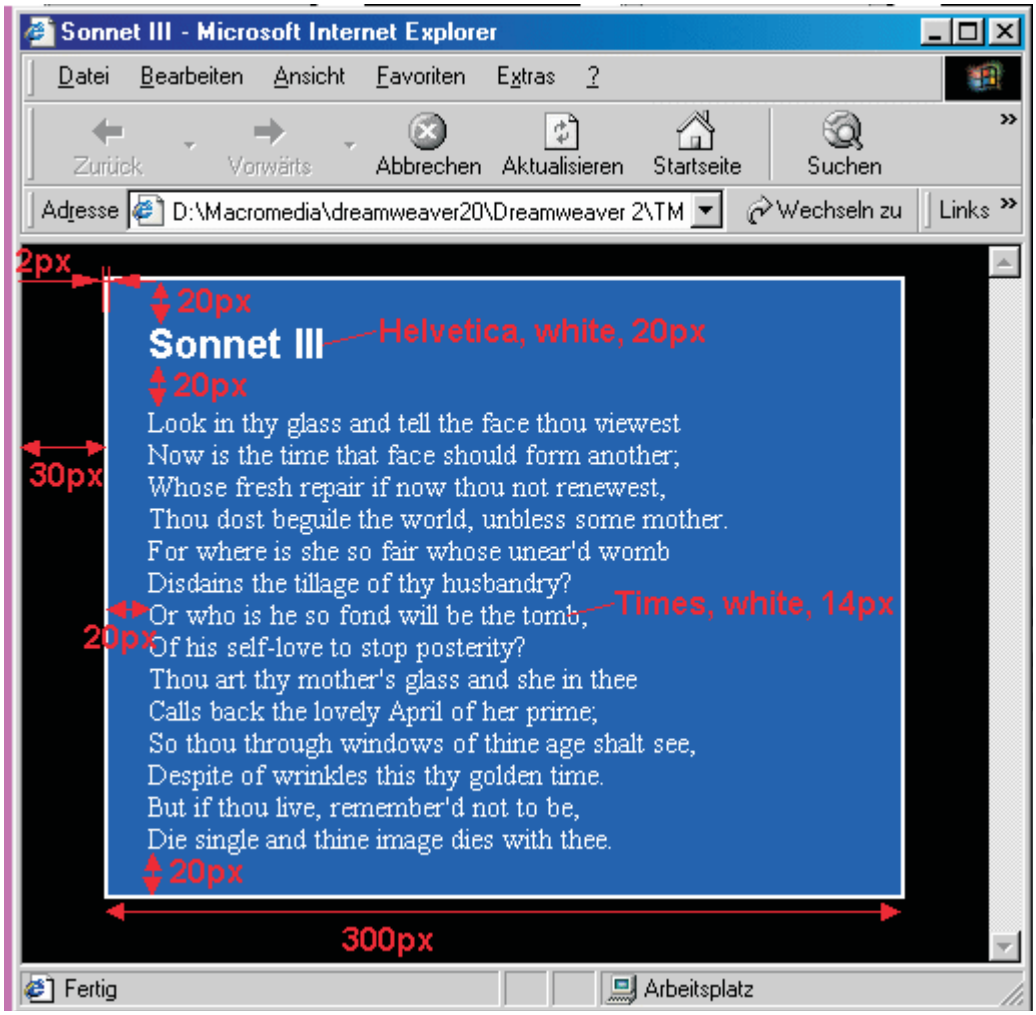


Fig. 2: Presentation of Sonnet

Hints:

- Set the margins of the paragraph element to zero. This will result in the desired line height. Do not use the `line-height` property (because its effect heavily depends on the browser).
- Use the properties of the grouping element to create the box.
- The presentation will depend on the browser you use. If possible use a current version of Amaya, Internet Explorer, Mozilla or Opera because they offer the best CSS support.

7 P. **Assignment 34**: Alice requests an HTML page from the site www.thisandthat.com. She uses "Wonder Browser V1.2" as user agent.

Write the HTTP/1.1 request according to the following feature list:

- The URI for the request is http://www.thisandthat.com/home/welcome.thml
- The browser is willing to accept images of type JPEG, GIF, and PNG.
- The browser's favourite language is Mexican Spanish (es-MX).
- The browser wants to make multiple requests over the same TCP/IP connection.

Assignments for Chapter "Usenet"

Assignment 35: Write a well-formed XML representation of the Email given in Listing 2. Structure the message by employing the following elements: `email`, `header`, `body`. Use further elements to structure the document semantically. **6 P.**

Listing 2: Email

```
From: Anne Nonymous <anne.nonymous@cs.nyu.edu>
To: thomas.demuth@FernUni-Hagen.de
Date: Mon, 10 Jul 2000 16:21:33 +0200
Content-type: text/plain; charset=ISO-8859-1
Content-transfer-encoding: Quoted-printable
Subject: Thanks
Priority: normal

Dear Mr. Demuth,
thank you for sending the research report.
Regards,
Anne Nonymous
```

Assignment 36: Your task is to manually send a news article to the news group **8 P.**

`feu.ice-bachelor.kurs.20018.diskussion`

at the news server "feunews.fernuni-hagen.de" with the subject "Newsgroup article manually sent via th NNTP port", stating your name and registration number. Additionally add a text/joke that you suppose to be funny in the body of the article.

