PURBANCHAL UNIVERSITY

MULTIMEDIA APPLICATION

(Compiled Notes)

YEAR-IV *(BCA452CO)* *SEMESTER-II*

Jeevan Poudel

BCA-VIII, 2077

Purbanchal University, Nepal

Multimedia Application (BCA452CO)

(Compiled Notes)

BCA-VIII

Jeevan Poudel

श्री गोमेन्द्र बहुमुखी महाविद्यालय
वितामोड, झापा
चैत ६, २०७७
(2021)

यो पाठ्य सामग्री तयार पार्न साथ, सहयोग र हौसला प्रदान
गर्नुहुने आदरणीय गुरु श्री सिद्धार्थ कायस्था र मेरा सबै
साथीहरुप्रति हार्दिक आभार प्रकट गर्दछु।

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER

1

MULTIMEDIA SYSTEM

1.1 Introduction, Concept and Structure

1.1.1 Introduction

The word multimedia is composed of two parts: the prefix *multi* and the root *media*. The prefix *multi* comes from the Latin word *multus*, which means "numerous".

The root *media* is the plural form of the Latin word *medium*.

In general, *medium* is a means for distribution and presentation of information., Examples of a medium are *text*, *graphics*, *speech* and *music*.
Multimedia is the use of a computer to present and combine text, graphics, audio, and video with links and tools that let the user navigate, interact, create, and communicate.

This definition contains four components essential to multimedia.

1. First, there must be a computer to coordinate what we see and hear, and to interact with.

2. Second, there must be links that connect the information.

3. Third, there must be navigational tools that let us traverse the web of connected information.

4. Finally, there must be ways for us to gather, process, and communicate our own information and ideas.

If one of these components is missing, we do not have multimedia. For example:

- If we have no computer to provide interactivity, we have mixed media, not multimedia.
- If there are no links to provide a sense of structure and dimension, we have a bookshelf, not multimedia.
- If there are no navigational tools to let us decide the course of action, we have a movie, not multimedia.
- If we cannot create and contribute our own ideas, we have a television, not multimedia.

Multimedia Applications

Examples of Multimedia Applications include:

- World Wide Web
- Animation
- 3D Mapping
- Video-on-demand
- Interactive TV
- Computer Games
- Virtual reality
- Digital video editing and production systems
- Image processing
- Voice recognition
- Augmented reality

1.1.2 Concept

Multimedia can have many definitions these include:

A computer system perspective definition

Multimedia means that computer information can be represented through audio, video, and animation in addition to traditional media (i.e., text, graphics/drawings, images).

General Definition

Multimedia is the field concerned with the computer controlled integration of text, graphics, drawings, still and moving images (Video), animation, audio, and any other media where every type of information can be represented, stored, transmitted and processed digitally.

1.1.3 Structure

Figure 1.1 shows the main fields of multimedia systems.

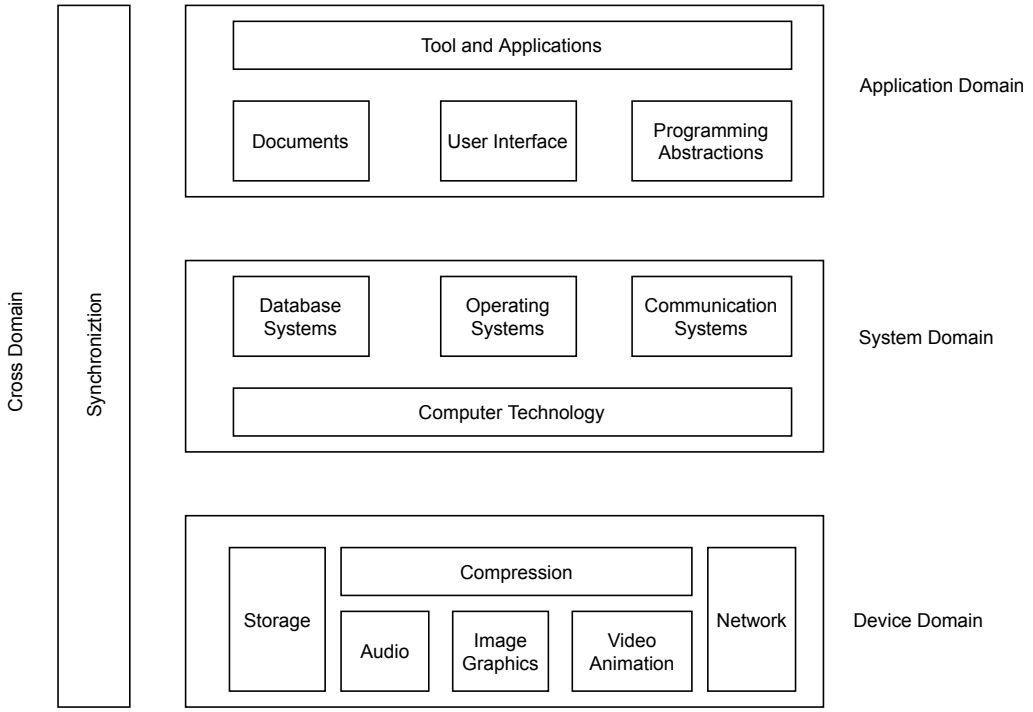

Figure 1.1: Main fields of multimedia systems.

The following areas can be distinguished:

Device Domain

Basic concepts for the processing of *digital audio* and *video* data are based on digital signal processing. Different methods for the processing of *image*, *graphics* and *animation* are included.

System Domain

The interface between the device domain and the system domain is specified by the *computer technology*. To utilize the device domain, several system services are needed. Basically, three services exist. These services are mostly implemented in software:

- The *operating system* serves as an interface between computer hardware/system software and all other software components.
- The *database system* allows structured access to data and a management of large databases.
- The *communication system* is responsible for data transmission according to the timing and reliability requirements of the networked multimedia application.

Application Domain

The services of the system domain are offered to the application domain through proper programming abstractions.

Another application domain is *document handling*. A document consists of a set of structured information, represented in different media, and generated or recorded at the time of presentation.

Many functions of document handling and other applications are accessible and presented to the user through a *user interface*.

Cross Domain

Some aspects, such as synchronization aspects, are difficult to locate in one or two components or domains. The reason is that *synchronization*, being the temporal relationship among various media, relates to many components across all domains.

1.2 Media Aspect Properties

Media can be classified with respect to different criteria as follows:

- perception media
- representation media
- presentation media
- storage media
- transmission media
- information exchange media

1.2.1 Perception Media

Perception media refers to the nature of information perceived by humans. The question to ask here is: *How do humans perceive information*? The answer is that the perception of information occurs mostly through seeing or hearing the information.

We distinguish primarily between what we see and what we hear.

- *Auditory media* include music, sound, and voice.
- *Visual media* include text, graphics, and still and moving pictures.

1.2.2 Representation Media

The term representation media refers to how information is represented internally to the computer. The question to ask here is: *How is information encoded in the computer?* The answer is that various formats are used to represent media information in a computer. There are several options:

- A text character is coded in American Standard Code for Information Interchange (ASCII) or Extended Binary Coded Decimal Interchange Code (EBCDIC) code.
- Graphics are encoded using Graphical Kernel System (GKS) graphics standard.
- An audio stream can be represented using Pulse Code Modulation (PCM)
- An image is in Joint Photographic Experts Group (JPEG) format.
- A combined audio-video sequence is stored in the computer in various TV standards (e.g., Phase Alternate Line (PAL), or National Television Standards Committee (NTSC), or in Moving Picture Experts Group (MPEG) format).

1.2.3 Presentation Media

Presentation media refers to the physical means used by systems to reproduce information for humans. The question to ask here is: *Which medium is used to output information from the computer or input in the computer?*

We distinguish primarily between output and input media:

- *output media*: paper, computer monitors, and loudspeakers, etc.
- *input media*: keyboards, cameras, and microphones, etc.

1.2.4 Storage Media

Storage media refers to various physical means for storing computer data, such as magnetic tapes, magnetic disks, or digital optical disks. However, data storage is not limited to the components available in a computer, which means that paper is also a storage medium. The question to ask here is: *Where is information stored?* Example of storage media includes: *Hard disk*, *Compact Disc (CD)*, *Universal Serial Bus (USB) flash drive*[1], etc.

1.2.5 Transmission Media

Transmission media refers to the physical media. The question to ask here is: *Which medium is used to transmit data?* The answer is that information is transmitted over networks, cables such as fiber or coaxial as well as free airspace transmission for wireless transmission.

1.2.6 Information Exchange Media

Information exchange media includes all data media used to transport information, e.g., all storage and transmission media. The question to ask here is: *Which data medium is used to exchange information between different locations?*

For example, information can be exchanged by storing it on a removable medium and transporting the medium from one location to another. These storage media include microfilms, paper, and USBs, etc.

1.3 Definition of Multimedia System

A Multimedia System is a system capable of processing multimedia data and applications. A Multimedia System is characterized by the processing, storage, generation, manipulation and interpretation of Multimedia information.

Characteristics of Multimedia System:

- They must be computer-controlled.
- They are integrated.
- They must support media independence.
- And lastly, they need to handle discrete and continuous media.

[1]बोलीचालीको भाषामा पेन ड्राइभ (pen drive) पनि भनिन्छ

1.3.1 Discrete and Continuous Media

If the application uses both discrete and continuous media then it is called multimedia. This means that a multimedia application should process at least one discrete and one continuous medium. A word processor with embedded graphics is not a multimedia application.

1.3.2 Independent Media

Media should not be tightly coupled together. They must be independent for example, digital audio and computer available text can be combined for the purpose of presentation.

1.3.3 Computer-Controlled Systems

The independence of media creates a way to combine media for presentation. For this purpose, the computer is the ideal tool. The system can be optionally programmed by a programmer. The simple video recording and playback of media is a system such as video recorder is not sufficient to meet the requirements for computer controlled systems.

1.3.4 Integration

Computer-controlled independent media streams can be integrated to form a global system so that, together, they provide a certain function. A word processor that supports text, spreadsheets, and graphics does not meet the integration criterion unless it allows program-supported references between the data.

1.4 Traditional Data Stream Characteristics

Distributed networked multimedia systems transmit both discrete and continuous media streams. In a digital system, information is split into units (packets) before it is transmitted. These packets are sent by one system component (the *source*) and received by another one (the *sink*). Source and sink can reside on different computers. A data stream consists of a (temporal) sequence of packets.

Packets can carry information from continuous and discrete media.

- Continuous medium: *transmission of voice in a telephone system*
- Discrete medium : *transmission of text file*

When we transmit information originating from various media, we obtain data streams that have very different characteristics. The attributes

a) asynchronous b) synchronous and c) isochronous

are traditionally used in the field of telecommunications to describe the characteristics of a data transmission.

1.4.1 Asynchronous Transmission Mode

A communication is called asynchronous if a sender and receiver do not need to coordinate before data can be transmitted.

- In asynchronous transmission, the transmission may start at any given instant.
- The bit synchronization that determines the start of each bit is provided by two independent clocks on both sender and receiver.
- Example: ASCII terminals attached to host computers. Whenever a character is pressed, a sequence of bits is generated and sent to the computer interface along with character.
- A special signal called *start* signal precedes the information bits.
- Another special signal called *stop* signal follows the last information bit.

1.4.2 Synchronous Transmission Mode

The synchronous transmission mode defines a maximum end-to-end delay for each packet of a data stream. This upper bound will never be violated.

In synchronous transmission, transmission begins when the clock signal is matched with the receiver.

1.4.3 Isochronous Transmission Mode

The isochronous transmission mode defines, besides a maximum end-to-end delay for each packet of a data stream, a minimum end-to-end delay.

- This mode is a form of data transmission in which individual characters are only separated by a whole number of bit-length intervals.

- For example, an end-to-end network connection is said to be isochronous if the bit rate over the connection is guaranteed and if the value of the delay *jitter* is also guaranteed and small.

1.5 Data Stream Characteristics For Continuous Media

Following are the data stream characteristics that relate to any audio/video data transfer in a multimedia system (multimedia data streams).

- The time interval between a complete transmission of consecutive packets.
- Variation of consecutive packet amount.
- Contiguous packets.

1.5.1 The Time Interval Between a Complete Transmission of Consecutive Packets

The first property of data streams relates to the time intervals between fully completed transmissions of consecutive information units or packets. Based on the moment in which the packets become ready, we distinguish between the following variants:

1.5.1.1 Strongly Periodic Data Stream

- When the time interval between neighboring packets is constant, then this data stream is called a *strongly periodic* data stream.
- This also means that there is minimal jitter—ideally zero.
- Example: PCM-encoded voice in telephone systems.

Figure 1.2 illustrates strongly periodic data stream.

1.5.1.2 Weakly Periodic Data Stream

The duration of the time intervals between neighboring packets is often described as a *function with finite period duration.* However, this time interval is not constant between neighboring packets. The data stream is called *weakly periodic.* The case is shown in Figure 1.3.

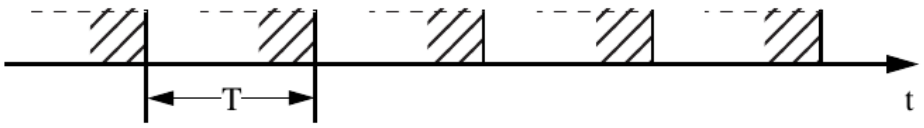

Figure 1.2: Strongly periodic data stream; time intervals have the same duration between consecutive packets.

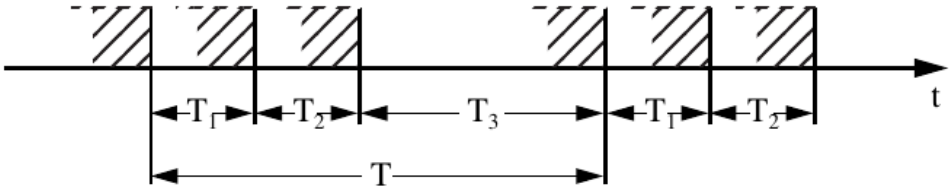

Figure 1.3: Weakly periodic data stream; time intervals between consecutive packets are periodic.

1.5.1.3 Aperiodic Data Stream

All other transmission options are called aperiodic data streams (excluding strongly and weakly data stream), which relates to the sequence of time interval duration, as shown in Figure 1.4.

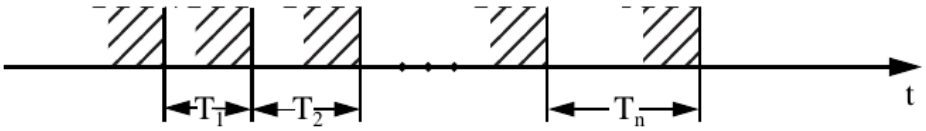

Figure 1.4: Aperiodic data stream; the time interval sequence is neither constant nor weakly periodic.

An example of an aperiodic data stream is a multimedia conference application with a common screen window. Often, the status (left button pressed) and the current coordinates of the mouse moved by another user have to be transmitted to other participants.

1.5.2 Variation of Consecutive Packet Amount

A second characteristic to qualify data streams concerns how the data quantity of consecutive information units or packets varies.

1.5.2.1 Strongly Regular Data Stream

- If the quantity of data remains constant during the entire lifetime of a data stream, then we speak of a *strongly regular data stream.*
- This characteristic is typical for an uncompressed digital audio-video stream.
- Examples are a the video stream taken from a camera in uncompressed form and the audio stream from an audio CD.

Figure 1.5 shows strongly regular data stream.

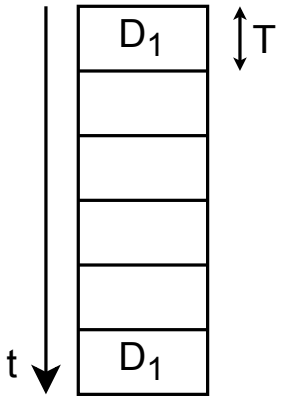

Figure 1.5: Strongly regular data stream; the data quantity is constant in all packets.

1.5.2.2 Weakly Regular Data Stream

- If the quantity of data varies periodically (over time), then this is a *weakly regular data stream.*
- Example: Compressed video stream which uses a compression method such as MPEG.

Figure 1.6 shows an example of weakly regular data stream.

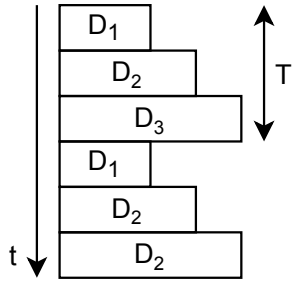

Figure 1.6: Weakly regular data stream; the packets' data stream varies periodically.

1.5.2.3 Irregular Data Stream

Data streams are called irregular when the data quantity is neither constant, nor changing by a periodic function (see Figure 1.7). This data stream is more difficult to transmit and process compared to the variants described earlier.

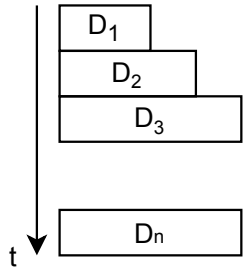

Figure 1.7: Irregular data stream; the packets' data quantity is not constant and does not vary periodically.

When applying a compression method that creates a data stream with a variable bit rate, the size of the single information units (each derived from a single image) is determined from the image content that has changed in respect to the previous image. The size of the resulting information units normally depends on the video sequence and the data stream is irregular.

1.5.3 Contiguous Packets

The third qualification characteristic concerns the continuity or the relationship between consecutive packets. Are packets transmitted progressively, or

is there a gap between packets? This can be seen as utilization of a certain system resource, such as a network.

1.5.3.1 Interrelated/Continuous Data Stream

- All packets are transmitted one after the other without gaps in between.
- Additional information to identify user data is included, e. g. error detection codes.
- In this case specific resource is utilized at 100%.
- Allows maximum throughput and achieves optimum utilization of a resource.

Figure 1.8 shows an interrelated/continuous information transfer.

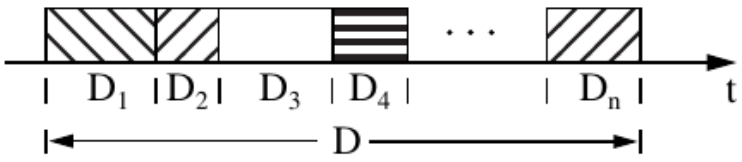

Figure 1.8: Interrelated data stream; packets are transmitted without gaps in between.

1.5.3.2 Non-interrelated/Discrete Data Stream

- The transmission of an interrelated data stream over a higher-capacity channel causes gaps between packets.
- Each data stream that includes gaps between its information units is called a *non-interrelated/discrete* data stream.
- It is not important if gaps exist among all packets or if the duration of gaps varies.

Figure 1.9 shows an example of non-interrelated data stream.

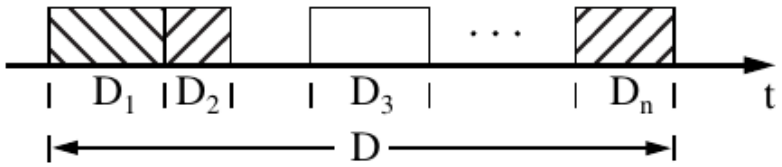

Figure 1.9: Non-interrelated data stream; there are gaps between packets.