Connect to the university computer center's host "bonsai" (bonsai.fernuni-hagen.de). Access is only possible via Secure Shell (ssh). Unix distributions mostly come with a ssh client. Windows users please download the free client using ftp://ftp.cert.dfn.de/pub/tools/net/ssh/SSHSecureShellClient-3.2.0.exe. Install and start the ssh client. Klick "File", then "Quick Connect". "Host name" is "bonsai.fernuni-hagen.de", "User Name" your account name at bonsai.

Then send the news article using NNTP. Do this using the script command, mentioned in assignment 4, and send the script/log to the email address for exercises (ea.k20018@ks.fernuni-hagen.de).

4 P. **Assignment 37**: Use your favorite news reader to look for the newsgroup at the university's news server handling with the topic "Linux".

What is the name of the newsgroup?

State the subject of one article of the list

Assignments for Chapter "FTP - File Transfer Protocol"

Assignments for Chapter "IPv6"

1 P. **Assignment 38**: Under which aspects do IPv4 and IPv6 differ?

a. Adress field length

b. optional headers

c. maximum payload length (ignore the obsolete IPv4 Jumbo-Payload option.)

2 P. **Assignment 39**: What are the advantages of *optional headers* against one general purpose header?

a. reduction of packet overhead

b. reduction of processing time for routing decision

c. ready for future extensions

d. easier fragmentation

1 P. **Assignment 40**: What is the minimum length of an IPv6 datagram?

a. 32 Bytes

b. 40 Bytes

c. 48 Bytes

4 P. **Assignment 41**: What statements are true related to the fragmentation in IPv4 and IPv6 ?

a. there is no difference

b. IPv4 protocol supports fragmentation

c. IPv6 protocol supports fragmentation

d. IPv6 datagrams can be fragmented at router

e. IPv4 datagrams can be fragmented at router

f. IPv6-Host determines fragmentation according to Path-MTU-size

g. IPv4-Host determines fragmentation according to Path-MTU-size

Assignment 42: What is the maximum number of fragments an IPv6-datagram can be divided into ? **1 P.**

a. 1

b. 4

c. 8

d. 32

e. no limitation

Assignment 43: **5 P.**

a) How many address classes are standardised in IPv6? **1 P.**

a. 4 (A, B, C, D)

b. 1 (Unicast, Multicast)

c. No classes are defined

b) IPv4 introduced address classes of different sizes. Class A, B, C, D: **2 P.**

What statements are true with respect to Class D addresses?

a. amount of addresses is 268435455

b. amount of addresses is 251658240

c. used for Unicast

d. used for Broadcast

e. used for Multicast

f. not used

c) What statements with for the comparison between IPv6 and IPv4 CIDR are true? **2 P.**

a. IPv6 needs classes

b. IPv4 CIDR needs classes

c. IPv6 Unicast addresses consist of Registry ID, Provider ID and subscriber range

d. IPv6 Unicast addresses use only the Format Prefix for routing

e. IPv4 CIDR does not need classes

f. IPv4 CIDR needs classes

Assignment 44: Multicast-Adresses are necessary for: **1 P.**

a. Point-to-point communication

b. Point-to-multipoint communication

c. Point-to-all communication

1 P. **Assignment 45**: Anycast-Adressen are necessary for:

a. Point-to-point communication

b. Point-to-multipoint communication

c. Point-to-all communication

5 P. **Assignment 46**: Assume an overloaded network. V is a video packet with a CoS of 12. M is an E-Mail-packet in the same network. What CoS does M need to have for Router R to prefer packet M instead of packet V ?

a. CoS = 1

b. CoS = 4

c. CoS = 8

d. CoS = 11

e. CoS = 12

f. CoS = 13

4 P. **Assignment 47**: IP-Multicast is an open-group-communication. To distribute identical content to a set of receiver nodes, there are several ways to solve this issue. One way is, establishing n connections to n-receivers. This wastes bandwidth if receiving nodes are neighbours, for example. A more efficient way is, let the router on the passed path decide, when to split identical content to different outgoing interfaces.

Question:

Person P1 sends n times the E-mails with the help of SMTP-Server S1. m E-mails (m<n) are addressed to company C1 and are additionally addressed to mail-server S2.

Which item is correct?

a. S1 establishes m point-to-point connection to S2

b. S1 establishes one multicast connection to S2

c. S1 establishes one point-to-point connection to S2 to transmit m e-mails

d. There is no point-to-point connection at all

e. There is no multicast connection at all

Solutions for Assignments

Solution for Assignment 1:

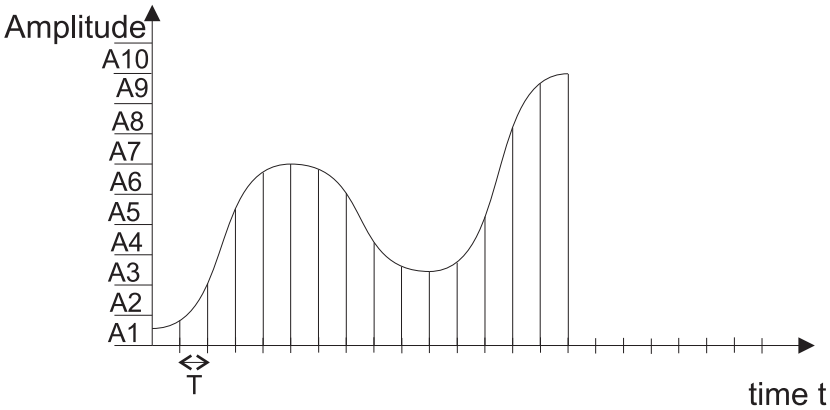
no solution

Solution for Assignment 2:

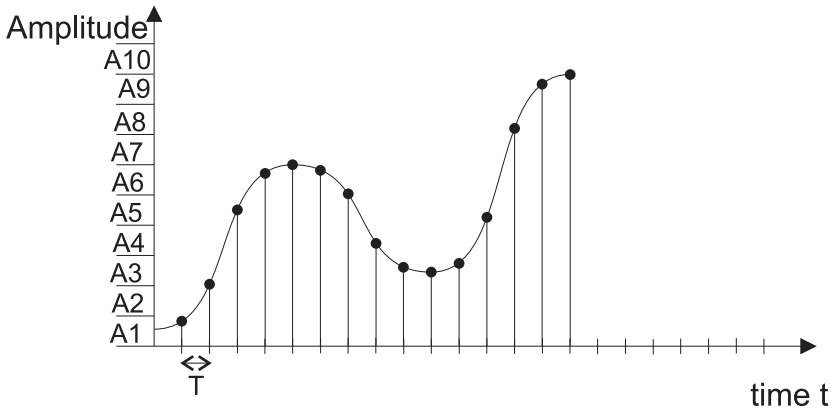
no solution

Solution for Assignment 3:

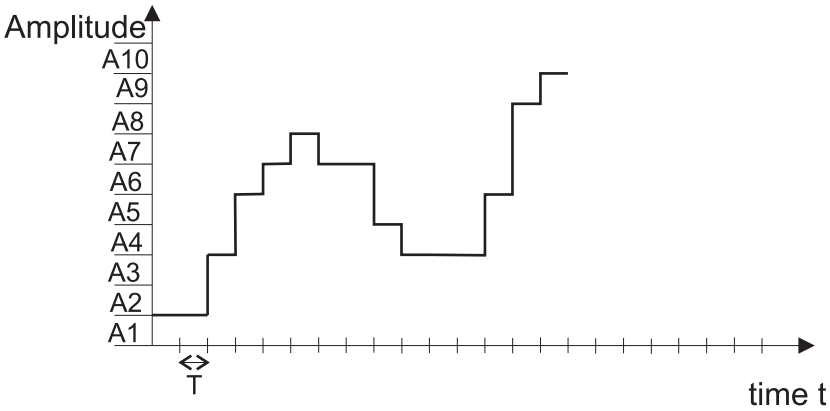
a)



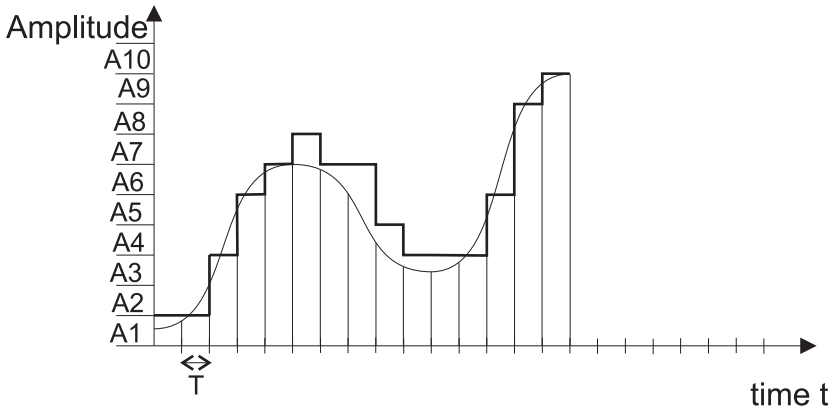
b)



c)



d)



Solution for Assignment 4:

	Circuit switching	Connection-less Packet switching	Connection-oriented packet switching
Most tolerant to network failures		x	
The physical links contained in a path are exclusively used by the communication partners for the whole duration of the communication process.	x		
Uses datagrams		x	
A virtual circuit is set up.			x
Is employed in the Internet.		x	
No path is set up prior to transmission		x	

Solution for Assignment 5:

	Network Access	Network	Transport	Application
Routing		x		
Email client				x
Datagrams		x		
Signals	x			
Connection of hosts over the Internet		x		
Reliable connection of hosts over the Internet			x	
Display of data				x
Transmission of frames	x			

"Reliable connection between hosts" means a connection between any two hosts over the Internet. Hence, the network access layer is not the correct layer for this

task, although it also provides error correction between two nodes which are attached to the same physical link. Two hosts which do not have a direct physical connection (e.g. which are not in the same LAN) have to use the transport layer to reliably exchange data.