1.6 Information Units

Continuous (time-dependent) media consist of a (temporal) sequence of information units. Such an information unit is called a Logical Data Unit (LDU), which is based on Protocol Data Unit (PDU). An LDU's information quantity and data quantities can have different meanings:

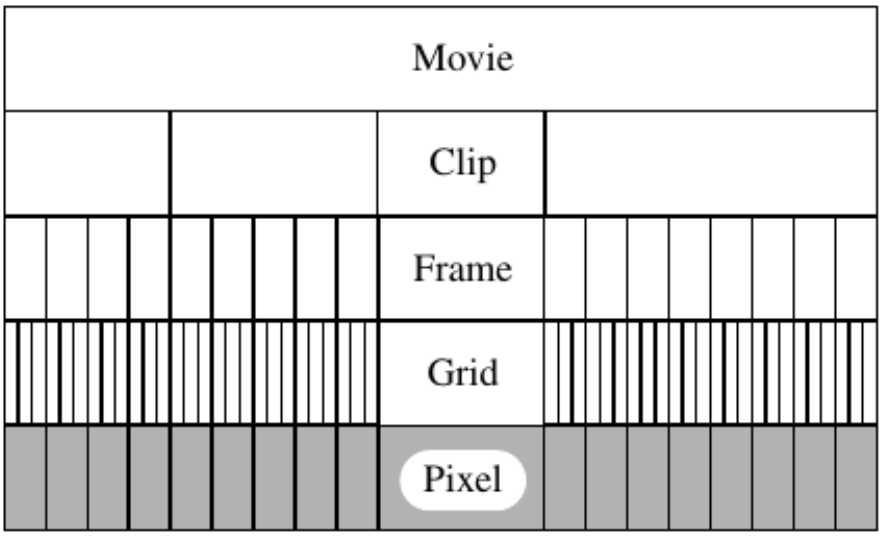

Figure 1.10: Granularity of a motion video sequence showing its LDUs

- In Figure 1.10, we see that the uncompressed video sequence consists of single *clips*, each representing a *scene*.
- Each of these scenes consists of a sequence of single *images*.
- An image can be divided, for example 16×16 groups of *pixels*.

- In turn, each pixel contains a *luminance* value and a *chrominance* value.

This means that a single image is not the only possible LDU in a motion video sequence. A scene or a pixel can also be an LDU. The redundancies in single image sequences of an MPEG-encoded video stream can be used to reduce the data quantity by applying an interframe compression method. In this case, the smallest self-sufficient meaningful units are single-image sequences.

A phenomenon called granularity characterizes the hierarchical decomposition of an audio or video stream in its components. This example uses a motion video to generally describe extensive information units.

We distinguish between closed and open LDUs:

- *Closed LDUs* have a well-defined duration. They are normally stored sequences. Example: data stream of audio samples in the computer.
- In *Open LDUs*, the data stream's duration is not known in advance. Such a data stream is delivered to the computer by a camera, a microphone, or a similar device.

CHAPTER

2

SOUND AND AUDIO

Sound is a physical phenomenon caused by vibration of material, such as a sarangi string (सारङ्गीको तार) or a madal (मादल). This type of vibration triggers pressure wave fluctuations in the air around the material. The pressure waves propagate through the air in a wave-like motion. When a wave reaches the human ear, We hear a sound.

2.1 Sound, Representation and Formats

2.1.1 Basic Sound Concept

Sound is produced by the vibration of matter. During the vibration, pressure variations are created in the air surrounding it. The pattern of this oscillation (see Figure 2.1) is called *waveform*.

This wave form occurs repeatedly at regular *intervals* or *periods*. Sound waves have a natural origin, so they are never absolutely uniform or periodic.

- A sound that has a recognizable periodicity is referred to as *music*.
- Examples of *periodic* sound: sounds generated by musical instruments, vocal sounds, wind sounds, or a bird's twitter.
- Examples of *non-periodic* sounds: drums, coughing, sneezing, etc.

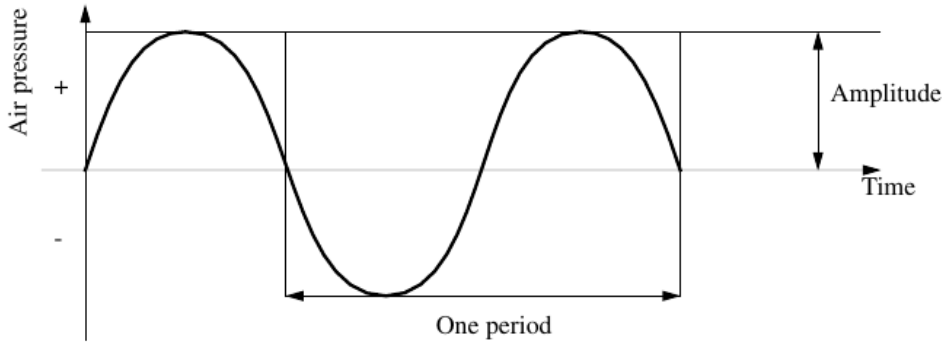

Figure 2.1: Pressure wave oscillation in the air.

2.1.1.1 Frequency

A sound's frequency is the *reciprocal value of its period*. The frequency represents the number of periods per second and is measured in Hertz (Hz) or Cycles Per Second (CPS).

A common abbreviation is Kilohertz (kHz), which describes 1,000 oscillations per second, corresponding to 1,000Hz. Sound processes that occur in liquids, gases, and solids are classified by frequency range:

- Infrasonic: 0 to 20Hz
- Audiosonic: 20Hz to 20kHz
- Ultrasonic: 20kHz to 1GHz
- Hypersonic: 1GHz to 10THz

Multimedia systems make use of sound only within frequency range of human hearing. Sound within human hearing range is called *audio*. The waves in the audiosonic frequency range are called *acoustic signals*.

- Speech is an acoustic signal produced by humans.
- Music signals have a frequency range between 20Hz and 20kHz.
- Beside speech and music, we denote any other audio signal as *noise*.

2.1.1.2 Amplitude

A sound has a property called *amplitude*, which humans perceive subjectively as loudness or volume. The amplitude of a sound is a measuring unit used to deviate the pressure wave from its mean value (idle state).

2.1.2 Sound Representation

The smooth, continuous curve of a sound waveform is not directly represented in a computer. A computer measures amplitude of the waveform at regular time intervals to produce a series of numbers. Each of these measurements is a *sample*. Figure 2.2 shows one period of a digitally sampled wave.

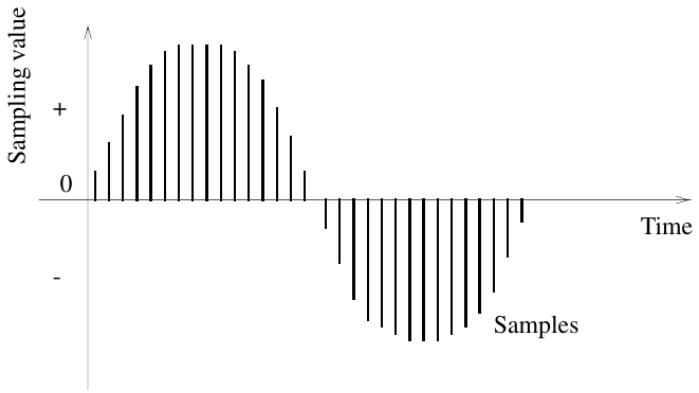

Figure 2.2: Sampling a wave.

The mechanism that converts an audio signal into a sequence of digital samples is called an Analog-to-Digital Converter (ADC) and a Digital-to-Analog Converter (DAC) is used to achieve the opposite conversion.

2.1.3 Sampling Rate

The rate at which a continuous wave form is sampled (see Figure 2.2) is called the *sampling rate*. It is measured in Hz.

For example, CDs are sampled at a rate of 44, 100Hz, which may appear to be above the frequency range perceived by humans. However, the bandwidth in this case, $20,000Hz - 20Hz = 19,980Hz$ that can represent a digitally sampled audio signal is only about half as big as a CD's sampling rate, because CDs use the **Nyquist sampling theorem**[1]. This means that a sampling rate of $44,100Hz$ covers only frequencies in the range from $0Hz$ to $22,050Hz$. This limit is very close to the human hearing capability.

2.1.4 Quantization

The digitization process requires two steps.

[1]Nyquist's Sampling Theorem: The Sampling frequency for a signal must be at least twice the highest frequency component in the signal

- First the analog signal must be sampled. This means that only a discrete set of values is retained at (generally regular) time or space intervals.
- The second step involves quantization. The *quantization* process consists of converting a sampled signal into a signal that can take only a limited number of values.

An 8-bit quantization provides 256 possible values, while a 16-bit quantization in CD quality results in more than $65,536$ possible values. Figure 2.3 shows a 3-bit quantization.

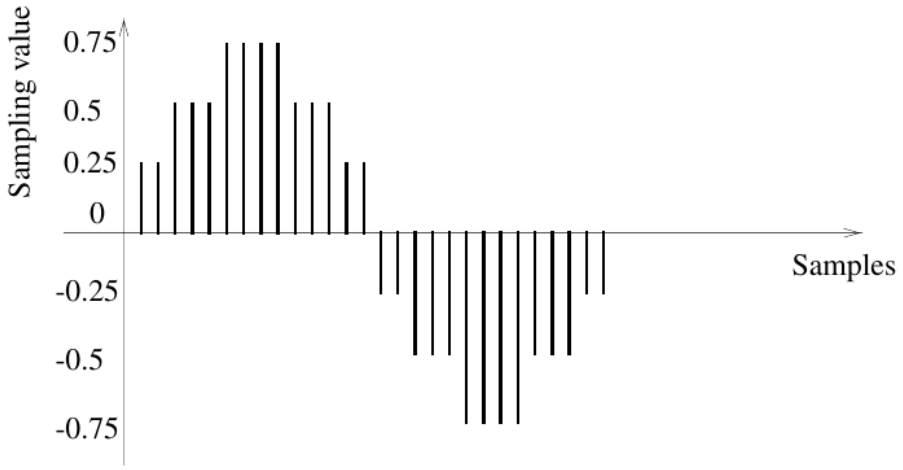

Figure 2.3: 3-bit quantization.

The values transformed by a 3-bit quantization process can accept eight different characteristics: $0.75, 0.5, 0.25, -0.25, -0.5, -0.75, and\ -1$, so that we obtain an "angular-shape" wave. This means that the lower the quantization (in bits), the more the resulting sound quality deteriorates.

2.1.5 Sound Hardware

Before sound can be processed, a computer needs input/output devices. Microphone jacks and built-in speakers are devices connected to an ADC and DAC, respectively for the input and output of audio.

2.1.6 Formats

- Popular audio file formats include:

  - `.au`(Origin: Unix, Sun),
  - `.aiff`(Mac),
  - `.wav` (PC)

- Compression can be utilized in some of the above but is not Mandatory.
- A simple and widely used (by above) audio compression method is Adaptive Delta Pulse Code Modulation (ADPCM).
  - Based on past samples, it predicts the next sample and encodes the difference between the actual value and the predicted value.
- Many formats linked to audio applications.
- Most audio formats use compression.

Audio format defines the quality and loss of audio data. Based on application different type of audio format are used. Audio formats are broadly divided into three parts:

1. *Uncompressed Format.* Example: Waveform Audio File Format (WAV), Audio Interchange File Format (AIFF), AU.
2. *Lossy Compressed format.* Example: MPEG-1 Audio Layer III (MP3), Advanced Audio Coding (AAC), Windows Media Audio (WMA) Lossy.
3. *Lossless Compressed Format.* Example: Free Lossless Audio Codec (FLAC), Apple Lossless Audio Codec (ALAC).

2.2 Basic Music (MIDI)

Music can be described in a symbolic way. On paper, we have the full scores. Computers and electronic musical instruments use a similar technique, and most of them employ the Musical Instrument Digital Interface (MIDI), a standard developed in the early 1980s. The MIDI standard defines how to code all the elements of musical scores, such as sequences of notes, timing conditions, and the instrument to play each note.

The MIDI interface between electronic musical instruments and computers is a small piece of equipment that plugs directly into the computer's serial port and allows transmission of music signals.

2.2.1 MIDI Concepts

MIDI represents a set of specifications used in instrument development so that instruments from different manufacturers can easily exchange musical information. A MIDI interface is composed of two different components:

- **Hardware** to connect the equipment. MIDI hardware specifies the physical connection of musical instruments. It adds a MIDI port to an instrument, it specifies a MIDI cable (that connects two instruments), and processes electrical signals received over the cable.
- **A data format** that encodes information to be processed by the hardware. The MIDI data format does not include the encoding of individual sampling values, such as audio data formats. Instead, MIDI uses a specific data format for each instrument, describing things like the start and end of scores, the basis frequency, and loudness, in addition to the instrument itself.

The MIDI data format is digital and data are grouped into MIDI messages. When a musician plays a key, the MIDI interface generates a MIDI message that defines the start of each score and its intensity. This message is transmitted to machines connected to the system. As soon as the musician releases the key, another signal (MIDI message) is created and transmitted.

2.2.2 Components of a MIDI System

Synthesizer

- It is a sound generator (various pitch, loudness, tone colour).
- A good (musician's) synthesizer often has a microprocessor, keyboard, control panels, memory, etc.

Sequencer

- It can be a standalone unit or a software program for a personal computer.
- It has one or more MIDI INs and MIDI OUTs.

Track

- Track in sequencer is used to organize the recordings.
- Tracks can be turned on or off on recording or playing back.

Channel

- MIDI channels are used to separate information in a MIDI system.
- There are 16 MIDI channels in one cable.
- Channel numbers are coded into each MIDI message.

Timbre

- The quality of the sound, e.g., flute sound, cello sound, etc.
- Multitimbral - capable of playing many different sounds at the same time (e. g. , piano, brass, drums, etc).

Pitch

- Musical note that the instrument plays.

Voice

- Voice is the portion of the synthesizer that produces sound.
- Synthesizers can have many (12, 20, 24, 36, etc.) voices.
- Each voice works independently and simultaneously to produce sounds of different timbre and pitch.

Patch

- The control settings that define a particular Timbre.

2.2.3 MIDI Devices

A computer uses the MIDI interface to control instruments for playout. The computer can use the same interface to receive, store, and process encoded musical data.

- In the MIDI environment, these data are generated on a *keyboard* and played out by a *synthesizer*.
- A typical synthesizer is similar to a regular piano keyboard.
- A *sequencer* is used to buffer or modify these data. In a multimedia application, the sequencer is a computer application.

The heart of any MIDI system is the MIDI *synthesizer* device. A typical synthesizer looks like a simple piano keyboard with a panel full of buttons. Most synthesizers have the following common components:

Sound Generator

- The principal purpose of the generator is to produce an audio signal that becomes sound when fed into a loudspeaker.
- By varying the voltage oscillation of the audio signal, a sound generator changes the quality of the sound – its pitch, loudness and tone – to create wide variety of sounds and notes.

Microprocessor

- The microprocessor communicates with the keyboard to know what notes the musician is playing, and with the control panel to know what commands the musician wants to send to the microprocessor.
- The microprocessor then specifies note and sound commands to the sound generators.

Keyboard

- The keyboard affords the musician's direct control of the synthesizer.
- Pressing keys on the keyboard signals the microprocessor knows what notes to play and how long to play them.

Control Panel

- The control panel controls those functions that are not directly concerned with notes and durations.
- It includes:

  a *slider* that sets the overall volume of the synthesizer,

  a *button* that turns the synthesizer on and off, and

  a *menu* that calls up different patches for the sound generators to play.

Auxiliary Controller

They are available to give more control over the notes played on the keyboard.

Memory

Synthesizer memory is used to store patches for the sound generators and settings on the control panel.

There are many other MIDI devices that augment the standard synthesizer in a MIDI system. Examples are drum machines which specialize in percussion sound's and rhythms, the master keyboard which increases the quality of the synthesizer keyboard, guitar controllers, guitar synthesizers, drum pad controllers and so on.

2.2.4 MIDI Messages

MIDI messages are used by MIDI devices to communicate with each other.

Structure of MIDI Messages

- MIDI message includes a *status byte* and up to two *data bytes*.
- *Status byte*
  - The most significant bit of status byte is set to 1.
  - The 4 low-order bits identify which channel it belongs to.
  - The 3 remaining bits identify the message.
- The most significant bit of data byte is set to 0.

Classification of MIDI Messages

A) **Channel messages**: Messages that are transmitted on individual channels rather than globally to all devices in the MIDI network.

a) **Channel voice messages**

- Instruct the receiving instrument to assign particular sounds to its voice.
- Turn notes on and off.
- Alter the sound of the currently active note or notes.

b) **Channel mode messages**

- Channel mode messages are a special case of the Control Change message. The difference between a Control message and a Channel Mode message is in the first data byte.

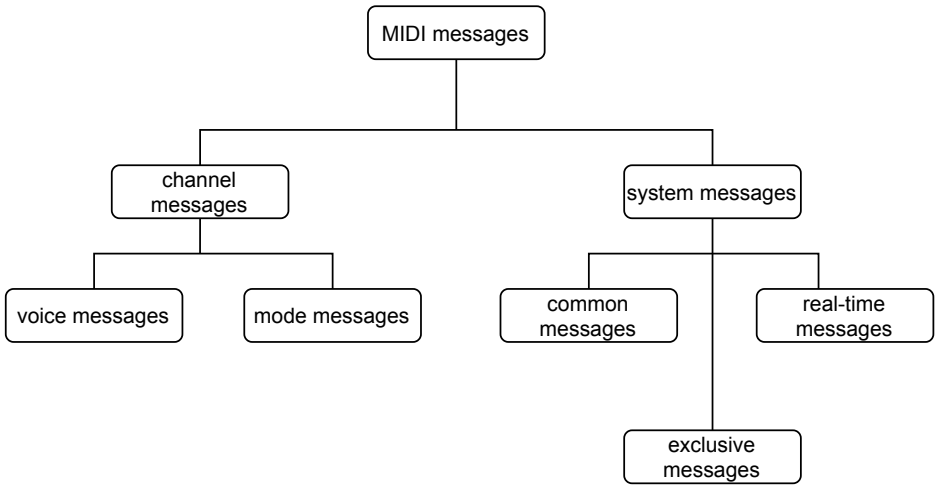

Figure 2.4: Classification of MIDI messages.

- Channel mode messages determine how an instrument will process MIDI voice messages.

B) **System Messages**: System messages carry information that is not channel specific, such as timing signal for synchronization, positioning information in pre-recorded MIDI sequences, and detailed setup information for the destination device.

(a) **System real-time messages**: messages related to synchronization

(b) **System common messages**: are commands that prepare sequencers and synthesizers to play a song. They are used for song selection, tuning the synthesizers etc.

(c) **System exclusive message**:

- Messages related to things that cannot be standardized.
- Addition to the original MIDI specification.
- It is just a stream of bytes, all with their high bits set to 0, bracketed by a pair of system exclusive start and end messages.

2.2.5 MIDI Standards

- The MIDI clock is used by a receiver to synchronize itself to the sender's clock.

- To allow synchronization, 24 identifiers for each quarter note are transmitted.

- Alternatively, the Society of Motion Picture and Television Engineers (SMPTE) timing code can be sent to allow receiver-sender synchronization.

- SMPTE defines a frame format by $hours : minutes : seconds :$, for example 30 $frames/s$. This information is transmitted in a rate that would exceed the bandwidth of existing MIDI connections.

- The MIDI time code is normally used for synchronization because it does not transmit the entire time representation of each frame.

2.2.6 MIDI Software

Once a computer is connected to a MIDI system, a variety of MIDI applications can run on it. Digital computers afford the composer or sound designer unprecedented levels of control over the evolution and combination of sonic events.

The software applications generally fall into four major categories:

Music recording and performance applications

This category of applications provide functions such as recording of MIDI messages as they enter the computer from other MIDI devices, and possibly editing and playing back the messages in performance.

Musical notations and printing applications

This category allows writing music using traditional musical notation. The user can then play back the music using a performance program or print the music on paper for live performance or publication.

Synthesizer patch editors and librarians

These programs allow information storage of different synthesizer patches on the computer's memory and disk drives, and editing of patches on the computer.

Music education applications

These software applications teach different aspects of music using the computer monitor, keyboard and other controllers of attached MIDI instruments.

2.3 Speech

2.3.1 Concept

Speech can be processed by humans or machines. The field of study of the handling of digitized speech is called digital speech processing.

Speech is based on spoken languages, which means that it has a semantic content. Human beings use their speech organs without the need to knowingly control the generation of sounds. Speech understanding means the efficient adaptation to speakers and their speaking habits.