Solution for Assignment 6:

b, f

Solution for Assignment 7:

*Solution:*2258 (min: 2257, max: 2258)

$$40dB = 10 \log_{10} \frac{S}{N}$$

$$\frac{S}{N} = 10^4 = 10000$$

$$\begin{aligned} C &= B \log_2(1 + \frac{S}{N}) \\ B &= \frac{C}{log_2(1 + \frac{S}{N})} \\ &= \frac{30000 \text{ bit/s}}{log_2(10001)} \\ &= 2258 \text{ Hz} \end{aligned}$$

Solution for Assignment 8:

	Basic Access ISDN	Primary Access ISDN	ADSL	V.34	RS-232
Which is the fastest access technology in the downstream?			x		
Which is the fastest access technology in the upstream?		x			
Which technology has the shortest range?					x
Which technologies employ digital modulation?			x	x	

Solution for Assignment 9:

c,d,e

Solution for Assignment 10:

b,c,f

Solution for Assignment 11:

a,e

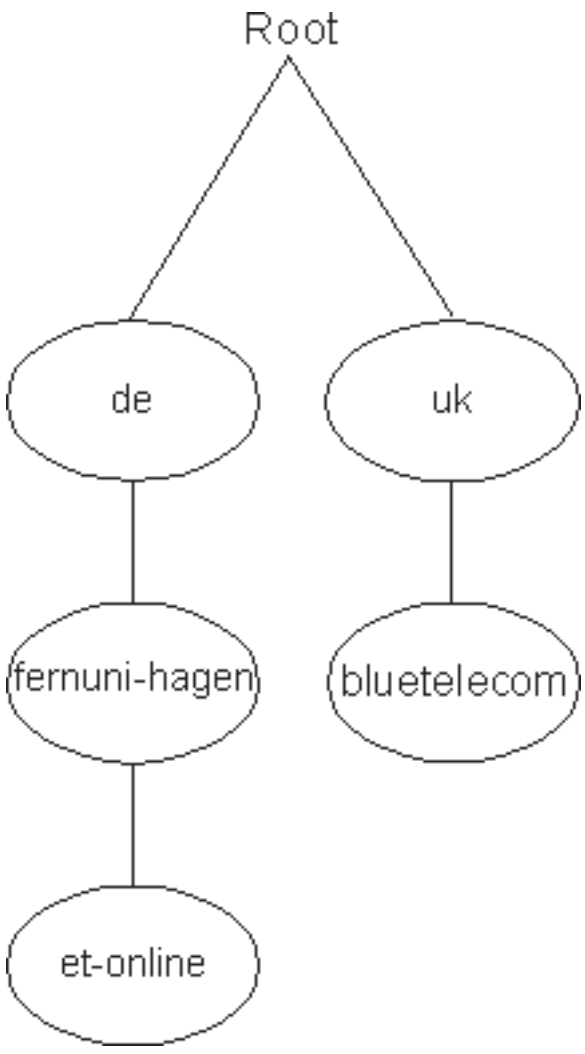
Solution for Assignment 12:

a) The network address belongs to address class C

b) 256, of course, some of them have a special meaning, but these are valid IP addresses, too.

c) 196.125.2.0 196.125.2.1 196.125.2.2 ... 196.125.2.255

Solution for Assignment 13:



a)

b) • The student’s computer contacts the DNS server of the internet service provider "Blue Telecom", domain ’bluetelecom’.

- The DNS server queries the root server for the domain ’uk’.
- The root server of the domain ’uk’ responds with the address of the DNS server of the domain ’.de’.
- The DNS server of Blue Telecom queries the DNS server of the domain ’de’ for the domain ’fernuni-hagen’.
- The root server ’de’ responds with the of the address of the DNS server of the university of Hagen.
- Querying the DNS server of the universiy of Hagen, this one responds with the address of the DNS server of the faculty of Electrical Engineering.
- This DNS server is able to point directly on the corresponding WWW server.
- Now the student’s computer can directly contact the WWW server of the faculy of Electrical Engineering and read the faculty’s homepage.

Solution for Assignment 14:

a) 2 fragments

b) fragment 1: 1196 bytes
fragment 2: 1124 bytes

c) fragment 1: Offset=0
fragment 2: Offset = 147=1180 Bytes /8 Bytes

d) The fragments have the same identification in the IP header.

Solution for Assignment 15:

a) ping bonsai.fernuni-hagen.de

b) The answer may differ in depedence of the internet provider you are using and of your distance from the university of Hagen.

An exemplary answer from bonsai may look like this:

```
Pinging bonsai.fernuni-hagen.de [132.176.114.21] with 32 bytes of data:
Reply from 132.176.114.21: bytes=32 time=1ms TTL=254
Reply from 132.176.114.21: bytes=32 time<10ms TTL=254
Reply from 132.176.114.21: bytes=32 time=1ms TTL=254
Reply from 132.176.114.21: bytes=32 time=1ms TTL=254
```

c) *We receive the IP address of bonsai which is 132.176.114.21, we see that bonsai replies within the duration 1ms up 10ms and we see the time to live of the responding packet which is 254 in this case. So, the TTL exhibits how many network nodes have been passed by the responding packet. In this case, it was only one intermediary node or router.*

Solution for Assignment 16:

- Host A starts sending data, beginning from byte 1 to byte 20.
- Afterwards it awaits an acknowledgement of host B concerning the receipt of these 20 bytes.
- If host B does not acknowledge the receipt of bytes 1-20, Host A waits for a certain time interval and then retransmits the data.
- If host B has acknowledged the receipt of 20 bytes, Host A sends the next 20 bytes starting with number 21 to 40.
- Afterwards host A awaits an acknowledgement of host B concerning the receipt of these bytes 21-40.
- After the receipt of the acknowledgement concerning bytes 21-40, host A transmits the last 10 bytes numbered from 41-50.
- Upon receipt of these data, host B again has to sent an acknowledgement. This last data is less than the window size, but the TCP Header told the receiver that it only contains 10 bytes. Therefore Host B does not wait for more bytes.

Solution for Assignment 17:

c) is correct

Solution for Assignment 18:

a) and d) are correct.

b) An IRC conversation is as secure or insecure as an e-mail since both can be observed by a third party. Only DCC provides a means of almost-secure chatting.

c) An e-mail is as personal or impersonal as a chat conversation.

Solution for Assignment 19:

b) and d) are correct.

Solution for Assignment 20:

a) and b) correct, c) not a netsplit issue and d) as well

Solution for Assignment 21:

b), c), d), e) correct, a) wrong, because disconnects are short messages and after the

receiption the disconnect frees server load. A "lag” does not take place. f) wrong, because DCC means DIRECT client connection, not taking up server load. "lag” refers to the time commands travel between users.

Solution for Assignment 22:

only a) correct

Solution for Assignment 23:

b), c) and d) correct. Both can be bots, but not usually.

Solution for Assignment 24:

a) and b) correct, the others are nonsense.

Solution for Assignment 25:

a) and f) correct, all others incorrect. A channel becomes orphaned without an operator. All users can leave the channel and the first user who re-joins is assigned operator status, but this is not mandatory.

Solution for Assignment 26:

a), c), d), f) correct, the other options are for IRC operators only.

Solution for Assignment 27:

a) According to the RFC 822 transport standard, header fields starting with "X-" will not be used for a message header standard. In result they are in exclusive for user-defined fields’ use. The "X-"-fields are seen as an extention of the standard header by introducing data of the user’s interest.

b)
```
Received: from elm.fernuni-hagen.de (elm.fernuni-hagen.de
[132.176.114.24]) by bonsai.fernuni-hagen.de (8.8.8+Sun/8.8.8)
with ESMTP id VAA28623 for <q5930324@bonsai.fernuni-hagen.de>;
Thu, 9 Nov 2000 21:35:04 +0100 (MET) Received: from
bonsai.fernuni-hagen.de (actually Shiva-HGW-135.fernuni-hagen.de)
by elm.fernuni-hagen.de via local-channel with ESMTP;
Thu, 9 Nov 2000 21:34:48 +0100
```

Received: This field contains the names of the sending and receiving hosts (elm.fernuni-hagen.de, bonsai.fernuni-hagen.de), as well as the time-of-receipt (Thu, 9 Nov 2000 21:35:04 +0100 (MET)). Following "with” the mail protocol is specified (ESMTP). "id” is followed by the internal message identifier (VAA28623). The "for” parameter gives the "original specification” (q5930324@bonsai.fernuni-hagen.de). The "via” parameter indicates the transmission’s physical mechanism (local-channel).

```
Message-ID: <3A0B0CD5.F423CD1F@bonsai.fernuni-hagen.de>
```

Message-ID: To each message in the internet belongs a unique identification number(3A0B0CD5.F423CD1F@bonsai.fernuni-hagen.de).

```
Date: Thu, 09 Nov 2000 21:45:09 +0100 From: aa <q5930324@bonsai.fernuni-hage
```

From: This field contains the sender's email account. It is possible either to fill in the actual operating one, or the one the sender wants to receive replies to.

```
X-Mailer: Mozilla 4.6 [en] (Win95; I) X-Accept-Language: en MIME-Version: 1.
```

MIME-Version: This field declares. with which MIME-Version (1.0) the message was composed.

Solution for Assignment 28:

a) *Sending email without an email client*

```
bash$ telnet primus.fernuni-hagen.de 25
Trying 132.176.12.20...
Connected to primus.fernuni-hagen.de.
Escape character is '^]'.
220 ks.fernuni-hagen.de ESMTP Sendmail 8.11.3/8.11.3/SuSE Linux 8.11.1-0.5;
helo primus.fernuni-hagen.de
250 ks.fernuni-hagen.de Hello bonsai.fernuni-hagen.de [132.176.114.21], plea
mail from: donald.duck@entenhausen.de
250 2.1.0 donald.duck@entenhausen.de... Sender ok
rcpt to: tutor.k20018@ks.fernuni-hagen.de
250 2.1.5 tutor.k20018@ks.fernuni-hagen.de... Recipient ok
data
354 Enter mail, end with "." on a line by itself
from: donald.duck@entenhausen.de
to: tutor.k20018@ks.fernuni-hagen.de
subject: email manually sent via the SMTP port
This is an email manually sent via the SMTP port of primus.
.
250 2.0.0 h4K8Rp616163 Message accepted for delivery
quit
221 2.0.0 ks.fernuni-hagen.de closing connection
Connection closed by foreign host.
bash$
```

b) *Confirmation of a known user*

```
telnet bonsai.fernuni-hageb.de 25
20 bonsai.fernuni-hagen.de ESMTP Sendmail 8.8.8+Sun/8.8.8; Thu, 23 Nov 2000
49:56 +0100 (MET)
VRFY sommer
250 Alfred Sommer <sommer@bonsai.fernuni-hagen.de>
quit
221 bonsai.fernuni-hagen.de closing connection
```

Solution for Assignment 29:

a) The field "Content-Disposition" gives additional information about the MIME-Element. In addition to the filename it figures the way, it is fitted into the message. If it is an "attachment"(out of the message's body) or embedded "inline"(in the message's body) in the email. Another option is extension-token, defined by IANA.

b) S/MIME means Secure/MIME, a version with extended security options.