The human speech signal comprises a subjective lowest spectral component known as the *pitch*, which is not proportional to frequency. The human ear is most sensitive in the range of $600Hz$ to $6000Hz$ Speech signals have two important characteristics that can be used by speech processing applications:

- Voiced speech signals have an almost periodic structure over a certain time interval, so that these signals remain *quasi-stationary* for about $30ms$.

- The spectrum of some sounds have characteristic maxima that normally involve up to five frequencies. These frequency maxima, generated when speaking, are called *formants*. By definition, a *formant* is a characteristic component of the quality of an utterance.

Speech Synthesis

Computers can translate an encoded description of a message into speech. This scheme is called *speech synthesis*. A particular type of synthesis is text-to-speech conversion.

Speech recognition is normally achieved by drawing various comparisons. The problems in speech recognition affecting the recognition quality include dialects, emotional pronunciations, and environmental noise.

2.3.2 Speech Generation/Output

A major challenge in speech output is how to generate these signals in real time for a speech output system to be able, for instance, to convert text to speech automatically. Some applications (e. g. , time announcements) handle this task with a limited vocabulary, but most use an extensive if not unlimited vocabulary.

The speech, a machine outputs has to be understandable and should sound natural. In fact, understandability is compulsory and naturalness a nice thing to have to increase user acceptance.

It is important to understand the most important technical terms used in relation to speech output, including:

- Speech basic *frequency* means the lowest periodic signal share in the speech signal. It occurs in voiced sounds.
- A *phoneme* is a member of the set of the smallest units of speech that serve to distinguish one utterance from another in a language or dialect. It is the smallest meaningful linguistic unit but does not carry content.
- *Allophones* specify variants of a phoneme as a function of its phonetic environment.
- A *morpheme* is a meaningful linguistic unit whether in free-form or bound form that contains no smaller meaningful parts. For example, house is a morpheme, while housing is not.
- A *voiced sound* is generated by oscillations of the vocal cords. The characters M, W, and L are examples. Voiced sounds depend strongly on the speaker.
- *Unvoiced sounds* are generated with the vocal cords open, for example, F and S. These sounds are relatively independent of the speaker.

Exactly, there are:

- **Vowels**: a speech sound created by the relatively free passage of breath through the larynx and oral cavity, usually forming the most prominent and central sound of a syllable.
- **Consonants**: a speech sound produced by a partial or complete obstruction of the air stream by any of the various constrictions of the speech organs.

2.3.3 Speech Analysis/Input

Speech analysis/input deals with various applications, as shown in Figure 2.5.

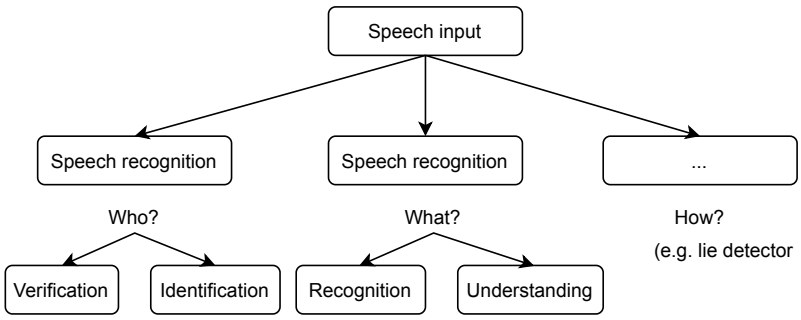

Figure 2.5: Speech input applications.

In the speech input context, we need to ask three questions to obtain correct answers: Who?, What?, and How?

Who?

Human speech has certain characteristics determined by a speaker. Hence, speech analysis can serve to analyze *who* is speaking, i. e. , to *recognize a speaker* for his/her *identification* and *verification*. The computer identifies and verifies the speaker using an acoustic[2] fingerprint.

What?

Another main task of speech analysis is to analyze *what has been said*, i. e. , to recognize and understand the speech signal itself. Based on speech sequence, the corresponding text is generated.

How?

Our third question relates to *how a speech sample should be studied*. For example, a spoken sentence sounds differently if a person is angry or calm. One typical application is a lie detector.

[2]An acoustic fingerprint is a digitally stored speech probe (e. g. , certain statement) of a person.

2.3.3.1 Speech Recognition

In combination with speech synthesis, speech analysis enables us to implement media transformations.

The primary quality characteristic of each speech recognition session is determined by a probability of ≤ 1 to recognize a word correctly. A word is always recognized only with a certain probability. Factors like environmental noise, room acoustics, and the physical and psychical state of the speaker play an important role.

For example, let's assume extremely bad individual word recognition with a probability of 0.95. This means that 5% of the words are incorrectly recognized. If we have a sentence with three words, the probability of recognizing the sentence correctly is $0.95 \times 0.95 \times 0.95 = 0.86$.

This small example shows that a speech recognition system should have a very high single-word recognition rate. Figure 2.6 shows the conceptual components of such a system.

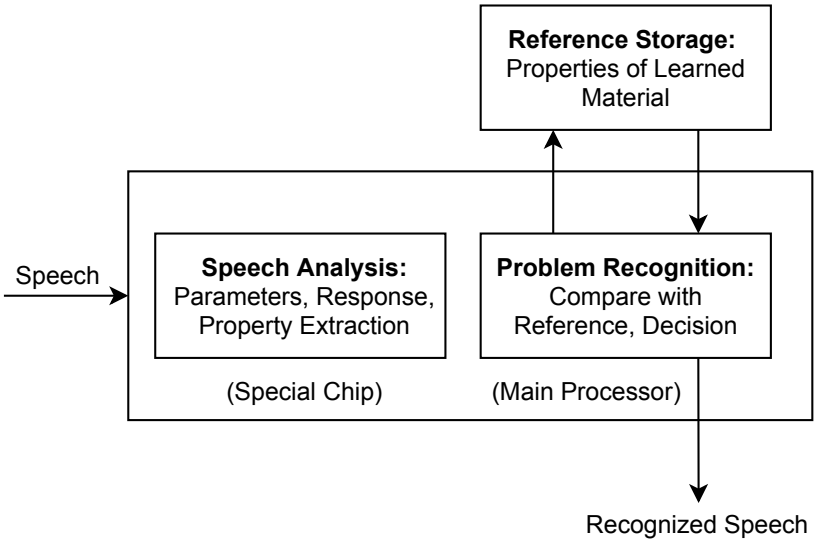

Figure 2.6: The speech recognition principle: the tasks are distributed to system components by the basic principle "extract characteristics to reduce data."

The system is divided into system components according to a basic principle: "Data Reduction Through Property Extraction".

First, speech analysis occurs where properties must be determined. Properties are extracted by comparison of individual speech element characteristics with a sequence of in advance given speech element characteristics. The characteristics are quantified where the concrete speech elements are present.

Second, the speech elements are compared with existent references to determine the mapping to one of the existent speech elements. The identified speech can be stored, transmitted or processed as a parameterized sequence of speech elements.

The principle shown in Figure 2.6 can be applied several times, each time referring to different characteristics. The application of the speech recognition principle can be divided into the steps shown in Figure 2.7.

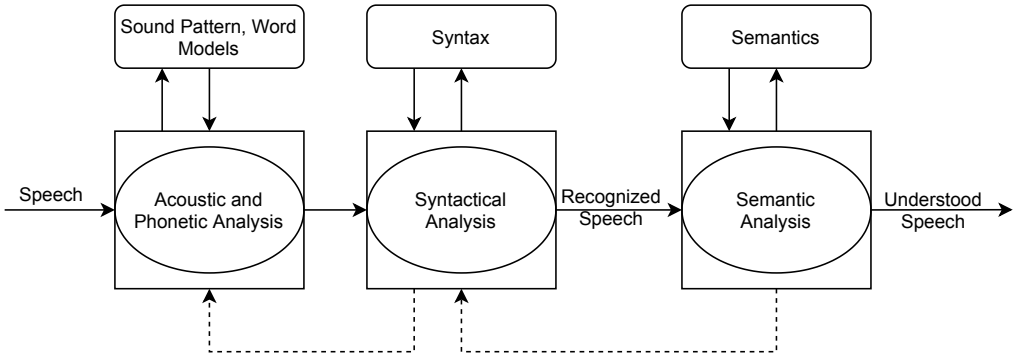

Figure 2.7: Speech recognition components.

The methods applied in the time and frequency ranges are:

Acoustic and phonetic analysis In the first step, the principle is applied to a sound pattern and/or word model. An acoustical and phonetical *analysis is performed.*

Syntactic analysis The second step uses the speech units determined in the first step to run a syntactic analysis on them. This process can detect errors in the first run. It serves as an additional decision tool because the first step does not normally provide a final decision. The result is a *recognized speech.*

Semantic analysis The third step analyzes the semantics of the speech sequence recognized to this point. This step can detect errors from the previous decision process and remove them by using another interplay with other analytical methods. The implementation of this step is extremely difficult. The result of this step is an *understood speech.*
These methods often work with characteristics in the time and/or frequency range. They are based on the same criteria and speech units (e. g. , formants or phonemes) as in speech output.

2.3.4 Speech Transmission

Speech transmission is a field relating to highly efficient encoding of speech signals to enable low-rate data transmission, while minimizing noticeable quality losses. Some principles that are connected to speech generation and recognition are:

Pulse Code Modulation / Signal Form Coding

Signal form encoding does not consider speech-dependent properties or parameters. Here, the goal is to achieve the most efficient coding of the audio signal. The data rate of a PCM-coded stereo-audio signal with CD-quality requirements is:

$$\begin{aligned} rate &= 2 \times \frac{44,100}{s} \times \frac{16bits}{8bits/byte} \\ &= 14,11,200 \times \frac{\cancel{bits}}{s} \times \frac{byte}{\cancel{8bits}} \\ &= 14,11,200 \times \frac{byte}{8s} \\ &= 1,76,400\frac{byte}{s} \\ &= 1,76,400 \times \frac{byte}{s} \times 8 \text{ (bits मा लैजान ८ ले गुणा गरेको)} \\ &= 14,11,200bits/s \end{aligned}$$

As a side note, telephone quality requires only $64Kbit/s$ compared to $1,76,400byte/s$ for the case studied here. Differential Pulse Code Modulation (DPCM) achieves $56Kbit/s$ in at least equal quality, while ADPCM enables a further reduction to $32Kbit/s$.

Source Encoding

Parametric systems use source encoding. They utilize speech-specific characteristics to reduce data, for example the channel vocoder shown in Figure 2.8

A vocoder is an electronic mechanism that reduces speech signals to slowly varying signals that can be transmitted over communication systems of limited frequency bandwidth. A channel vocoder uses an enhanced subband encoding method.

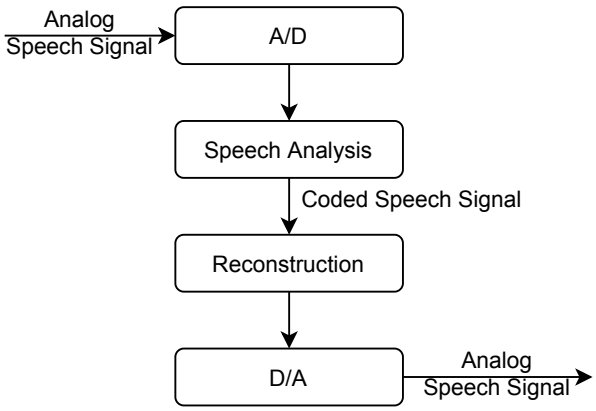

Figure 2.8: Components of a speech transmission system using source encoding.

In addition, the technique utilizes differences between *voiced* and *unvoiced* sounds.

- Unvoiced sounds are generated by means of a noise generator.
- A pulse sequence is selected to generate voiced sounds.

Recognition/Synthesis Methods

There have been attempts to reduce the transmission rate using pure recognition/synthesis methods. Speech analysis (recognition) follows on the sender side of a speech transmission system and speech synthesis (generation) follows on the receiver side (see Figure 2.9)

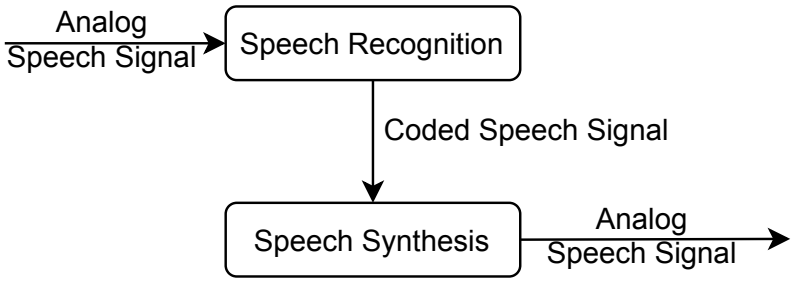

Figure 2.9: Components of a recognition/synthesis system for speech transmission.

Only the speech element characteristics are transmitted, for example formants containing data about the center frequencies and bandwidths for use by digital filters.

Achievable Quality

One of the most important aspects of speech and audio transmission in multimedia systems is the minimal achievable data rate in a defined quality. A data rate of less than $8Kbit/s$ for telephone quality can be achieved.

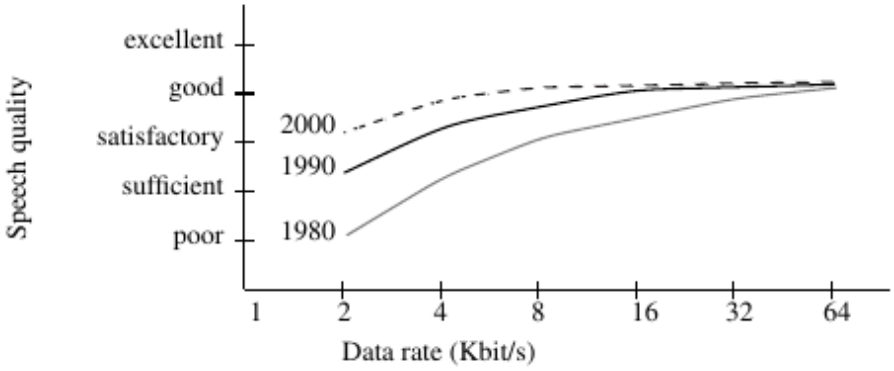

Figure 2.10: Quality of compressed speech in relation to the compressed signal's data rate.

Figure 2.10 relates the audio quality to the number of bits per sampling value. This ratio provides an excellent CD quality at a reduction of $16bits$ per sampling value to $2bits$ per sampling value, which means that only one eighth of the actual data rate is required to achieve this quality.

CHAPTER

3

IMAGE AND GRAPHICS

An image is a spatial representation of an object, a two-dimensional or three-dimensional scene or another image. It can be real or virtual. An image may be abstractly thought of as a continuous function defining usually rectangular region of a plane.

A recorded image may be in a photographic, analog video signal or digital format. In computer vision, an image is usually a recorded image such as a-video-image, digital image or picture. In computer graphics, an image is always a digital image. In multimedia applications, all formats can be presented.

Graphics are normally created in a graphics application and internally represented as an assemblage of objects such as lines, curves, or circles. Attributes such as style, width, and color define the appearance of graphics. We say that the representation is aware of the semantic contents. The objects graphics are composed of can be individually deleted, added, moved, or modified later. In contrast, images can be from the real world or virtual and are not editable in the sense given above. They ignore the semantic contents. They are described as spatial arrays of values.

The smallest addressable image element is called a *pixel*. The array, and thus the set of pixels, is called a *bitmap*.

Drawback of Bitmaps

The drawback of bitmaps is that they need much more storage capacity than graphics.

Advantage of Bitmaps

Their advantage is that no processing is necessary before displaying them, unlike graphics where the abstract definition must be processed first to produce a bitmap.

3.1 Basic Concepts, Digital Image Processing and Format and Graphics Format

3.1.1 Basic Concepts

An image might be thought of as a function with resulting values of the light intensity at each point over a planar region. For digital computer operations, this function needs to be sampled at discrete intervals. The sampling quantizes the intensity values into discrete levels.

Digital Image Representation

A digital image is represented by a matrix of numeric values each representing a quantized intensity value. When I is a two-dimensional matrix, then $I(r, c)$ is the intensity value at the position corresponding to row r and column c of the matrix.

- The points at which an image is sampled are known as *picture elements*, commonly abbreviated as *pixels*.
- The pixel values of intensity images are called *gray scale levels*.
- The intensity at each pixel is represented by an integer and is determined from the continuous image by averaging over a small neighborhood around the pixel location.
- If there are just two intensity values, for example, black and white, they are represented by the numbers 0 and 1; such images are called *binary-valued images*.
- When 8-bit integers are used to store each pixel value, the gray levels range from 0 (black) to 255 (white).

3.1.2 Digital Image Processing

Digital image processing is the use of a digital computer to process digital images through an algorithm. Digital image processing includes the following sub-areas:

- *Image analysis*: Is concerned with techniques for extracting descriptions from images that are necessary for higher level scene analysis methods.
- *Image recognition*: Is concerned with the techniques for recovering information about objects in the image.
- *Image enhancement*: Is concerned with the technique to improve the image and to correct some defects, such as:

  colour and tonal adjustment

  Transformation e. g. , scale, rotate

  Special effects e. g. , texture, stylize, blur, sharpen

3.1.3 Image and Graphics Formats

Most image formats incorporate some variation of a compression technique due to the large storage size of image files. Compression techniques can be classified into either *lossless* or *lossy*.

A digital image consists of many picture elements, termed *pixels*. The number of pixels that compose a monitor image determine the quality of the image (*resolution*). Higher resolution always yields better quality.

A bit-map representation stores the graphic/image data in the same manner that the computer monitor contents are stored in video memory.

Monochrome/Bit-Map Images

- Each pixel is stored as a single bit (0 or 1)
- Dithering is often used for displaying monochrome images

Gray-scale Images

Each pixel is usually stored as a byte (value between 0 to 255)

8-bit Colour Images

- One byte for each pixel
- Supports 256 out of the millions s possible, acceptable colour quality
- Requires Colour Look-Up Tables (LUTs)[1]

24-bit Colour Images

- Each pixel is represented by three bytes (e.g., RGB)
- Supports $256 \times 256 \times 256$ possible combined colours $(1, 67, 77, 216)$
- Most 24-bit images are 32-bit images, the extra byte of data for each pixel is used to store an alpha value representing special effect information

3.1.3.1 Image Formats

There are different kinds of image formats. Here we consider the image format that comes out of an image frame grabber, i. e. , the *captured image format*, and the format when images are stored, i. e. , the *stored image format*.

Captured Image Format The format of an image is defined by two parameters:

- *spatial resolution*, indicated in $pixels \times pixels$ and
- *color encoding*, measured in *bits per pixel*.

The values of both parameters depend on the hardware and software used to input and output images.

Stored Image Format To store an image, the image is represented in a *two-dimensional matrix*, in which each value corresponds to the data associated with one image pixel. In bitmaps, these values are binary numbers. In color images, the values can be one of the following:

- Three numbers that normally specify the intensity of the *red*, *green*, and *blue* components.

[1] A color loop-up table (LUT) is a mechanism used to transform a range of input colors into another range of colors.

- Three numbers representing references to a table that contains the red, green, and blue intensities.
- A single number that works as a reference to a table containing color triples.
- An index pointing to another set of data structures, which represents colors.
- Four or five spectral samples for each color.

When storing an image, information about each pixel, i. e. , the value of each color channel in each pixel, has to be stored. Additional information may be associated to the image as a whole, such as width and height, depth, or the name of the person who created the image. The necessity to store such image properties led to a number of flexible formats, such as RIFF (Resource Interchange File Format), or BRIM (derived from RIFF), which are often used in database systems. RIFF includes formats for bitmaps, vector drawings, animation, audio, and video. In BRIM, an image consists of width, height, authoring information, and a history field specifying the generation process or modifications.

The most popular image storing formats include:

- PostScript,
- GIF,
- JPEG,
- X11 BMP,
- TIFF, and
- BMP.

PostScript PostScript is a fully fledged programming language optimized for printing graphics and text (whether on paper, film, or CRT). It was introduced by Adobe in 1985. The main purpose of PostScript was to provide a convenient language in which to describe images in a device-independent manner. This device independence means that the image is described without reference to any specific device features (e. g. , printer resolution) so that the same description could be used on any PostScript printer without modification.

Graphics Interchange Format (GIF) The Graphics Interchange Format (GIF) was developed by CompuServe Information Service in 1987. Three variations of the GIF format are in use. The original specification, GIF87a,

became a de facto standard because of its many advantages over other formats. GIF images are compressed to, 20 to 25 percent of their original size with no loss in image quality using a compression algorithm called *LZW*.

Tagged Image File Format (TIFF) The Tagged Image File Format (TIFF) was designed by Aldus Corporation and Microsoft in 1987 to allow portability and hardware independence for image encoding. It has become a de facto standard format. It can save images in an almost infinite number of variations. TIFF documents consist of two components:

- The baseline part describes the properties that should support display programs.
- The second part are extensions used to define properties, that is, the use of the CMYK color model to represent print colors.

3.1.3.2 Graphics Formats

X11 Bitmap (XBM) and X11 Pixmap (XPM) X11 Bitmap (XBM) and X11 Pixmap (XPM) are graphic formats frequently used in the UNIX world to store program icons or background images. These formats allow the definition of monochrome (XBM) or color (XPM) images inside a program code. The two formats use no compression for image storage.

Bitmap (BMP) BMP files are device-independent bitmap files most frequently used in Windows systems. The BMP format is based on the RGB color model. BMP does not compress the original image. The BMP format defines a header and a data region. The header region (BITMAPINFO) contains information about size, color depth, color table, and compression method. The data region contains the value of each pixel in a line.

Valid color depth values are 1, 4, 8, and 24. The BMP format uses the run-length encoding algorithm to compress images.

3.2 Image Processing Fundamentals, Synthesis, Analysis, and Transformation

Computer graphics deal with the graphical synthesis of real or imaginary images from computer-based models. In contrast to this technique, image processing involves the opposite process, that is, the analysis of scenes, or the reconstruction of models from images representing 2D or 3D objects.

The following sections describe some image analysis (image recognition) and image synthesis (image generation) basics.

3.2.1 Image Analysis

Image analysis involves techniques to extract descriptions from images, which are required by methods used to analyze scenes on a higher level. Techniques applied to analyzing images include:

- the calculation of perceived colors and brightness,
- a partial or full reconstruction of three-dimensional data in a scene, and
- the characterization of the properties of uniform image regions.

Some image processing fields include:

- image improvement,
- pattern discovery and recognition,
- scene analysis, and
- computer vision.

Image improvement is a technique to improve the image quality by eliminating noise (due to external effects or missing pixels), or by increasing the contrast.

Pattern discovery and pattern recognition involve the discovery and classification of standard patterns and the identification of deviations from these patterns. An important example is OCR (Optical Character Recognition) technology, which allows efficient reading of print media, typed pages, or handwritten pages into a computer.

Scene analysis and computer vision concern the recognition and reconstruction of 3D models of a scene consisting of various 2D images. A practical example is an industrial robot that measures the relative sizes, shapes, positions, and colors of objects.

3.2.1.1 Image Recognition

The complete process of recognizing objects in an image implies that we recognize a match between the sensorial projection (e. g. , by a camera) and the observed image. How an object appears in an image depends on the spatial configuration of the pixel values. The following conditions have to be met for the observed spatial configuration and the expected projection to match:

- The position and the orientation of an object can be explicitly or implicitly derived from the spatial configuration.
- There is a way to verify that the derivation is correct.

Image recognition involves a number of different steps to successively transform object data into recognition information. A recognition method should include the following six steps:

1. image formatting
2. conditioning
3. marking
4. grouping
5. extraction and
6. matching

These steps are shown schematically in Figure 3.1.

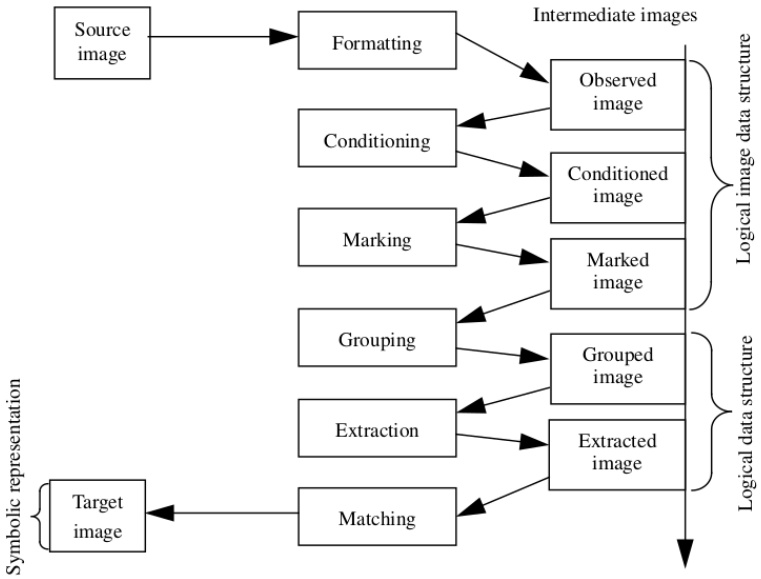

Figure 3.1: Steps involved in image recognition.

The Image-Recognition Procedure/Steps The formatting step shoots an image by use of a camera and transforms the image into digital form (i. e. , pixels). The conditioning, marking, grouping, extraction, and matching steps form a canonical division of the image recognition problem, where each step prepares and transforms the data required for the next step. Depending on the application, it may be necessary to apply this sequence to more than one level of recognition and description. These five steps are:

Step 1: *Formatting*: The formatting step shoots an image by use of a camera and transforms the image into digital form (i. e. , pixels).

Step 2: *Conditioning*:

- Conditioning is based on a model that assumes an image that can be observed is composed of information patterns, which are disturbed by irrelevant variations.
- Conditioning estimates the information pattern based on the observed image, so that noise can be suppressed.
- Also, conditioning can normalize the background by ignoring irrelevant systematic or patterned variations.

Step 3: *Marking/Labeling*:

- Marking is based on a model that assumes that information patterns have a structure in the form of spatial object arrangements, where each object is a set of interconnected pixels.
- Marking determines to which spatial objects each pixel belongs.

Step 4: *Grouping*:

- The grouping operation identifies objects marked in the previous step by grouping pixels that are part of the same object.
- The grouping step includes the edge connection step.
- A grouping operation that groups edges into lines is also called *line fitting*.
- The grouping operation changes the logical data structure.
- The original images, the conditioned, and marked images are all available as digital image data structures.

Step 5: *Extraction*:

- The grouping operation defines a new set of units, but they are incomplete because they have an identity but no semantic meaning.
- The extraction operation calculates a list of properties for each pixel group.

- Such properties can include center (gravity), surface, orientation, spatial moments, spatial grayscale moments, and circles.

Step 6: *Matching*:

- When the extraction operation is finished, the objects occurring in an image are identified and measured, but they have no content meaning.
- We obtain a content meaning by attempting an observation-specific organization in such a way that a unique set of spatial objects in the segmented image results in a unique image instance of a known object.
- The matching operation determines how to interpret a set of related image objects, to which a given object of the three-dimensional world or a two-dimensional form is assigned.
- The classical method is template matching, which compares a pattern against stored models (templates) with known patterns and selects the best match.

3.2.2 Image Synthesis

Image synthesis is the process of creating new images from some form of image description. The kinds of images that are typically synthesized include:

- *Test Patterns*, Scenes with simple two-dimensional geometric shapes.
- *Image Noise*, Images containing random pixel values, usually generated from specific parameterized distributions.
- *Computer Graphics*, Scenes or images based on geometric shape descriptions. Often the models are three-dimensional, but may also be two-dimensional.

Synthetic images are often used to verify the correctness of operators by applying them to known images. They are also often used for teaching purposes, as the operator output on such images is generally 'clean', whereas noise and uncontrollable pixel distributions in real images make it harder to demonstrate unambiguous results. The images could be binary, gray level or color.

Image synthesis is an integral part of all computer-supported user interfaces and a necessary process to visualize 2D, 3D, or higher-dimensional objects. A large number of disciplines, including education, science, medicine, construction, advertising, and the entertainment industry, rely heavily on graphical applications, for example:

- **User interfaces** Applications based on the Microsoft Windows operating system have user interfaces to run several activities simultaneously and offer point-and-click options to select menu items, icons, and objects on the screen.
- **Office automation and electronic publishing** The use of graphics in the production and distribution of information has increased dramatically since desktop publishing was introduced on personal computers. Both office automation and electronic publishing applications can produce printed and electronic documents containing text, tables, graphs, and other types of drawn or scanned graphic elements.
- **Simulation and animation for scientific visualization and entertainment** Animated movies and presentations of temporally varying behavior of real and simulated objects on computers have been used increasingly for scientific visualization. For example, they can be used to study mathematical models of phenomena, such as flow behavior of liquids, relativity theory, or nuclear and chemical reactions. Cartoon actors are increasingly modeled as three-dimensional computer-assisted descriptions.

3.2.3 Image Transmission

Image transmission takes into account transmission of digital images through computer networks. There are several requirements on the networks when images are transmitted:

- The network must accommodate bursty data transport because image transmission is bursty (the burst is caused by the large size of the image).
- Image transmission requires reliable transport.
- Time-dependence is not a dominant characteristic of the image in contrast to audio/video transmission.

Image size depends on the image representation format used for transmission. There are several possibilities:

Raw image data transmission

- In this case, image is generated through a video digitizer and transmitted in its digital format.
- The size can be computed in the following manner:

$$Size = spatial\_resolution \times pixel\_resolution$$

- For example, the transmission of an image with a resolution of 640×480 pixels and pixel quantization of 8 bits per pixel requires transmission of $3,07,200 bytes$ through the network.

Compressed image data transmission

- In this case, the image is generated through a video digitizer and compressed before transmission.
- Methods such as JPEG or MPEG are used to downsize the image.
- The reduction of image size depends on the compression method and compression rate.

Symbolic image data transmission

- In this case, the image is represented through symbolic data representation as image primitives (e.g. 2D or 3D geometric representation), attributes and other control information.
- This image method is used in computer graphics.
- Image size is equal to the structure size, which carries the transmitted symbolic information of the image.

CHAPTER

4

VIDEO AND ANIMATION

Video data can be generated in two different ways:

- by recording the real world and
- through synthesis based on a description

4.1 Basic Video Concepts

The human eye is the human receptor for taking in still pictures and motion pictures. Its inherent properties determine, in conjunction with neuronal processing, some of the basic requirements underlying video systems.

4.1.1 Video Signal Representation

In conventional black-and-white television sets, the video signal is usually generated by means of a Cathode Ray Tube (CRT). An electron beam carries corresponding pattern information, such as intensity in a viewed scene.

The representation of a video signal comprises three aspects:

- visual representation
- transmission, and

- digitization.

4.1.1.1 Visual Representation

A key goal is to present the observer with as realistic as possible a representation of a scene. In order to achieve this goal, the television picture has to accurately convey the spatial and temporal content of the scene. Important measures for this are:

1. *Vertical details and viewing distance*

   The geometry of a television image is based on the ratio of the picture width W to the picture height H. This width-to-height ratio is also called the aspect ratio. The conventional aspect ratio (for television) is $4/3 = 1.33$. Figure 4.1 shows an example of this ratio.

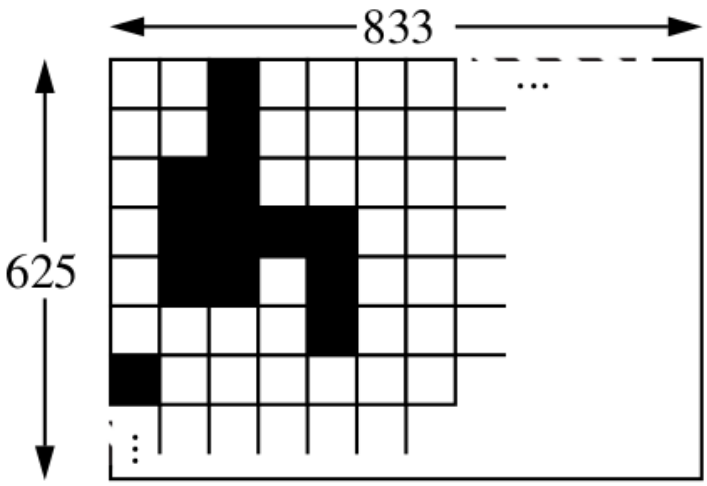

Figure 4.1: Decomposition of a motion picture.

   The viewing distance D determines the angular field of view. This angle is usually calculated as the ratio of the viewing distance to the picture height (D/H).

2. *Horizontal detail and picture width*

   The picture width normally used for television is ($\frac{4}{3} \times$ *picture height*) i. e. (4/3 times the picture height). The horizontal field of view can be determined using the aspect ratio.

3. *Total detail content of a picture*

   The vertical resolution is equal to the number of picture elements of the picture height, while the number of horizontal picture elements is equal to the product of the vertical resolution and the aspect ratio.

   The product of the picture's elements vertically and horizontally is the total number of picture elements in the image.

   However, in the case of television pictures, not all lines (and columns) are visible to the observer. The invisible areas are often used to transmit additional information.

4. *Depth perception*

   In nature, humans perceive the third dimension, depth, by comparing the images perceived by each eye, which view from different angles. In a flat television picture, a considerable portion of depth perception is derived from the perspective appearance of the subject matter.

   Further, the choice of the focal length of the camera lens and changes in depth of focus influence depth perception.

5. *Luminance and Chrominance*

   Color perception is achieved by three signals, proportional to the relative intensities of *red*, *green*, and *blue* light (RGB) present in each portion of the scene. These are conveyed to the monitor separately and the tube reproduces them at each point in time (unlike a camera). Often a different signal division is used for transmission and storage:

   - one brightness signal (luminance), and
   - two color difference signals (chrominance).

6. *Temporal aspects of illumination*

   This property is used in television, in films, and for video data in computer systems. The impression of motion is created by presenting a rapid succession of barely differing still pictures (frames). Between frames, the light is cut off briefly.

   Two conditions must be met in order to represent a visual reality through motion pictures.

   - First, the rate of repetition of the images must be high enough to ensure continuity of movements (smooth transition) from frame to frame.

- Second, the rate must be high enough that the continuity of perception is not disrupted by the dark intervals between pictures.

7. *Continuity of motion*

   It is known that continuous motion is only perceived as such if the frame rate is higher than 15 frames per second. To make motion appear smooth, at least 30 frames per second must be used if the scene is filmed by a camera and not generated synthetically.

8. *Flicker*

   If the refresh rate is too low, a periodic fluctuation of the perceived brightness can result. This is called the flicker effect. The minimum refresh rate to avoid flicker is $50Hz$. Achieving continuous, flicker-free motion would thus require a high refresh rate. However, in both movies and television, there are technical measures that allow lower refresh rates to be used.

9. *Temporal aspect of video bandwidth*

   Temporal specification depends on the rate of the visual system to scan pixels, as well as on the human eye's scanning capabilities. From human visual perspective, the eye requires that a video frame be scanned every 1/25 second. This time is equivalent to the time during which a human eye does not see the flicker effect.

4.1.1.2 Transmission

Video signals are often transmitted to the receiver over a single television channel. In order to encode color, consider the decomposition of a video signal into three subsignals. For reasons of transmission, a video signal is comprised of a luminance signal and two chrominance (color) signals.

Several approaches to color encoding are described below.

- *RGB Signal*

  An RGB signal consists of separate signals for red, green, and blue. Every color can be encoded as a combination of these three primary colors. The values R (for red), G (for green), and B (for blue), are normalized such that white results when $R+G+B=1$ in the normalized representation.

- *YUV Signal*

Since human vision is more sensitive to brightness than to color, a more suitable encoding separates the luminance from the chrominance (color information). Instead of separating colors, the brightness information (luminance Y) is separated from the color information (two chrominance channels U and V).

The YUV signal can be calculated as follows:

$$Y = 0.30R + 0.59G + 0.11B$$
$$U = (B - Y) \times 0.493$$
$$V = (R - Y) \times 0.877$$

- *YIQ signal*

  A similar encoding exists for NTSC's YIQ signal:

$$Y = 0.30R + 0.59G + 0.11B$$
$$I = 0.60R - 0.28G - 0.32B$$
$$Q = 0.21R - 0.52G + 0.31B$$

Digitalization

Before a motion picture can be processed by a computer or transmitted over a network, it must be converted from an analog to a digital representation.

This digitization process consists of the three steps of:

1. *sampling*,
2. *quantization*, and
3. *coding*.

In determining the sampling frequency, the **Nyquist Theorem** must be followed.

Nyquist theorem states that the signal being sampled cannot contain any frequency components that exceed half the sampling frequency.

In order to prevent the base band from overlapping with repeating spectra and to allow for real hardware components not behaving ideally, the sampling rate is normally chosen somewhat higher than the limit dictated by the Nyquist Theorem.

Digitalization consists of sampling the gray (color) level in the picture at $M \times N$ array of points. The next step in the creation of digital motion video is to digitize pictures in time and get a sequence of digital image per second for analog motion video.

Since the gray value of a sampled spot can take on any value in a continuous range, it must be quantized in order to be processed digitally. The gray-scale is subdivided into several ranges, and each pixel is assigned only one of these values.

4.1.2 Computer Video Format

The computer video format depends on the input and output devices for the motion video medium.

Current video digitalization hardware differs with respect to the resolution of the digital images (frames), quantization, and the frame rate (frames/second).

Motion video output depends on the display hardware used, usually a raster display. The typical architecture of such a device is shown in Figure 4.2.

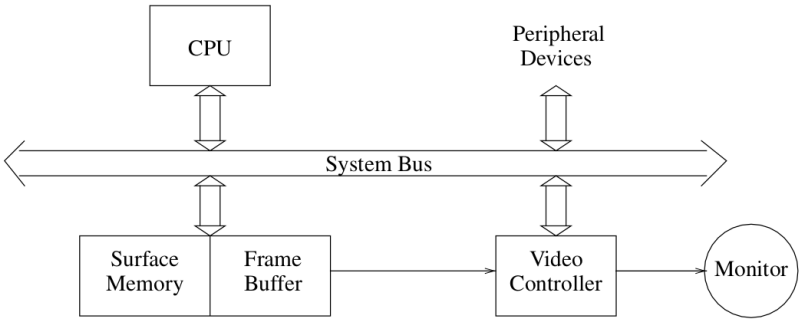

Figure 4.2: Architecture of a raster display.

- The video controller displays the image stored in the frame buffer, accessing the buffer through a separate port as often as required by the video scanning rate.
- The most important task is the constant refresh of the display.
- Due to the disturbing flicker effect, the video controller cycles through the frame buffer, one scan line at a time, typically 60 times/second.
- To display different colors on the screen, the system works with a Color Look-Up Table (CLUT or LUT).
- At any given time, a limited number of colors (n) are available for the whole picture.
- The set of the n most frequently used colors is chosen from a color palette consisting of m colors, whereby in general $n \ll m$.

Some computer video controller standards are given here as examples. Each of these systems supports different resolution and color presentation.