Solution for Assignment 30:

An element is a structural component of the HTML document. It may consist of start tag, end tag, content, attributes. An element has a relation to other elements in the document, like parent, child and sibling. A tag is a graphical cue for the author to define start and end points for an element. It is possible for an element to exist in an HTML document although no start and end tags are visible.

Solution for Assignment 31:

Listing: DTD for Sonnet

```
<!ELEMENT sonnet     (title, lines) >
<!ELEMENT title      (#PCDATA) >
<!ELEMENT lines      (line+) >
<!ELEMENT line       (#PCDATA) >

<!ATTLIST sonnet
   author CDATA #IMPLIED >
```

Solution for Assignment 32:

Listing: HTML Sonnet

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
          "http://www.w3.org/TR/REC-html40/strict.dtd">
<html>
 <head>
  <title>Sonnet III</title>
 </head>
 <body>
  <div class="Sonnet">
   <h3>Sonnet III</h3>
   <p>Look in thy glass and tell the face thou viewest</p>
   <p>Now is the time that face should form another;</p>
   <p>Whose fresh repair if now thou not renewest,</p>
   <p>Thou dost beguile the world, unbless some mother.</p>
   <p>For where is she so fair whose unear'd womb</p>
   <p>Disdains the tillage of thy husbandry?</p>
   <p>Or who is he so fond will be the tomb,</p>
   <p>Of his self-love to stop posterity?</p>
```

```
   <p>Thou art thy mother’s glass and she in thee</p>
   <p>Calls back the lovely April of her prime;</p>
   <p>So thou through windows of thine age shalt see,</p>
   <p>Despite of wrinkles this thy golden time.</p>
   <p>But if thou live, remember’d not to be,</p>
   <p>Die single and thine image dies with thee.</p>
  </div>
 </body>
</html>
```

Solution for Assignment 33:

Listing: HTML Sonnet with CSS

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
          "http://www.w3.org/TR/REC-html40/strict.dtd">
<html>
 <head>
  <title>Sonnet III</title>
  <style type="text/css">
    body { background-color: black; }
    div.Sonnet{
           width: 400px;
           margin-left: 30px;
           padding: 20px;
           border: solid 2px white;
           background-color: blue;
           color: white;
           }
    h3 {           font-family: sans-serif;
           font-size: 20px;
           margin: 0px 0px 20px 0px;
       }
    p {
           font-family: serif;
           font-size: 14px;
           text-align: left;
           margin: 0px;
      }
  </style>
 </head>
 <body>
  <div class="Sonnet">
   <h3>Sonnet III</h3>
   <p>Look in thy glass and tell the face thou viewest</p>
   <p>Now is the time that face should form another;</p>
   <p>Whose fresh repair if now thou not renewest,</p>
   <p>Thou dost beguile the world, unbless some mother.</p>
   <p>For where is she so fair whose unear’d womb</p>
   <p>Disdains the tillage of thy husbandry?</p>
   <p>Or who is he so fond will be the tomb,</p>
   <p>Of his self-love to stop posterity?</p>
   <p>Thou art thy mother’s glass and she in thee</p>
   <p>Calls back the lovely April of her prime;</p>
```

```
   <p>So thou through windows of thine age shalt see,</p>
   <p>Despite of wrinkles this thy golden time.</p>
   <p>But if thou live, remember'd not to be,</p>
   <p>Die single and thine image dies with thee.</p>
  </div>
 </body>
</html>
```

Solution for Assignment 34:

```
GET /home/welcome.html HTTP/1.1
Accept: image/gif, image/jpeg, image/png, */*
Accept-Language: es-MX
User-Agent: Wonder Browser V1.2
Host: www.thisandthat.com
Connection: Keep-Alive
```

Solution for Assignment 35:

```
<email>
 <header>
  <From>
   <Name>Anne Nonymous</Name>
   <Email>anne.nonymous@cs.nyu.edu</Email>
  </From>
  <To>thomas.demuth@FernUni-Hagen.de</To>
  <Date>Mon, 10 Jul 2000 16:21:33 +0200</Date>
  <Content-type>text/plain; charset=ISO-8859-1</Content-type>
  <Content-transfer-encoding>Quoted-printable</Content-transfer-encoding>
  <Subject>Thanks</Subject>
  <Priority>normal</Priority>
 </header>
 <body>
  <p>Dear Mr. Demuth,</p>
  <p>thank you for sending the research report.</p>
  <p>Regards,</p>
  <p>Anne Nonymous</p>
 </body>
</email>
```

Solution for Assignment 36:

```
demuth@primus:~ > telnet news.fernuni-hagen.de 119
Trying 132.176.114.41...
Connected to oak.FernUni-Hagen.de.
Escape character is '^]'.
200 oak.fernuni-hagen.de InterNetNews NNRP server INN 2.2 21-Jan-1999 ready (posting ok).
post
340 Ok
Newsgroups: feu.ice-bachelor.kurs.20018.diskussion
From: thomas.demuth@fernuni-hagen.de
```

```
Subject: nntp example

foo bar
.
240 Article posted
quit
205 .
Connection closed by foreign host.
demuth@primus:~
```

Solution for Assignment 37:

feu.comp.linux

[time dependent, take a look at the newsgroup]

Solution for Assignment 38:

a), b)

Solution for Assignment 39:

a), b), c)

Solution for Assignment 40:

b)

Solution for Assignment 41:

b), c), e), f)

Solution for Assignment 42:

e)

Solution for Assignment 43:

a) No classes are defined. Instead there are predefined addresses, e.g. Multicast, Broadcast.

b) a) correct: 224.0.0.0 - 239.255.255.255

e) correct

c) correct: c), e)

See Section 11.4 (CIDR - Classless Inter-Domain Routing).