- The *Color Graphics Adapter (CGA)* has a resolution of 320×200pixels with simultaneous presentation of four colors. Therefore, the storage capacity per image is:

$$320 \times 200\, pixels \times \frac{2\, bit/pixel}{8\, bit/byte} = 16,000\, bytes$$

- The *Enhanced Graphics Adapter (EGA)* supports display resolution of 640×350 pixels with 16 simultaneous colors. The necessary storage capacity per frame is:

$$640 \times 350\, pixels \times \frac{4\, bit/pixel}{8\, bit/byte} = 1,12,000\, bytes$$

- The *Video Graphics Array (VGA*) works mostly with a resolution of 640×480 pixels with 256 simultaneous colors. The monitor is controlled via an analog RGB output. The necessary storage capacity per frame is:

$$640 \times 480\, pixels \times \frac{8\, bit/pixel}{8\, bit/byte} = 3,07,200\, bytes$$

- The *Super Video Graphics Array (SVGA*) can present 256 colors at a resolution of $1,024 \times 768$pixels. The necessary storage capacity per frame is:

$$1,024 \times 768\, pixels \times \frac{8\, bit/pixel}{8\, bit/byte} = 7,86,432\, bytes$$

  Other SVGA modes include $1,280 \times 1,024$ pixels and $1,600 \times 1,280$ pixels.

4.2 Animation

To animate something is, literally, to bring it to life. An animation covers all changes that have a visual effect. Visual effects can be very different attributes:

- positions (*motion dynamics*), form, color, transparency, structure, and
- texture of an object (*update dynamics*), as well as
- changes in lighting, camera position, orientation, and focus.

Today, computer-based animations are produced, edited, and generated with the help of a computer using graphical tools to create visual effects. Naturally, the discipline of traditional, non-computer based animation continues to exist.

4.2.1 Basic Concepts of Animation

4.2.1.1 Input Process

- Before the computer can be used, drawings must be digitized to create key frames.
- These digitized images can be produced by the computer using appropriate programs or created by digitizing photos or drawings.
- The drawings may need to be carefully post-processed (e. g. , filtering) in order to clean up any glitches arising from the input process.

4.2.1.2 Composition Stage

- Individual frames in a completed animation are generated by using image composition techniques to combine foreground and background elements.

4.2.1.3 Inbetween Process

- The animation of movement from one position to another requires the composition of frames with intermediate positions (intermediate frames) between key frames.
- In computer-based animation, this in-between processing is done using interpolation methods.
- In interpolation, the system obtains only the beginning and end positions.
- Linear interpolation, sometimes called *lerping*, is the simplest method.

- For example, if one uses lerping to calculate the intermediate positions of a ball that has been thrown in the air and uses only three key frames, as shown in Figure 4.3(a), then the resulting motion of the ball, depicted in Figure 4.3(b), is totally unrealistic.

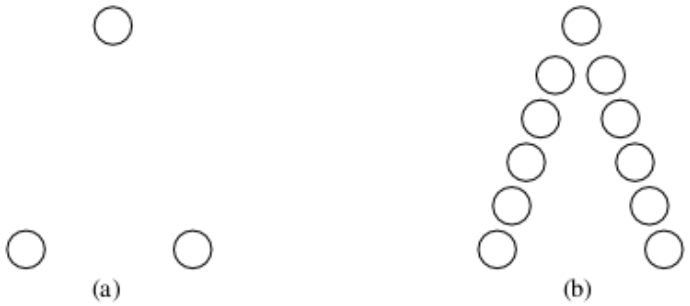

Figure 4.3: Linear interpolation of the motion of a ball: (a) key frames, (b) additional intermediate frames

Due to the disadvantages of lerping, splines are often used to smooth out the interpolation between key frames. Splines can be used to smoothly vary different parameters as a function of time. With splines, individual points (or individual objects) can be moved in a natural fashion through space and time.

4.2.1.4 Changing Colors

To process color changes, computer-based animation uses the Color Look-Up Table (CLUT) or Look-Up Table (LUT).

- The animation is generated by manipulating the LUT.
- The simplest method is to cyclically change the colors of the LUT, thereby changing the colors of different parts of an image.
- Performing LUT animation is relatively fast.

4.2.2 Animation Languages

Various languages exist to describe animations and new formal specification are currently being researched and further developed.

These specifications can be divided into the following three categories:

4.2.2.1 Linear-List Notations

In linear list notation each event in an animation is described by:

- a beginning frame number,
- an end frame number, and
- an action (event) that is to be performed.

Actions typically accept input parameters in the form of an instruction such as the following:

```
42, 53, B, ROTATE "PALM", 1, 30
```

This instruction means that between frames 42 and 53, the object denoted PALM should be rotated 30 degrees around axis 1. A table determines the rotation in each individual frame, allowing for animations with either uniform or accelerated movement.

4.2.2.2 High-Level Programming Language Notations

Another way to describe animations is by embedding animation control in a general-purpose programming language. The values of variables in the language can then be used as parameters for animation routines.

ASAS is an example of such a language based on an extension of LISP. The language's primitives include vectors, colors, polygons, surfaces, groups, points of view, subworlds, and aspects of lighting. ASAS also includes a large collection of geometric transformations that operate on objects.

The following ASAS program fragment describes an animated sequence in which an object called *my-cube* is rotated while the camera pans. This fragment is evaluated for each frame in order to generate the entire sequence.

(**grasp** my-cube); *Make the cube the current object*
(**cw** 0.05); *Small clock-wise rotation*
(**grasp** camera); *Make the camera the current object*
(**right** panning-speed); *Move it to the right*

4.2.2.3 Graphical Languages

A problem with traditional, textual programming languages is that graphical actions cannot be easily visualized by examining scripts.

- Graphical animation languages describe animations in a more visual fashion.

- Such languages are used to name and edit the changes taking place simultaneously in an animation and to visualize the effects created.

- The description of actions to be carried out is done using visual paradigms.

- *GENESYS*, *DIAL* and the *S-Dynamics System* are examples of such systems.

4.2.3 Methods of Controlling Animation

Animation control is independent of the language used to describe animation. There are various techniques for controlling animation.

4.2.3.1 Full Explicit Control

- The simplest type of animation control.

- The animator provides a description of all events that could occur in an animation.

- This can be done by specifying simple transformations-such as *scalings*, *translations*, and *rotations*-or by specifying key frames.

- An example of this type of control is the *BBOP* system.

4.2.3.2 Procedural Control

- Procedural control is based on communication among different objects whereby each object obtains knowledge about the static or dynamic properties of other objects.

- This information can be used to verify that objects move consistently.

- In particular, in systems that represent physical processes, the position of an object can influence the movement of other objects (for example, ensuring that balls cannot move through walls).

- In actor-based systems, individual actors can pass their positions along to others in order to influence their behavior.

4.2.3.3 Constraint-Based Control

Although some objects in the real world move along straight lines, this is not always the case. Many objects' movements are determined by other objects with which they come in contact.

- It is thus much simpler to specify an animation sequence using constraints instead of explicit control.
- Sutherland's *Sketchpad* and Borning's *ThingLab* are examples of systems using constraints for control.

4.2.3.4 Tracking Live Action

By examining the motions of objects in the real world, one can animate the same movement by creating corresponding sequences of objects. Traditional animation uses *rotoscoping*.

- A film is made in which people or animals act out the parts of the performers in the animation.
- Afterwards, animators process the film, enhancing the background and replacing the human actors with the animated equivalents they have created.
- Another such technique is to attach indicators to key points on the body of a human actor.
- The coordinates of the corresponding key points in an animated model can be calculated by observing the position of these indicators.
- An example of this sort of interaction mechanism is the data glove, which measures the position and orientation of the wearer's hand, as well as the flexion and hyperextension of each finger point.

4.2.3.5 Kinematic and Dynamic Control

Kinematic Control

- Kinematics refers to the position and velocity of points.
- A kinematic description of a scene, for example, might say,

> "The cube is at the origin at time $t = 0$. Thereafter, it moves with constant acceleration in the direction $(1\,meter, 1\,meter, 5\,meters)$."

Dynamic Control

In contrast, dynamics takes into account the physical laws that govern kinematics (for example, the Newtonian laws for the movement of large bodies, or the Euler-Lagrange equations for fluids).

- particle moves with an acceleration proportional to the forces acting on it; the proportionality constant is the mass of the particle.
- A dynamic description of a scene might be:

  > "At time $t = 0$, the cube is at position (`0 meter, 100 meter, 0 meter`). The cube has a mass of 100 $grams$. The force of gravity acts on the cube."

- The natural reaction in a dynamic simulation is that the cube would fall.

4.2.4 Display of Animation

- To display animations with raster systems, the animated objects must be scan-converted and stored as a *pixmap* in the frame buffer.
- A rotating object can be shown by displaying successive views from slightly different locations.
- The scan-conversion must be done at least 10 (preferably 15 to 20) times per second in order to give a reasonably smooth visual effect; hence a new image must be generated in at most $100ms$.
- The actual scan-conversion of an object should take only a small portion of this time otherwise distracting ghost effects appears.
- *Double buffering* is used to avoid this (distracting ghost effects) problem.

As an example, consider the display of a rotation animation. Assuming that the two halves of the pixmap are $image_0$ and $image_1$, the process is as

follows:

`Load LUT` to display values as background color
`Scan-convert` object into $image_0$
`Load LUT` to display only $image_0$
Repeat
`Scan-convert` object into $image_1$
`Load LUT` to display only $image_1$
`Rotate` object data structure description
`Scan-convert` object into $image_0$
`Load LUT` to display only $image_0$
`Rotate` object data structure description
Until (termination condition)

If rotating and scan-converting the object takes more than $100ms$, the animation is quite slow, but the transition from one image to the next appears to be instantaneous. Loading the LUT typically takes less than $1ms$.

4.2.5 Transmission of Animation

Animated objects can be represented symbolically using graphical objects or scan-converted pixmap images. Hence, the transmission of an animation may be performed using one of two approaches:

4.2.5.1 Symbolic Representation

The symbolic representation (e.g., circle) of an animation's objects (e. g. , ball) is transmitted together with the operations performed on the object (e. g. , roll the ball). The receiver displays the animation as described earlier (see Section 4.2.4, Page No. 61). Since the byte size of such a representation is much smaller than a pixmap representation, the transmission time is short. However, the display time is longer since the receiver must generate the corresponding pixmaps.

In this approach, the bandwidth (e. g. $bytes/second$) required to transmit an animation depends on:

- the size of the symbolic representation structure used to encode the animated object
- the size of the structure used to encode the operation command, and
- the number of animated objects and operation commands sent per second

4.2.5.2 Pixmap Representation

The pixmap representations of the animated objects are transmitted and displayed by the receiver. The transmission time is longer compared to the previous approach because of the size of the pixmap representation. However, the display time is shorter.

The necessary bandwidth is at least proportional to the size of a single pixmap image and to the image repetition rate. These values are significantly higher than in the case of a symbolic representation.

CHAPTER

5

DATA COMPRESSION

5.1 Data Compression and Coding Fundamentals

5.1.1 Data Compression

Uncompressed graphics, audio, and video data require substantial storage capacity, which is not possible in the case of uncompressed video data. The same is true for multimedia communications. Data transfer of uncompressed video data over digital networks requires that very high bandwidth be provided for a single point-to-point communication. To be cost-effective and feasible, multimedia systems must use compressed video and audio streams.

The most important compression techniques in use today are JPEG for single pictures, H.264 for video, MPEG for video and audio.

5.1.2 Coding/Compression Fundamentals

Images have considerably higher storage requirements than text, and audio and video have still more demanding properties for data storage. Moreover, transmitting continuous media also requires substantial communication data rates. The figures cited below clarify the qualitative transition from simple

text to full-motion video data and demonstrate the need for compression. In order to be able to compare the different data storage and bandwidth requirements of various visual media (text, graphics, images, and video), the following specifications are based on a small window of 640×480 pixels on a display. The following holds always:

$$1\,kbit = 1000\,bit$$
$$1\,Kbit = 1024\,bit$$
$$1\,Mbit = 1024 \times 1024\,bit$$

1. For the representation of the text medium, two bytes are used for every 8×8 pixel character.

$$\text{Character per screen page} = \frac{640 \times 480}{8 \times 8} = 4,800$$

$$\begin{aligned}\text{Storage required per screen page} &= 4,800 \times 2\,\text{byte} \\ &= 9,600\,\text{byte} \\ &= 9.4\,\text{Kbyte}\end{aligned}$$

2. For the representation of vector images, we assume that a typical image consists of 500 lines. Each line is defined by its coordinates in the x direction and the y direction, and by an $8bit$ attribute field. Coordinates in the x direction require 10 bits($\log_2(640)$), while coordinates in the y direction require $9bits$ ($log_2(480)$).

$$\begin{aligned}\text{Bits per line} &= 9\text{bits} + 10\text{bits} + 9\text{bits} + 10\text{bits} + 8\text{bits} \\ &= 46\text{bits}\end{aligned}$$

$$\begin{aligned}\text{Storage required per screen page} &= 500 \times \frac{46}{8} \\ &= 2,875\,\text{byte} \\ &= 2.8\,\text{Kbyte}\end{aligned}$$

3. Individual pixels of a bitmap can be coded using 256 different colors, requiring a single byte per pixel.

$$\begin{aligned}\text{Storage required per screen page} &= 640 \times 480 \times 1\,\text{byte} \\ &= 307,200\,\text{byte} \\ &= 300\,\text{Kbyte}\end{aligned}$$

The next examples specify continuous media and derive the storage required for one second of playback.

1. Uncompressed speech of telephone quality is sampled at $8kHz$ and quantized using $8bit$ per sample, yielding a data stream of $64Kbit/s$.

$$\begin{aligned}\text{Required storage space/s} &= \frac{64\text{Kbit/s}}{8\text{bit/byte}} \times \frac{1\text{ s}}{1,024\text{ byte/Kbyte}} \\ &= 8\text{ Kbyte}\end{aligned}$$

2. An uncompressed stereo audio signal of CD quality is sampled at $44.1kHZ$ and quantized using $16bits$.

$$\begin{aligned}\text{Data rate} &= 2 \times \frac{44,100}{s} \times \frac{16\text{ bit}}{8\text{ bit/byte}} \\ &= 1,76,400\text{ byte/s}\end{aligned}$$

3. A video sequence consists of 25 full frames per second. The luminance and chrominance of each pixel are coded using a total of $3bytes$.

   According to the European PAL standard, each frame consists of 625 lines and has a horizontal resolution of more than 833 pixels. The luminance and color difference signals are encoded separately and transmitted together using a multiplexing technique $(4:2:2)$.

   According to CCIR 601 (studio standard for digital video), the luminance (Y) is sampled at $13.5MHz$, while chrominance $(R-Y$ and $B-Y)$ is sampled using $6.75MHz$. Samples are coded uniformly using $8bits$.

$$\begin{aligned}\text{Bandwidth} &= (13.5\text{MHz} + 6.75\text{MHz} + 6.75\text{MHz}) \times 8\text{bit} \\ &= 216 \times 10^6\text{ bit/s}\end{aligned}$$

$$\begin{aligned}\text{Data rate} &= 640 \times 480 \times 25 \times 3\text{ bytes/s} \\ &= 2,30,40,000\text{ byte/s}\end{aligned}$$

$$\begin{aligned}\text{Required storage space/s} &= 2,304 \times 10^4\text{ bytes} \times \frac{1\text{ s}}{1,024\text{ byte/Kbyte}} \\ &= 22,500\text{ Kbyte}\end{aligned}$$

Classification of Coding/Compression Techniques

Table 5.1: Overview of some coding and compression techniques.

Coding Type	**Basis**	**Technique**
Entropy Coding	Run-length Coding Huffman Coding Arithmetic Coding	
Source Coding	Prediction	DPCM DM
	Transformation	FFT DCT
	Layered Coding (according to importance)	Bit Position Subsampling Subband Coding
	Vector Quantization	

Table 5.1 continued from previous page

Coding Type	Basis	Technique
Hybrid Coding	JPEG MPEG H.264	

5.2 Source, Entropy and Hybrid coding

Compression techniques can be categorized as shown in Table 5.1. We distinguish among three types of coding:

1. *entropy*,
2. *source*, and
3. *hybrid* coding.

Entropy coding is a lossless process, while source coding is often lossy. Most multimedia systems use hybrid techniques; most are only combinations of entropy and source coding, without any new processing algorithms.

5.2.1 Entropy Coding

- Entropy coding is used regardless of the media's specific characteristics.
- The data stream to be compressed is considered to be a simple digital sequence and the semantics of the data is ignored.
- Entropy encoding is an example of *lossless* encoding as the decompression process regenerates the data completely.
- *Run-length coding* is an example of entropy encoding that is used for data compression in file systems.

5.2.2 Source Coding

- Source coding takes into account the semantics of the information to be encoded.
- The degree of compression attainable with this often lossy technique depends on the medium.
- In the case of lossy compression, a relation exists between the uncoded data and the decoded data; the data streams are similar but not identical.
- In the case of speech, a considerable reduction of the amount of data can be achieved by transforming the time-dependent signal into the frequency domain, followed by an encoding of the formants[1].

[1] *Formants* are defined as the maxima in the voice spectrum.

5.2.3 Hybrid Coding

To Understand the hybrid schemes, we consider a set of typical processing steps common to all techniques (entropy, source, and hybrid).

Major Steps of Data Compression

Figure 5.1 shows the typical sequence of operations performed in the compression of still images and video and audio data streams. The following example describes the compression of one image:

1. The *preparation* step (here picture preparation) generates an appropriate digital representation of the information in the medium being compressed. For example, a picture might be divided into blocks of $8 \times 8\, pixels$ with a fixed number of bits per pixel.

2. The *processing* step (here picture processing) is the first step that makes use of the various compression algorithms. For example, a transformation from the time domain to the frequency domain can be performed using the Discrete Cosine Transform (DCT).

3. *Quantization* takes place after the mathematically exact picture processing step.

4. *Entropy coding* starts with a sequential data stream of individual bits and bytes.

Figure 5.1 shows the compression process applied to a still image; the same principles can also be applied to video and audio data.

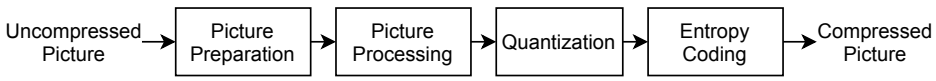

Figure 5.1: Major steps of data compression.

Decompression is the inverse process of compression. Specific coders and decoders can be implemented very differently. Symmetric coding is characterized by comparable costs for encoding and decoding, which is especially desirable for dialogue applications. In an asymmetric technique, the decoding process is considerably less costly than the coding process.

This is intended for applications where:

- compression is performed once and

- decompression takes place very frequently, or if the decompression must take place very quickly

For example, an audio-visual course module is produced once, but subsequently decoded by the many students who use it. The main requirement is real-time decompression. An asymmetric technique can be used to increase the quality of the compressed images.

5.3 Basic Data Compression Techniques

Compression basically employs redundancy in the data:

- *Temporal* in 1D data, 1D signals, audio, between video frames etc.
- *Spatial* correlation between neighboring pixels or data items.
- *Spectral* e. g. correlation between color or luminescence components. This uses the frequency domain to exploit relationships between frequency of change in data.
- *Psycho-visual* exploit perceptual properties of the human visual system.

Compression methods can also be categorized in two broad ways:

1. *Lossless compression*:
   - after decompression gives an exact copy of the original data.
   - *Example*: Entropy encoding schemes (Shannon-Fano, Huffman coding), arithmetic coding, LZ/LZW algorithm (used in GIF image file format).
2. *Lossy compression*:
   - *Example*: Transform coding — FFT/DCT based quantization used in JPEG/MPEG differential encoding, vector quantization.
   - Lossy methods are typically applied to high resolution audio, image compression.
   - Have to be employed in video compression (apart from special cases).

5.3.1 Run-Length Coding

- Data often contains sequences of identical bytes.
- By replacing these repeated byte sequences with the number of occurrences, a substantial reduction of data can be achieved.
- This is known as *run-length coding*.

A special marker M is needed in the data that does not occur as part of the data stream itself. To illustrate this, we define the exclamation mark '!' to be the 'M-byte'. A single occurrence of an exclamation mark is interpreted as the 'M-byte' during decompression. Two consecutive exclamation marks are interpreted as an exclamation mark occurring within the data.

The 'M-byte' can thus be used to mark the beginning of a run-length coding. In the following example, the character `C` occurs eight times in a row and is compressed to the three characters `C!8` :

Uncompressed data: `ABCCCCCCCCDEFGGG`

Run-length coded: `ABC!8DEFGGG`

5.3.2 Zero Suppression

- Run-length coding is a generalization of **zero suppression**, which assumes that just one symbol appears particularly often in sequences.
- The blank (space) character in text is such a symbol; single blanks or pairs of blanks are ignored.

5.3.3 Pattern Substitution / Diatomic Encoding

- A technique that can be used for text compression substitutes single bytes for patterns that occur frequently.

5.3.4 Huffman Coding

- Huffman coding algorithm determines the optimal coding using the minimum number of bits.
- Hence, the length (number of bits) of the coded characters will differ.
- The most frequently occurring characters are assigned to the shortest code words.

- A Huffman code can be determined by successively constructing a binary tree, whereby the leaves represent the characters that are to be encoded.
- Every node contains the relative probability of occurrence of the characters belonging to the subtree beneath the node.
- The edges are labeled with the bits 0 and 1.

Algorithm

1. Initialization: put all nodes in a list L, keep it sorted at all times (e. g. , ABCDE).
2. Repeat until the list L has more than one node left.
   (a) From L pick two nodes having the lowest frequencies/probabilities, create a parent node of them.
   (b) Assign the sum of the children's frequencies/probabilities to the parent node and insert it into L.
   (c) Assign code 0/1 to the two branches of the tree, and delete the children from L
3. Assign a codeword for each leaf based on the path from the root.

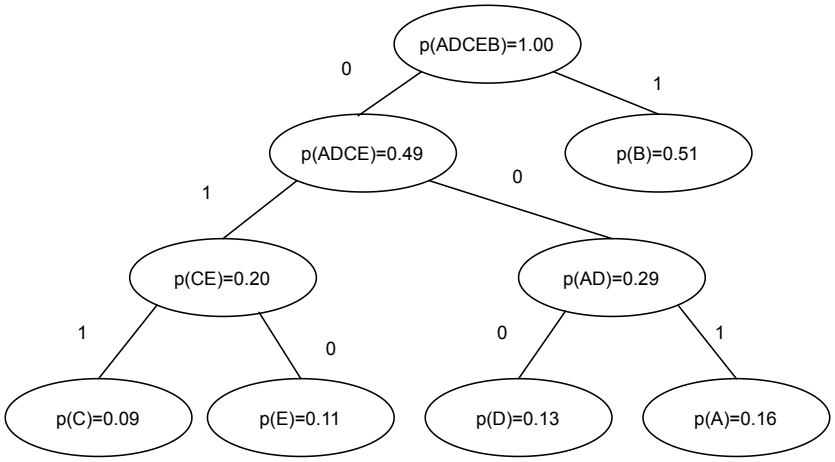

Figure 5.2: Example of a Huffman code represented as a binary tree

The following brief example illustrates this process:

1. The letters A, B, C, D, and E are to be encoded and have relative probabilities of occurrence as follows:

$$p(A) = 0.16, p(B) = 0.51, p(C) = 0.09, p(D) = 0.13, p(E) = 0.11$$

2. The two characters with the lowest probabilities, C and E, are combined in the first binary tree, which gives the probability of 0.20 to their root node CE.

3. Nodes with the following relative probabilities remain:

$$p(A) = 0.16, p(B) = 0.51, p(CE) = 0.20, p(D) = 0.13$$

   The two nodes with the lowest probabilities are D and A. These nodes are combined to form the leaves of a new binary tree. The combined probability of the root node AD is 0.29.

4. Nodes with the following relative probabilities remain:

$$p(AD) = 0.29, p(B) = 0.51, p(CE) = 0.20$$

   The two nodes with the lowest probabilities are AD and CE. These are combined into a binary tree. The combined probability of their root node $ADCE$ is 0.49.

5. Two nodes remain with the following relative probabilities:

$$p(ADCE) = 0.49, p(B) = 0.51$$

   These are combined to a final binary tree with the root node $ADCEB$.

6. Figure 5.2 shows the resulting Huffman code as a binary tree. The result is the following code words, which are stored in a table:

$$w(A) = 001, w(B) = 1, w(C) = 011, w(D) = 000, w(E) = 010$$

Such a table could be generated for a single image or for multiple images together. The same table must be available for both encoding and decoding.

5.3.5 Arithmetic Coding

- Unlike Huffman coding, arithmetic coding does not code each symbol separately.
- Each symbol is instead coded by considering all prior data.

5.3.6 Other Basic Techniques

Apart from the compression techniques described earlier, some additional well known techniques are used today:

- Video compression techniques often use *Color Look-Up Tables (CLUT)*.
- A simple technique for audio is silence suppression, whereby data is only encoded if the volume exceeds a certain threshold.

5.4 Coding Standard JPEG, MPEG and DVI

5.4.1 JPEG

Joint Photographic Experts Group (JPEG) is an adaptive transformation coding techniques based on the Discrete Cosine Transform (DCT). In 1992, JPEG became an ISO International Standard. JPEG applies to color and gray-scaled still images.

Figure 5.3 outlines the fundamental steps of JPEG compression in accordance with the general scheme illustrated in Figure 5.1. JPEG defines several image compression modes by selecting different combinations of these steps.

JPEG Modes

JPEG defines four modes, which themselves include additional variations:

- The *lossy*, sequential DCT-based mode (baseline process, base mode) must be supported by every JPEG decoder.
- The *expanded lossy*, DCT-based mode provides a set of further enhancements to the base mode.
- The *lossles*s mode has a low compression ratio and allows a perfect reconstruction of the original image.
- The *hierarchical* mode accommodates images of different resolutions by using algorithms defined for the other three modes.

Lossy Sequential DCT-Based Mode

After image preparation, the uncompressed image samples are grouped into data units of 8×8 pixels, as shown in Figure 5.4; the order of these data units is defined by the MCUs. In this baseline mode, each sample is encoded using $p = 8bit$. Each pixel is an integer in the range 0 to 255.

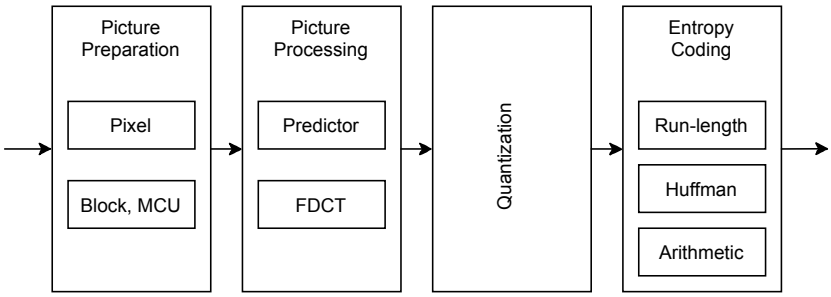

Figure 5.3: Steps of the JPEG compression technique: summary of the different modes.

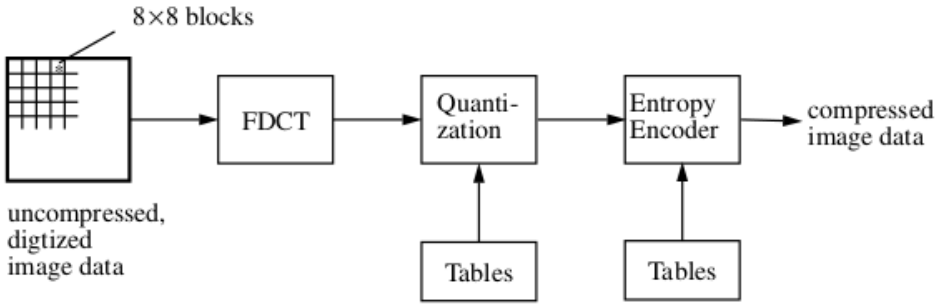

Figure 5.4: Steps of the lossy sequential DCT-based coding mode, starting with an uncompressed image after image preparation.

Steps of JPEG Image Compression

Step 1: *Splitting*: The input image is divided into a small block which is having 8×8 dimensions. This dimension is sum up to 64 units. Each unit of the image is called *pixel*.

Step 2: *Color Space Transform*: JPEG uses $[Y, Cb, Cr]$ model instead of using the $[R, G, B]$ model. RGB is converted into $YCbCr$.

- Y is for brightness,
- Cb is color blueness, and
- Cr for color redness.

Step 3: *Apply DCT*: After the conversion of colors, it is forwarded to DCT. DCT uses a cosine function and does not use complex numbers. It converts information which are in a block of pixels from the spatial domain to the frequency domain.

Step 4: *Quantization* is used to reduce the number of bits per sample.

Step 5: *Serialization*: The zigzag scan is used to map the 8×8 matrix to a 1×64 vector. Zigzag scanning is used to group low-frequency coefficients to the top level of the vector and the high coefficient to the bottom.

Step 6: *Vectoring*: The different pulse code modulation (DPCM) is applied to the DC component. DPCM encodes the difference between the current block and the previous block.

Step 7: *Encoding*: Run Length Encoding (RLE) is applied to AC components. This is done because AC components have a lot of zeros in it. It encodes in a pair of `(skip, value)` in which `skip` is non-zero value and `value` is the actual coded value of the non-zero components.

5.4.2 MPEG

MPEG was developed and defined by ISO/IEC JTC1/SC 29/WG 11 to cover motion video as well as audio coding.

5.4.2.1 Video Encoding

MPEG supports four types of image coding. In order to achieve a high compression ratio, temporal redundancies of successive images need to be exploited. Fast random access requires that images be coded individually. Hence the following image types are distinguished:

5.4.2.1.1 I-Frames (Intra-coded frames)

- Are coded without using information about other frames.
- An *I*-frame is treated as a still image. Here, MPEG falls back on the results of JPEG.
- Unlike JPEG, real-time compression must be possible.
- The compression rate is thus the lowest within MPEG.
- *I*-frames form the anchors for random access.
- *I*-frames are encoded by performing a DCT on the 8×8 blocks defined within the macro blocks.

5.4.2.1.2 P-Frames (Predictive-coded frames)

- Require information about previous I and/or P frames for encoding and decoding.
- Decoding a P-frame requires decompression of the last I frame and any intervening P-frames.
- In return, the compression ratio is considerably higher than for I frames.
- A P-frame allows the following P-frame to be accessed if there are no intervening I frames.

5.4.2.1.3 B-frames (Bidirectionally predictive-coded frames

- Require information from previous and following I and/or P frames.
- B-frames yield the highest compression ratio attainable in MPEG.
- A B-frame is defined as the difference from a prediction based on a previous and a following I or P-frame.
- It cannot serve as a reference for prediction coding of other frames.

5.4.2.1.4 D-frames (DC coded frames

- Are intraframe-coded and can be used for efficient fast forward.
- During the DCT, only the DC-coefficients are coded; the AC coefficients are ignored.

Figure 5.5 shows a sequence of I, P, and B-frames. This example illustrates the prediction for the first P-frame and the bidirectional prediction for a B-frame. The order in which the images are presented differs from the actual decoding order if B-frames are present in an MPEG-coded video stream.

The pattern of I, P, and B-frames in a sequence is determined by the MPEG application. For random access, the ultimate resolution would be attained by encoding the entire stream using I-frames. The highest compression rate can be achieved by using as many B-frames as possible. For practical applications, the sequence `IBBPBBPBBIBBPBBPBB...` has proven to be useful. This permits random access with a resolution of nine still images (i. e. , about $330ms$) and still provides a very good compression ratio. Every 15 images includes one I-frame.

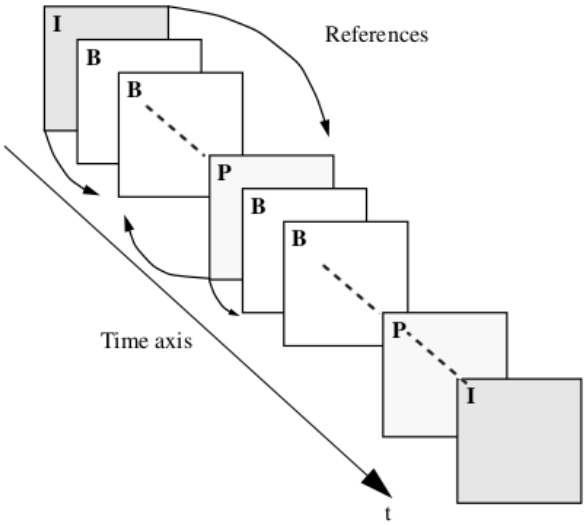

Figure 5.5: Types of individual images in MPEG: I, B, and P frames.

5.4.2.2 Audio Coding

MPEG audio coding uses the same sampling frequencies as Compact Disc Digital Audio (CD-DA) and Digital Audio Tape (DAT), i. e. , $44.1kHz$ and $48kHz$, and additionally, $32kHz$ is available, all at $16bits$.

Three quality levels (layers) are defined with different encoding and decoding complexity. An implementation of a higher layer must be able to decode the MPEG audio signals of lower layers.

Similar to two-dimensional DCT for video, a transformation into the frequency domain is applied for audio. The Fast Fourier Transform (FFT) is a suitable technique. As shown in Figure 5.6:

- The relevant portion of the spectrum is divided into 32 non-overlapping subbands.
- The audio signal is thus split into 32 subbands.
- Different components of the spectrum can then be quantized differently.
- In parallel with the actual FFT,the noise level in each subband is determined using a psychoacoustic model.
- At a higher noise level, a coarser quantization is performed.
- A lower noise level results in finer quantization.

- In the first and second layers, the appropriately quantized spectral components are simply PCM-encoded.
- The third layer additionally performs Huffman coding.

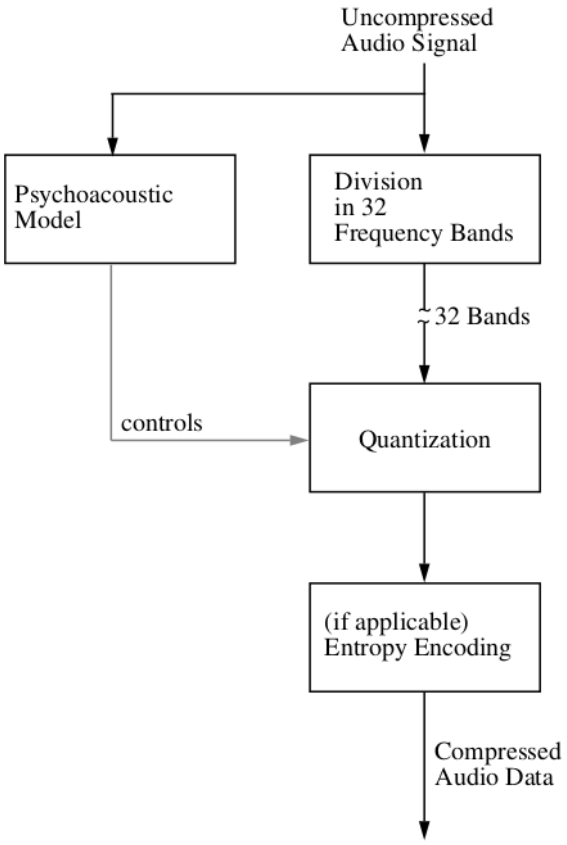

Figure 5.6: MPEG audi encoding.

Audio coding can be performed on stereo sound, a single channel, or two independent channels. MPEG provides for two types of stereo sound.

- In the first case, two channels are processed completely independently.
- In the joint stereo mode, MPEG achieves a higher compression ratio by exploiting redundancies between the two channels.

5.4.3 DVI

Digital Video Interactive(DVI) is a technology that includes coding algorithms. The fundamental components are a VLSI chip set for the video

subsystem, a well-specified data format for audio and video files, an application user interface to the audio-visual kernel (AVK, the kernel software interface to the DVI hardware) and compression, as well as decompressing, algorithm.

DVI can process *data, text, graphics, still images, video* and *audio*. The original essential characteristic was the asymmetric technique of video compression and decompression known as Presentation-Level Video (PLV).

5.4.3.1 Audio and Still Image Encoding

Audio signals are digitized using $16bits$ per sample and are either PCM-encoded or compressed using the Adaptive Differential Pulse Code Modulation (ADPCM) technique. Thereby, a reduction to about four bits per sample is achieved at a quality corresponding to stereo broadcasting.

When encoding still images, different video input formats can be used. These can be composite, as well as component video signals like, RGB. In the case of an RGB signal, the color of each pixel is split into portions of the three colors of the spectrum-red, green and blue - and each color is processed separately.

5.4.3.2 Video Encoding

For motion video encoding DVI distinguishes two techniques with different resolutions and dissimilar goals:

- *Presentation-Level Video (PLV)* is characterized by its better quality. This is achieved at the expense of a very time-consuming asymmetric compression performed by specialized compression facilities.
- *Real-Time Video (RTV)* is a symmetric compression technique that works with hardware and software and can be performed in real-time.

Where,

1. *Asymmetric coding* requires considerably more effort for encoding than for decoding. Compression is carried out once, whereas decompression is performed many times. A typical application area is retrieval systems.
2. *Symmetric compression* is characterized by a comparable effort for the compression and decompression processing.

CHAPTER

6

OPTICAL STORAGE MEDIA

Conventional magnetic data carriers in the form of hard disks or removable disks are traditionally used in computers as secondary storage media. These offer low average access time and provide enough capacity for general computer data at an acceptable price. However, audio and video data, even in compressed form, place heavy demands on available storage capacity. The storage cost for continuous media is thus substantial unless other data carriers are used.

Optical storage media offer higher storage density at a lower cost. The *audio compact disc* (CD) has been commercially successful in the consumer electronics industry as the successor to *long-playing records* (LPs) and is now a mass-produced product.

6.1 Basic Technology

- In optical storage media, information is represented by using the intensity of laser light reflected during reading.

- A laser beam having a wave length of about $780nm$ can be focused to a resolution of approximately $1\mu m$.

- In a polycarbonate substrate layer, there are depressions, called *pits*, corresponding to the data to be encoded.
- The areas between the pits are called *lands*.

Figure 6.1 shows a sectional view through an optical disc, running lengthwise along a data track. The pits and lands are represented schematically in the middle of the figure.

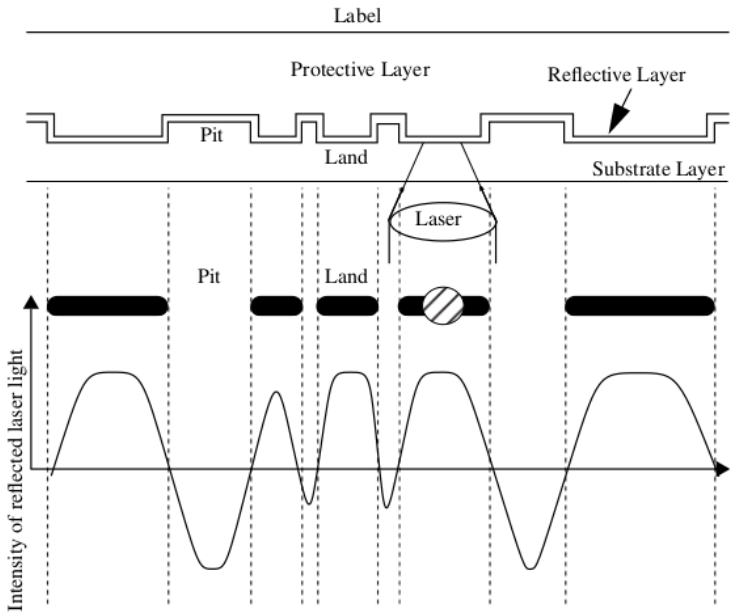

Figure 6.1: Sectional view of an optical disc along the data track. Schematic representation of the layers (top), the "lands" and the "pits" (middle), and the signal waveform (bottom).