Solution for Assignment 44:

b)

Solution for Assignment 45:

b), a) is a special case of b).

Solution for Assignment 46:

f)

Solution for Assignment 47:

It would be nice, if e-mails would use the multicast feature of networks. E-mail servers do not use multicast. In this special case there would be only a need for one TCP/IP connection from Server S1 to Server S2. Then the mail servers had to multiplex the mails in one connection. Mail servers do not use/provide this feature.

a) e)

References

References for Chapter "Basics of Communication Networks"

[Fei99] Sidnie Feit: TCP/IP- Architecture, Protocols and Implementation with IPv6 and IP Security, Mc Graw Hill, 1999

[Kad] Firoz Kaderali: Digitale Kommunikationstechnik I, Vieweg Verlag

[Los99] Pete Loshin: TCP/IP Clearly Explained, 3rd Edition, Morgan Kaufmann, 1999

[Sta] William Stallings: High-Speed Networks: TCP/IP and ATM Design Principles, Prentice Hall, 199

[Wal91] Jean Walrand: Communication Networks: A First Course, Aksen Associates, 1991

[ISC] Internet Software Consortium, http://www.isc.org

[Boa] http://boardwatch.internet.com/isp/

[ISOC] The history of the Internet: http://info.isoc.org/guest/zakon/Internet/History/HIT.html

References for Chapter "The Network Access Layer"

[Pro94] Proakis, Salehi, Communication Systems Engineering, Prentice-Hall, 1994

[Tan] Tanenbaum, Andrew S., Computer Networks, Prentice-Hall

[Hal96] Halsall, Fred, Data Communications, Computer Networks and Open Systems, Fourth Edition, Addison Wesley, 1996

[Com99] Comer, Douglas E., Computer Networks and Internets, Second Edition, Prentice-Hall, 1999

[Sie96] Siegmund, Gerd, Technik der Netze, 3. Auflage, v. Decker, 1996

References for Chapter "Protocols at the Transport Layer"

[Fei99] Sidnie Feit: TCP/IP- Architecture, Protocols and Implementation with IPv6 and IP Security, Mc Graw Hill, 1999

[Los99] Pete Loshin: TCP/IP Clearly Explained, 3rd Edition, Morgan Kaufmann, 1999

References for Chapter "IRC"

[Fei99] Sidnie Feit: TCP/IP - Architecture, Protocols and Implementation, Signature Edition, Mc-Graw Hill 1999

[Los99] Pete Loshin: TCP/IP Clearly Explained, 3rd Edition, Academic Press, 1999

[CL00] David Caraballo and Joseph Lo. The IRC Prelude, www.irchelp.org/irchelp/new2irc.h January 2000

[Bri97] Aaron Brinton: The IRC Operators Guide, August 1997

[RFC1459] Internet Engineering Task Force IETF: RFC 1459: Internet Relay Chat Protocol, J. Oikarinen & D. Reed. www.ietf.org, May 1993.

[IRC] http://www.irchelp.org/irchelp/security/.

[ZRM94] Klaus Zeuge, Troy Rollo, Ben Mesander: The Client To Client Protocol, http://www.irchelp.org/irchelp/rfc/ctcpspec.html, 1994

[Rol94] Troy Rollo: Direct Client Connection - a description of the DCC Protocol, http://www.irchelp.org/irchelp/rfc/dccspec.html, 1994

[RFC2810] Internet Engineering Task Force IETF: RFC 2810: Internet Relay Chat Protocol: Architecture, C. Kalt, www.ietf.org, April 2000.

[RFC2811] Internet Engineering Task Force IETF: RFC 2811: Internet Relay Chat Protocol: Channel Management, C. Kalt, www.ietf.org, April 2000.

[RFC2812] Internet Engineering Task Force IETF: RFC 2812: Internet Relay Chat Protocol: Client Protocol, C. Kalt, www.ietf.org, April 2000.

[RFC2813] Internet Engineering Task Force IETF: RFC 2813: Internet Relay Chat Protocol: Server Protocol, C. Kalt, www.ietf.org, April 2000.

References for Chapter "Email"

[RFC821] RFC 821, "Simple Mail Transfer Protocol", August 1982, ftp://ftp.ripe.net/rfc/rfc821.txt

[RFC822] RFC 822, "Standard for the Format of ARPA Internet Messages", August 13, 1982, ftp://ftp.ripe.net/rfc/rfc1822.txt

[RFC1425] RFC 1425, "SMTP Service Extensions", February 1993, ftp://ftp.ripe.net/rfc/rfc1425.txt

[RFC1731] RFC 1731, "IMAP4 Authentication Mechanisms", December 1994, ftp://ftp.ripe.net/rfc/rfc1731.txt

[RFC1806] RFC 1806, "Communicating Presentation Information in Internet Messages: The Content-Disposition Header", June 1995, ftp://ftp.ripe.net/rfc/rfc1806.txt

[RFC1939] RFC 1939, "Post Office Protocol - Version 3", May 1996, ftp://ftp.ripe.net/rfc/rfc1939.txt