- The substrate layer is smooth and coated with a thin, reflective layer.
- The laser beam is focused at the height of the reflective layer from the substrate level.
- The reflected beam thus has a strong intensity at the lands.
- The pits have a depth of $0.12\mu m$ from the substrate surface.
- Laser light hitting pits will be lightly scattered, that is, it will be reflected with weaker intensity.

- The signal waveform shown schematically at the bottom of Figure 6.1 represents the intensity of the reflected laser light; the horizontal line represents a threshold value.
- The laser in the figure is currently sampling a land.

According to Figure 6.1, a Compact Disc (CD) consists of:

- the label,
- the protective layer,
- the reflective layer and
- the substrate.

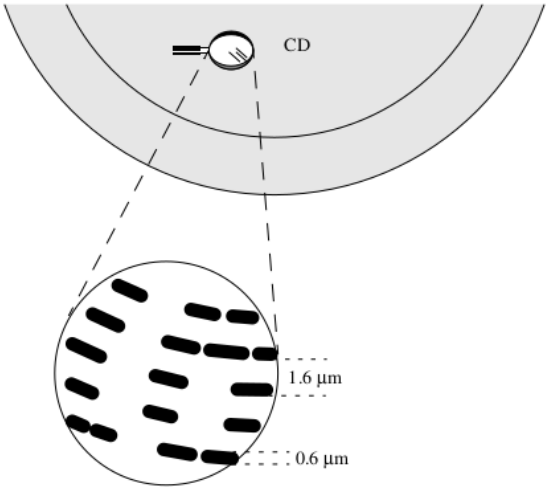

Figure 6.2: Data on a CD as an example of an optical disc. Track with "lands" and "pits".

- An optical disc consists of a sequential arrangement of pits and lands within a track.
- The pits and lands represent data on the surface (see Figure 6.2).
- All the information on an optical disc is placed on one track.
- The stored information can thus be played back at a continuous data rate
- A track is in the form of a spiral.
- In the case of a CD, the spacing between adjacent coils of the spiral—the track pitch is $1.6\mu m$.
- The track width of the pits is $0.6\mu m$, though their lengths can vary.

The light source of the laser can be positioned at a distance of approximately $1mm$ from the disk surface and thus does not touch the disk directly, or float on an air cushion. This reduces wear and tear on the components used and increases the life of the device.

6.2 Video Disc Fundamentals

- Video discs in the form of *LaserVision* are used for the reproduction of motion picture and audio data.
- The data are stored on the disc in an analog-coded format, and
- the sound and picture quality are excellent.
- *LaserVision* discs have a diameter of approximately $30cm$ and store approximately $2.6Gbytes$.

Motion pictures are frequency-modulated on the video disc, and the audio signal is mixed with the modulated video signal. Figure 6.3 shows the principle used to record data.

- The important information of the mixed audio-video signal is the temporal sequence of the zero transitions.
- Each zero transition corresponds to a change between a pit and a land on the disc.
- Such a change can occur at any time, and is written to the disc in a non-quantized form, that is, the pit length is not quantized.
- This method is thus time-continuous and can be characterized as analog.

The video disc was conceived as a Read Only Memory. Since then, many different write-once optical storage systems have come out, known as Write Once Read Many (WORM). An example is the Interactive Video Disc, which operates at Constant Angular Velocity (CAV). On each side, up to 36 minutes of audio and video data at 30 frames per second can be stored and played back. Alternatively, approximately 54,000 studio quality still images can be stored per side.

WORMs have the following special properties:

- The term *Media overflow* refers to problems that can occur when a WORM disc is almost full.

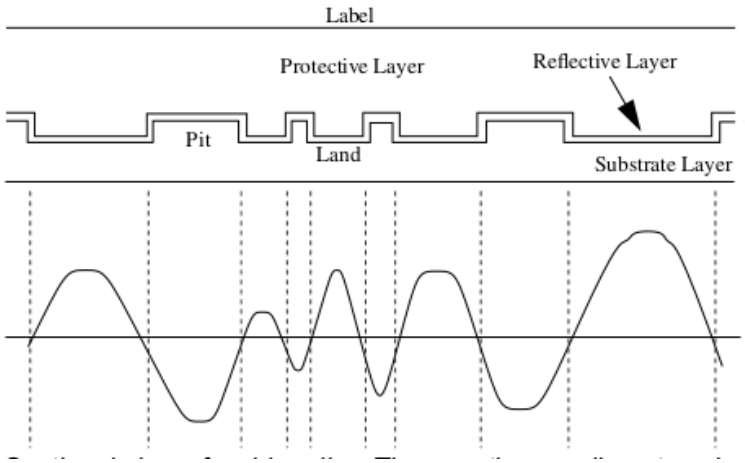

Figure 6.3: Sectional view of a video disc. Time-continuous discrete value coding.

- *Packaging* refers to problems stemming from the fixed block structure of WORMs.
- *Revision* refers to the problem of subsequently marking areas as invalid.

6.3 CD Audio, CD ROM and Extended Architecture

6.3.1 Compact Disc Digital Audio

The Compact Disc Digital Audio (CD-DA) was developed jointly by N. V. Philips and the Sony Corporation *for storing audio data*. The basic technology of the CD-DA was developed by N. V. Philips.

6.3.1.1 Technical Basics

- CDs have a diameter of $12cm$ and are played at a *Constant Linear Velocity (CLV)*.
- The number of rotations per time unit thus depends on the radius of the data currently being sampled.
- The spiral-shaped CD track has approximately 20,000 windings.

Information is stored according to the principle depicted in Figure 6.1 and Figure 6.4, whereby the length of the pits is always a multiple of $0.3\mu m$. A change from pit to land or from land to pit corresponds to the coding of a 1 in the data stream. If there is no change, a 0 is coded.

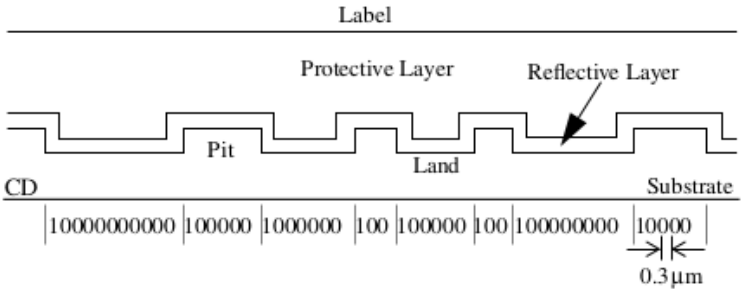

Figure 6.4: "Pits" and "lands". Discrete time, discrete value storage.

6.3.1.2 Audio Data Rate

The audio data rate can easily be derived from the given sampling rate of $44,100Hz$ and the $16bit$ linear quantization. The stereo audio signal is pulse code modulated, and the data rate is as follows:

$$
\begin{aligned}
&\text{Audio data rate}_{CD-DA} \\
&= 16\frac{bits}{sample\ value} \times 2\ channels \times 44,100\frac{sample\ values}{s \times channel} \\
&= 14,11,200 \times \frac{bits}{s} \\
&= 14,11,200 \times \frac{bits/s}{8bits/byte} \\
&= 14,11,200 \times \frac{\cancel{bits}}{s} \times \frac{byte}{8\cancel{bits}} \\
&= 14,11,200 \times \frac{byte}{8s} \\
&= 1,76,400 \times \frac{byte}{s} \\
&= 176.4\frac{Kbytes}{s}\ (१,\ ७६,\ ४००\ \text{लाई}\ १,०००\ \text{ले भाग गरेको}) \\
&\cong 172.3\frac{Kbytes}{s}\ (१,\ ७६,\ ४००\ \text{लाई}\ १,०२४\ \text{ले भाग गरेको})
\end{aligned}
$$

Where, $\cong$ *means "is congruent to (that is the equality up to a displacement)".*

Analog long-playing records and casette tapes have a signal-to-noise ratio of approximately 50 to $60dB$. The quality of the CD-DA is substantially higher. As a rule of thumb, one can assume $6dB$ per bit used during the sampling process. Given $16bit$ linear sampling:

$$S/N_{CD-DA} \cong 6\frac{dB}{bit} \times 16bits = 96dB$$

The signal-to-noise ratio is exactly $98dB$.

Capacity

The play time of a CD-DA is at least *74 minutes*. Using this value, the capacity of a CD-DA can easily be determined. The capacity given below applies only to storage used for audio data, without, for example, taking into account data used for error correction:

$$\begin{aligned}
\text{Capacity}_{CD-DA} &= 74min \times 14,11,200\frac{bits}{s} \\
&= 74 \times 60 \times 14,11,200\frac{bits}{s} \\
&= 6,26,57,28,000\frac{bits}{s} \\
&= 6,26,57,28,000bits \times \frac{1}{8\frac{bits}{byte}} \times \frac{1}{1,024\frac{bytes}{Kbyte}} \times \frac{1}{1,024\frac{Kbytes}{Mbyte}} \\
&\cong 747Mbytes
\end{aligned}$$

6.3.2 CD-ROM

The Compact Disc Read Only Memory (CD-ROM) was conceived as a storage medium for general computer data, in addition to uncompressed audio data. Further, CD-ROM technology was intended to form the basis for the storage of other media. It was specified by N. V. Philips and the Sony Corporation in the *Yellow Book* and later accepted as an ECMA standard.

CD-ROM tracks are divided into *audio* (corresponding to CD-DA) and *data* types. Each track may contain exclusively data of one type. A CD-ROM can contain both types of tracks. In a mixed mode disc, the data tracks are usually located at the beginning of the CD-ROM, followed by the audio tracks.

6.3.2.1 Blocks

Since CD-ROMs store general computer data, they require better error correction and higher resolution random access to data units than are specified for CD-DA. A CD-DA has an error rate of 10^{-8} and allows random access to individual tracks and index points.

This data unit is called a *block*, meaning the *physical block*. In the ISO 9660 standard, there also exists the notion of a *logical block*. The logical block has similar properties to the sectors of other media and file system.

A CD-ROM block consists of the $2,352 bytes$ of a CD-DA block. Thus the $defacto$ CD-DA standard can serve as the basis for the de facto CD-ROM standard.

Of the $2,352 bytes$ of a block, $2,048 bytes$ (computer data) or $2,336 bytes$ (audio data) are available for user data. The remaining bytes are used for identification for random access and for another error correction layer that further reduces the error rate.

Figure 6.5 shows the data hierarchy of a CD-ROM or CD-DA.

$75 blocks$ per second are played back, each consisting of $32 frames$. Each frame is $73.5 bytes$ ($588 bits$).

$$\begin{aligned} \text{Blocks} &= 14,11,200 \frac{bits}{s} \times \frac{1}{75} s \times \frac{1}{8 bits/byte} \\ &= 2,352 bytes \end{aligned}$$

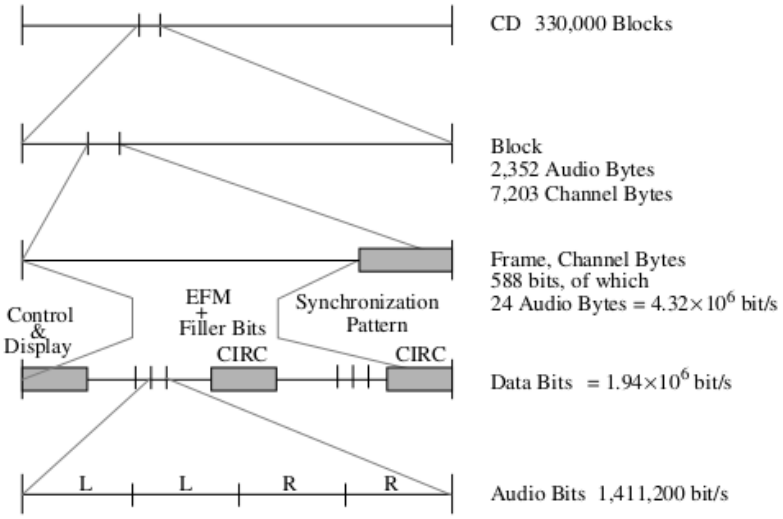

Figure 6.5: CD-ROM data hierarchy. Audio blocks as on a CD-DA

6.3.2.2 Modes

The CD-ROM specification was defined with the goal of storing uncompressed CD-DA data and computer data, as well as serving as the basis for other media. This is achieved by using two CD-ROM modes:

- *mode 1* and
- *mode 2*

An additional mode 0, where all $2,336$ user data bytes are set to zero, serves to separate storage areas.

CD-ROM Mode 1

- CD-ROM mode 1 is used to store computer data, as shown in Figure 6.6.
- Of the $2,352$ total bytes in each block, $2,048 bytes$ are available for storing information.

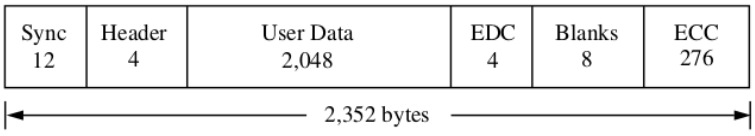

Figure 6.6: CD-ROM mode 1 block (sector) layout.

To be more exact, the $2,352 bytes$ can be broken down into the following groups:

- $12 bytes$ for synchronization as the start-of-block indicator,
- $4 bytes$ for the header. This contains an unambiguous block identifier.
  - The first two bytes contain minutes and seconds, respectively;
  - the third byte contains the block number,
  - while the fourth byte identifies the mode.
- $2,048 bytes$ of user data,
- $4 bytes$ for error detection,
- 8 unused bytes, and

- $276 bytes$ for error correction, whereby an error rate of 10^{-12} can be achieved.

A CD-ROM contains $3,33,000 blocks$ to be played in$74 minutes$. The capacity of a CD-ROM with all blocks in *mode 1* can be calculated as follows:

$$\begin{aligned} Capacity_{CD-ROM_{Mode\,1}} &= 3,33,000 blocks \times 2,048 \frac{bytes}{block} \\ &= 68,19,84,000 bytes \\ &= 68,19,84,000 bytes \times \frac{1}{1,024 \frac{bytes}{Kbyte}} \times \frac{1}{1,024 \frac{Kbytes}{Mbyte}} \\ &\cong 650 Mbytes \end{aligned}$$

The data rate in *mode 1* is:

$$\begin{aligned} Rate_{CD-ROM_{Mode\,1}} &= 2,048 \frac{bytes}{block} \times 75 \frac{blocks}{s} \\ &= 153.6 \frac{Kbytes}{s} \\ &\cong 150 \frac{Kbytes}{s} \end{aligned}$$

CD-ROM Mode 2

- CD-ROM mode 2 serves as the basis for additional specifications for storage of other media.
- A block in this mode is shown in Figure 6.7.
- Of the $2,352$ total bytes in each block, $2,336 bytes$ are available for storing information.

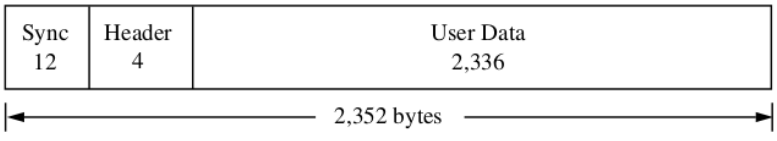

Figure 6.7: CD-ROM mode 2 block (sector) layout.

The synchronization and header are dealt with as in mode 1. The additional error correction is left out.

The capacity of a CD-ROM with all blocks in mode 2 can be calculated as follows:

$$\begin{aligned} Capacity_{CD-ROM_{Mode\,2}} &= 3,33,000 blocks \times 2,336 \frac{bytes}{block} \\ &= 77,78,88,000 bytes \\ &= 741.8518 Mbytes \end{aligned}$$

The data rate in *mode 2* is:

$$\begin{aligned} Rate_{CD-ROM_{Mode\,2}} &= 2,336 \frac{bytes}{block} \times 75 \frac{blocks}{s} \\ &= 175.2 \frac{Kbytes}{s} \end{aligned}$$

6.3.3 CD ROM Extended Architecture

- The *Compact Disc Read Only Memory Extended Architecture* (CD-ROM/XA), is based on the CD-ROM specification.
- Was established by N. V. Philips, the Sony Corporation, and Microsoft.
- The main motivation was to address the inadequate consideration paid until then to concurrent output of multiple media.

The *Red Book* specifies a track for uncompressed audio data. The *Yellow Book* specifies tracks for computer data using CD-ROM *mode 1* (Figure 6.6) and tracks for compressed media using CD-ROM *mode 2* (Figure 6.7).

- CD-ROM/XA uses CD-ROM *mode 2* in order to define its own blocks and additionally defines a sub-header that describes each block (sector) as shown in Figure 6.8 and Figure 6.9.
- This makes it possible to interleave different media using only *mode 2* blocks, since these can contain different media.
- The individual CD-ROM/XA data streams are separated during playback.

Form 1 and Form 2

CD-ROM/XA differentiates blocks with form 1 and form 2 formats, similar to the CD-ROM modes:

6.3.3.1 Form 1

- The XA format form 1 in CD-ROM mode 2 provides improved error detection and correction.
- Like CD-ROM mode 1, 4 bytes are needed for error detection and $276 bytes$ for error correction.
- Unlike CD-ROM mode 1, the 8 bytes unused in CD-ROM mode 1 are used for the subheader.
- Figure 6.8 shows a block (sector), where $2,048 bytes$ are used for data.

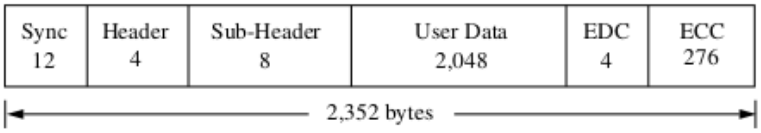

Figure 6.8: Sector layout (1) for CD-ROM/XA according to the "Green Book". Data layout of a CD-ROM block in mode 2, form 1

6.3.3.2 Form 2

- The XA format form 2 in CD-ROM mode 2 allows a 13 percent increase in actual data capacity, to $2,324 bytes$ per block, at the expense of error handling.
- Form 2 blocks can be used to store compressed data of various media, including audio and video data.

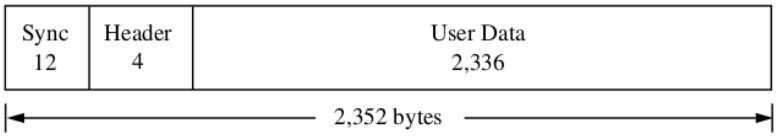

Figure 6.9: Sector layout (2) for CD-ROM/XA according to the "Green Book". Data layout of a CD-ROM block in mode 2, form 2.

On a CD-DA, CD-ROM, or mixed mode disc, a track always consists of homogeneous data, meaning exclusively audio or computer data. It is thus not possible for the computer to, for example, concurrently read uncompressed audio data and computer data.

Advantage of CD-ROM/XA

- Blocks of different media can be stored in one track since they are all coded in CD-ROM mode 2.

6.4 Principles of CD-WO and CD-MO

CD Write-Once

So far, all the CD technologies considered do not allow the user to write to the disk. Thus, the application scope is limited. This has led research laboratories to develop, besides the *Read Only Storage Media*, compact disks that can be recorded once or several times.

The *Compact Disk Write Once* (CD-WO), like WORM (Write Once Read Many), allows the user to write once to a CD and afterwards to read it many times. CD-WO is specified in the second part of the *Orange Book*.

6.4.1 Principle of CD-WO

Figure 6.10 shows a cross-section of a CD-WO, vertical to the disk surface and data track. In the case of read-only CDs, the substrate (a polycarbonate) lies directly next to the reflection layer.