[RFC2045] RFC 2045, "Multipurpose Internet Mail Extensions: Format of Internet Message Bodies", November 1996, ftp://ftp.ripe.net/rfc/rfc2045.txt

[RFC2046] RFC 2046, "Multipurpose Internet Mail Extensions: Media Types", November 1996,ftp://ftp.ripe.net/rfc/rfc2046.txt

[RFC2047] RFC 2047, "Multipurpose Internet Mail Extensions: Message Header Extensions for Non-ASCII Text", November 1996, ftp://ftp.ripe.net/rfc/rfc2047.txt

[RFC2048] RFC 2048, "Multipurpose Internet Mail Extensions: Registration Procedures", November 1996, ftp://ftp.ripe.net/rfc/rfc2048.txt

[RFC2049] RFC 2049, "Multipurpose Internet Mail Extensions: Conformance Criteria and Examples", November 1996, ftp://ftp.ripe.net/rfc/rfc2049.txt

[RFC2060] RFC 2060, "Internet Message Access Protocol - Version 4rev1", December 1996, ftp://ftp.ripe.net/rfc/rfc2060.txt

References for Chapter "World Wide Web"

[Wil99] Wilde,Erik, Wilde’s WWW - Technical Foundations of the World Wide Web, Springer, 1999

[HTML401] W3C, HTML 4.01 Specification, http://www.w3.org/pub/WWW/TR/html401

[CSS1] W3C, Cascading Style Sheets level 1, http://www.w3.org/pub/WWW/TR/REC-CSS1

[Len97] Lennon, Jennifer A., Hypermedia Systems and Applications, Springer, 1997

References for Chapter "Usenet"

[W3C-1] W3C, XML 1.0 Specification, http://www.w3.org/TR/REC-xml

[W3C-2] W3C, XHTML 1.0 Specification, http://www.w3.org/TR/xhtml1/

[W3C-3] W3C, XSL Specification, http://www.w3.org/Style/XSL/

[Mar99] Marujama, Tamura, Uramoto, XML and Java - Developing Web Applications, Addison-Wesley, 1999

[W3C-4] W3C, Resource Description Framework (RDF), http://www.w3.org/RDF/Overview.

[Har99] Harold, Elliotte Rusty, XML Bible, IDG Books Worldwide Inc., 1999

[Wil99] Wilde,Erik, Wilde's WWW - Technical Foundations of the World Wide Web, Springer, 1999

[Fei98] Feit, Sidnie, TCP/IP, McGraw-Hill,1998

[RFC1630] RFC 1630

[RFC822] RFC 822, "Standard for the Format of ARPA Internet Messages", August 13, 1982, ftp://ftp.ripe.net/rfc/rfc1822.txt

[RFC977] RFC 977, "Network News Transfer Protocol", February, 1986, ftp://ftp.ripe.net/rfc/rfc977.txt

[RFC1036] RFC 1036, "Standard for Interchange of USENET Messages", December 1987, ftp://ftp.ripe.net/rfc/rfc1036.txt

[URL] http://www.faqs.org/faqs/usenet/spam-faq/index.html

References for Chapter "IPv6"

[RFC959] RFC 959, "FILE TRANSFER PROTOCOL (FTP)", October 1985, ftp://ftp.ripe.net/rfc/rfc959.txt

[RFC1752] S. Bradner, A. Mankin, RFC 1752, "The Recommendation for the IP Next Generation Protocol", January 1995.

[RFC1338] V. Fuller, et al, "Supernetting: an Address Assignment and Aggregation Strategy", RFC 1338, June 1992.

[Dee94] S. Deering, "Simple Internet Protocol Plus (SIPP) Specification (128-bit address version)", Internet Draft, July 1994.

[Pos81] J. Postel, "Internet Protocol", RFC-791, September, 1981.

[DH95] S. Deering, R. Hinden, Editors, "Internet Protocol, Version 6 (IPv6) Specification", Internet Draft, March 1995.

[Pos94] J. Postel, "Assigned Numbers", RFC-1700, October 1994.

[Hin95] R. Hinden, Editor, "IP Version 6 Addressing Architecture", Internet Draft, April 1995.

[HUI94] C. Huitema, "The H Ratio for Address Assignment Efficiency" RFC-1715, November 1994.

[RLi95] Y. Rekhter, T. Li, "An Architecture for IPv6 Unicast Address Allocation", Internet Draft, March 1995.

[RLo95] Y. Rekhter, P. Lothberg, "An IPv6 Global Unicast Address Format", Internet Draft, March 1995.

[Tho95] S. Thomson, "IPv6 Address Autoconfiguration", Internet Draft, February 1995.

[CD95] A. Conta, S. Deering, "ICMP for the Internet Protocol Version 6 (IPv6)", Internet Draft, January 1995.

[Atk95a] R. Atkinson, "IPv6 Security Architecture" Internet Draft, March 1995.

[Atk95b] R. Atkinson, "IPng Encapsulating Security Payload (ESP)", Internet Draft, March 1995.

[GN95] R. Gilligan, E. Nordmark, "Transition Mechanisms for IPv6 Hosts and Routers", Internet Draft, March 1995.

[Com98] D.E. Comer, "Computer Networks and Internets", Prentice Hall, February 1998.

[FLYV93] Fuller, V., Li, T., Yu, J. and K. Varadhan, "Classless Inter- Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", RFC 1519, September 1993.

Index