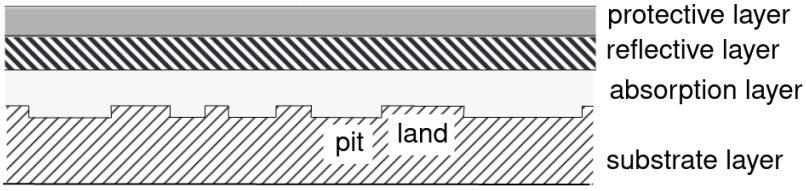

Figure 6.10: Cross-section of a CD-WO disc.

- In the case of a CD-WO, an *absorption layer* exists between the substrate and the reflection layer.
- This layer can be irreversibly modified through strong thermal influence, which changes the reflection properties of the laser beams.
- In its original state, a CD-WO player recognizes a track consisting of lands.

- The absorption layer in the pre-grooved track is heated to above $250°C$ with a laser three to four times the intensity of a reading player.
- Hence, the absorption layer changes such that the reflection of the laser light now corresponds to a pit.
- This method determines the most remarkable property of the CD-WO: its data can be played by any devices which are meant only for read-only CDs.

CD Magneto Optical

The Compact Disc Magneto Optical (CD-MO), specified in the first part of the *Orange Book*, has a high storage capacity and allows the CD to be written multiple times.

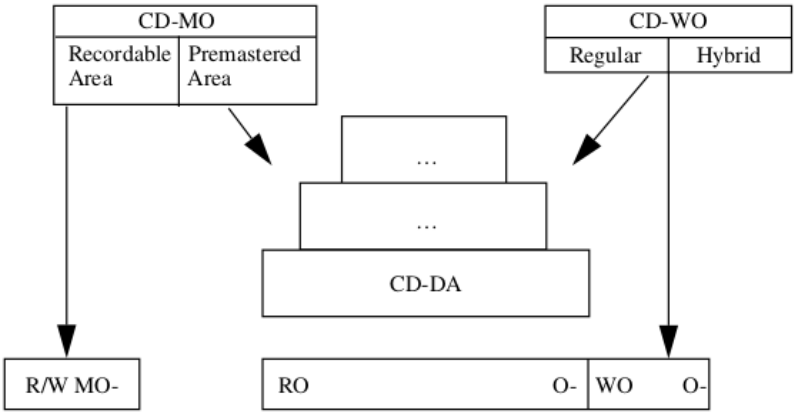

Figure 6.11: CD-WO and CD-MO in relation to other CD technologies: structure in multiple layers.

6.4.2 Principle of CD Magnetic-Optical

- The magneto-optical technique is based on the principle that at higher temperatures, a weak magnetic field is needed to polarize the dipoles in certain materials.
- The block (sector) to be written is heated to above $150°C$.
- At the same time, a magnetic field about ten times the strength of the Earth's magnetic field is applied.

- At this point, the material's dipoles are polarized towards this magnetic field.
- A pit is coded with a downwards-facing magnetic north pole.
- A land is coded using the opposite orientation.

In order to erase a block (sector), the area around the block is subjected to a constant magnetic field while it is heated.

If the CD is illuminated by a laser, the polarization of the light changes depending on the magnetization of the CD. In this way, the information can be read.

Areas of a CD-MO

- A CD-MO consists of an optional read-only area and the actual rewritable area.
- The read only area (premastered area in Figure 6.11) contains data written on the disc in a format specified for this purpose.
- In Figure 6.11, this relationship is indicated using the arrows between the premastered area of a CD-MO and the read only technologies.
- Thus, the read only area can be read using existing playback devices.

The rewritable area of a CD-MO cannot be read using existing CD-DA, CD-ROM, CD-ROM/XA, or CD-WO devices due to the fundamentally different technology used for reading and writing. Figure 6.11 shows the relationships between this recordable area and the underlying magneto-optical technology. This technology is thus incompatible with all the other CD technologies.

CHAPTER

7

DOCUMENTS, HYPERTEXT AND MHEG

7.1 Document Architecture and Multimedia Integration

Document is a set of structured information which may be present in various media and may be generated or input at the time of presentation. A document is used by humans and is available for editing in a computer.

Documents

- A *multimedia document* is characterized by information which is coded in at least one continuous (time-dependent) and one discrete (time-independent) medium.

- The integration of various media is possible because of close relationships between information units.

- This is also called *synchronization*.

- A multimedia document should be viewed in the environment of tools, data abstractions, basic concepts, and document architectures.

Fundamental system concepts use abstractions of multimedia data, they serve as a concept for information architectures, and they can be implemented by use of tools. In this respect, the terms *document architecture* and *information architecture* are synonymous.

7.1.1 Document Architecture

- Exchange of documents means the communication of both content and structure.
- In addition to common communication protocols, it also requires the use of a document architecture.
- This includes standardized architectures, such as SGML (Standard Generalized Markup Language).
- Such information architectures use data abstractions and their concepts for specification and implementation.
- A document architecture describes the interplay of models (see Figure 7.1).

Figure 7.2 shows a multimedia information architecture characterized by the internal interplay of individual information units from discrete and continuous media.

In the interplay of models, the manipulation describes operations that can be performed on multimedia information.

- *Communication* and *storage* define the protocols used to exchange this information between different computers and the format used to store data.
- The *presentation* of multimedia information collects relationships between individual parts of information, which have to be maintained in the presentation of this information.

Not every architecture includes all properties or models mentioned here.

7.1.2 Multimedia Integration

See Section 8.4.2, Page No. 164.

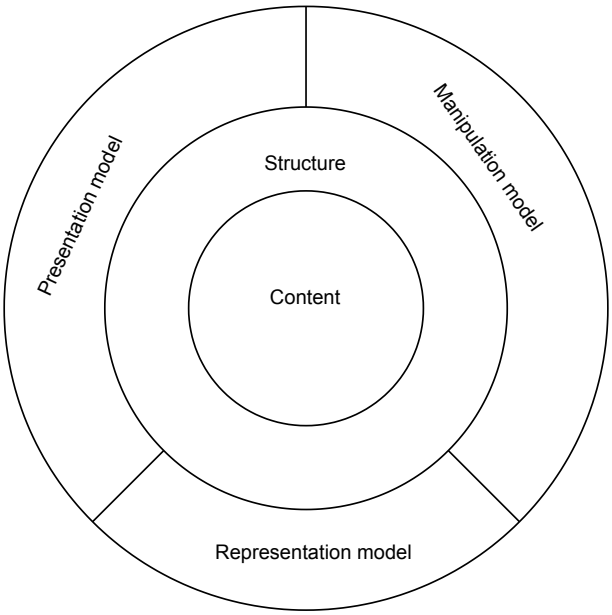

Figure 7.1: Architecture of documents and their components.

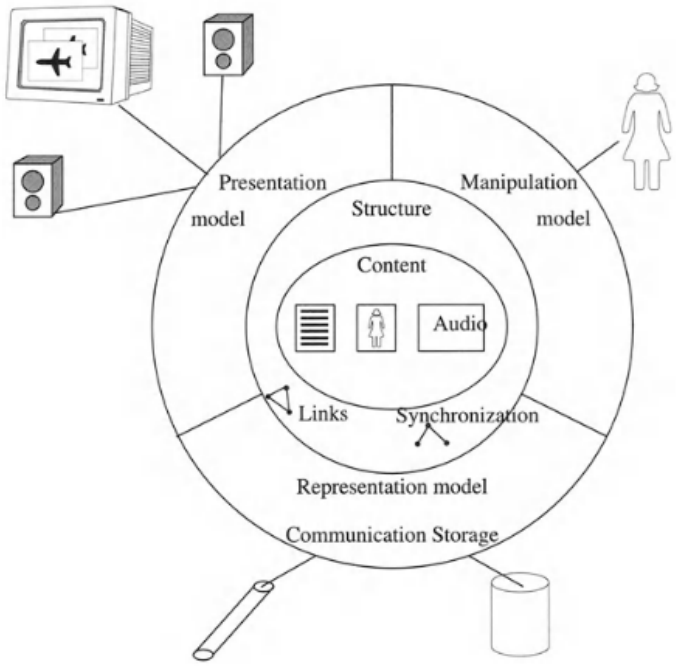

Figure 7.2: Architecture of multimedia documents and their components.

7.2 Hypertext, Hypermedia and Multimedia

A book or an article in paper form has a pre-determined structure and is present in sequential form. However, we can read specific sections in a targeted way without having read previous sections. Authors normally assume sequential reading, so that many sections are based on knowledge previously acquired. Both fictional literature and movies always assume a purely sequential reception. Scientific literature can consist of independent chapters. However, in this context too, sequential reading is assumed.

Technical documentation (e. g. manuals) consists often of a collection of relatively independent information units. A lexicon or reference book about the Airbus, for example, is generated by several authors and always only parts are read sequentially. There also exist many cross-references in such documentations which lead to multiple searches at different places for the reader. Here, an electronic help facility, consisting of information links, can be very significant.

Figure 7.3 shows an example of such a concatenation or linking.

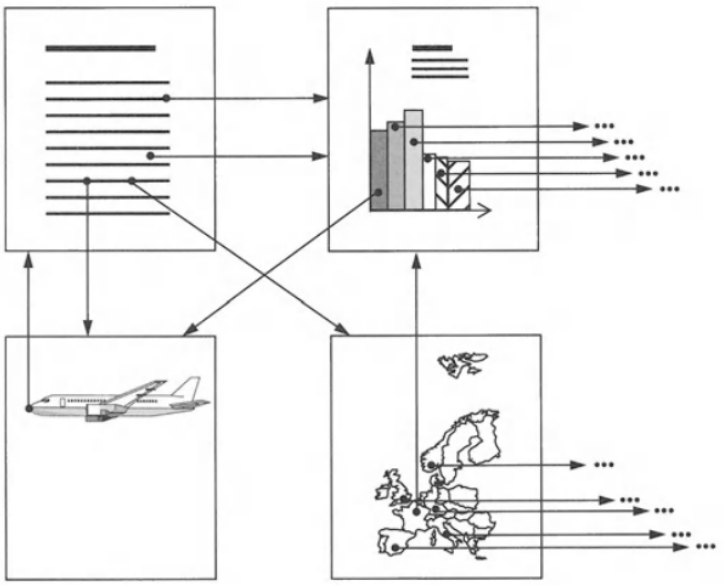

Figure 7.3: Hypertext data. An example of linking information of different media.

1. Each arrow specifies a relationship between logical information units (LDUs).
2. A piece of text (top left in the figure) includes a reference to the climb properties of an air-

craft.

3. These properties are demonstrated in a video sequence (bottom left in the figure).

4. At a different location in the same text, the sales subsidiaries located in Europe are listed (and the list is visualized in a graphical map, shown at the bottom right in the figure).

5. More information about each sales point can be viewed by selecting that location in the graphical map.

6. A special piece of information about the number of different aircrafts sold in the Paris subsidiary is shown in the form of a bar chart (top right in the figure).

7. Internally, all information contained in this chart are present only in tabular form.

8. The left bar refers to the aircraft, which can be seen in a video.

Non-linear Linking of Information

1. The most important property of hypertext and hypermedia is non-linear information linking.

2. There is a reading sequence, but the reader also decides about the reading path.

3. For example, when browsing in a lexicon, the reader can start with the term hypertext and then use the cross-reference systems to jump to a description of AppleTalk.

4. Using this association, based on reference links, the author of the information normally determines the links.

A hypertext structure is a *graph*, consisting of *nodes* and *edges*.

(a) The *nodes* are the actual information units.

(b) They are, for example, the text elements, individual graphics, audio or video LDUs.

(c) The *edges* form a relationship between different information units. They are called *reference* or *link*.

(d) A reference is normally a directed edge.

(e) All linking elements contain information.

Figure 7.4 shows the relationship between multimedia, hypertext, and hypermedia.

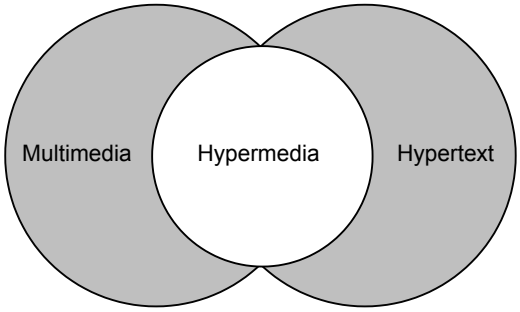

Figure 7.4: Hypertext, hypermedia and multimedia, and their relationships.

7.2.1 Hypertext System

- A hypertext system is essentially characterized by non-linear linkage of information.
- Pointers/References connect the nodes.
- The data of various nodes can be present in one medium or several media.
- In a pure text system, only the text sections are linked.
- The term *hypertext* means that several media can be linked, in addition to simple text.

7.2.2 Multimedia System

- A multimedia system contains information encoded at least in one continuous and one discrete medium.
- For example, when text data are connected by a non-linear link, then this belongs to the hypertext category.
- A video conference with simultaneous transmission of text and graphics from a conventional linear program for document editing is a multimedia application.

7.2.3 Hypermedia System

- A hypermedia system includes the non-linear linkage of information which is normally present in either a continuous or a discrete medium.
- For example, text and video data within a non-linear linkage structure belong to the hypermedia, multimedia, and hypertext categories.

Figure 7.4 shows that each hypermedia system belongs to the hypertext category and is a multimedia system at the same time. It forms the intersection set of both multimedia and hypertext.

Application Areas of of Hypermedia

- *Computer-based applications* to include the help function of modern graphical interface.
- *Commercial applications* include repair and operating instructions.
- *Intellectual applications* include organization of ideas, brainstorming, or text creation.
- *Education and research* are areas with a large potential for improvement by use of continuous media.
- The *entertainment industry* has used hypertext for information or guiding systems for tourists and interactive science-fiction movies.

7.3 Systems: Architecture, Nodes and Pointers

7.3.1 Architecture

The architecture of a hypertext system can be divided into *three levels* with different functionalities.

1. *Presentation Level*
2. *Hypertext Abstract Machine*
3. *Storage/Databse Level*

7.3.1.1 Top Level: Presentation Level

- The top or *presentation level* accommodates all functions relating to the user interface.
- This level is used to map nodes and references to the user interface.
- The user interface offers the visualization of one or several sections.
- This level determines the data to be displayed and how it should be represented based on the structure and the display chosen by the user.
- This level is also responsible for the control of all inputs.

7.3.1.2 Hypertext Abstract Machine (HAM)

- The HAM is located between the presentation level (level One) and the storage level (level Two).
- This level takes database-like functions to store multimedia data within a local or distributed environment from the lower level.
- HAM level does not have to worry about the input and output of the upper level.
- HAM knows the structure of a document, and it disposes of knowledge about the references and their attributes.
- This means that the HAM level builds the data structure, or an object hierarchy, for document management.

Compared to the other two levels, the HAM level is the one least dependent on the system. This means that it is also best suitable for standardization.

7.3.1.3 Storage/Database Level

- The *storage level* (also called *database level*) forms the lowest level.
- It includes all functions relating to the storage of data, i. e. , secondary storage management.
- In this respect, the specific properties of the different sets of data from discrete and continuous media are important.
- This is also the level where capabilities of traditional database systems, i. e. , persistence, multi-user operation (synchronization, locking), and error recovery (transaction concept), are expected.
- The nodes and pointers of a hypertext document are processed as data objects without any special semantics.

Unfortunately, most implementations of hypertext systems do not clearly separate these different levels. The reasons are normally a shorter development phase, a more efficient implementation, and shortcomings in most generally available multimedia interfaces for the lowest level.

7.3.2 Nodes

A node is an information unit (LDU) in a hypertext document. The most important distinguishing criterion between different implementations is the

maximum data quantity a node can accommodate.

7.3.2.1 Maximum Data Quantity

- The *maximum data quantity* contained in a node can be limited to match the screen size.
- A motion picture sequence and an audio clip could be limited to a duration of, for example, 36 seconds.

An author is forced eventually to distribute logical connected text content to several cards, although it is not desired. Applying it to video clips and audio passages, it would mean that the close interconnection among the distributed sequences could get lost easily. An advantage is the *clear and intuitive environment.*

7.3.2.2 Unlimited Data Quantity

- One alternative are window-based systems with a basically *unlimited data quantity* per node.
- Forward and backward scrolling of pages is offered analogous to other windows at the user interface.
- *Intermedia* is such a system. Intermedia does not limit the duration of continuous media with regard to the data volume its nodes can accommodate.

This means that single nodes can have very different lengths and still appear equal-ranking. The presentation of additional information also uses two different methods at the user interface:

- First, there is a way to switch between the nodes,
- and second, a node uses the mechanisms known from window-based systems for *scrolling.*

A secondary criterion concerns the *time when a piece of information is created.* In general, the author can specify the entire contents of the nodes while creating a document.

7.3.3 Pointers

Pointers/References form the edges in a hypertext graph. Hypertext systems differ by various edge criteria. One of the first questions to ask here would be: *Which information is contained in a pointer?*

- *Simple pointers* connect two nodes of a graph without containing any information themselves. They merely establish a relationship between those nodes.
- *Typified pointers* connect two nodes and include additional information. A label is assigned to each reference. This label is used to create comments to each reference (e. g. , author and creation date).

Another Property of the pointers is connected to the question: *What does the pointer mean?* Often, pointers with very different meanings are used together. This usage complicates the understanding. The author of a hypertext should know about this problem and use unambiguous pointers. The following relations can be expressed through pointers:

- *To be* - A is part of B. This represents a quantity relationship.
- *To present* - A is an example of B, or A demonstrates B. This case uses an example to state a fact.
- *To produce* - A produces B, or B is a result of A. This case can describe consequences from a fact in more detail.
- *To require or required by* - A requires B, B needs A. This relationship expresses an absolute necessity.
- *To own* - A owns B, or A is associated to B. This relationship expresses an ownership.
- *To include* - A includes B, or A consists of B, or A occurs in B. This relationship represents different meanings of an inclusion.
- *Similarity* - A is similar to B, A is different from B, A replaces B, A is the alternative to B. This relationship expresses similarities.

Another fundamental property of pointers can be described by the following question: *Who states a pointer?* We distinguish as follows:

- *Implicit References*: A relationship between nodes can be created automatically by a hypertext system. The author specifies only the algorithms used to create the references.
- *Explicit References*: All references are created by the author.

A reference can be created at different times, which leads us to the question: *When is the target of a pointer stated?*

- In the classical case, a reference is created during the creation of the hypertext document, and both the origin and the target node are specified. When editing the document, the author specifies explicitly how the information units should be linked.
- A target node can be determined when the reference is used, i. e. , while the document is read. The author specifies an algorithm to be used to create the references, while the references are determined when the user reads the context. The system calculates the target node.

In most systems, however, each pointer has exactly one origin and one target node. On the other hand, we could ask the following question:

- *What direction does the pointer have?* and
- *What is the number of outgoing pointers?*

- The direction is normally unidirectional, while the system itself supports *backtracking*. This means that we would always get back on track.
- The alternative would be bidirectional references, which means that we would have to highlight or mark both the target nodes and the anchors. When introducing bidirectional references, it could easily happen that several nodes refer to the same target nodes. This means that these references have to be explicitly distinguished at the target node.
- The same applies to references leading from one origin to several target nodes.
- Most systems support unidirectional references with only one target node each. This is easier to understand and to implement.

As our last question, we would have to deal with the appearance of an anchor on the user interface. More specifically, we can ask the following question: *How can a reference be represented?*

Anchors

The forward movement in linear sorted documents is called a *navigation* through the graph. At the user interface, the origin of references has to be marked, so that the user can jump from one information unit to another. This origin of a pointer is called an *anchor*.

- A *media-independent representation* can be realized with general graphic elements, like the ones commonly used for selection, e. g. , buttons. The information about the target node should be included in such an element:
  - If the target node is a piece of text, then an abbreviated, descriptive (verbose) summary of the contents could be represented.
  - For a single image, the respective image contents could be displayed on the screen in the form of a miniature or "thumbnail".
  - A visual representation of video contents could be implemented in the form of moving icons (micons[1]).
  - If the contents of the target node consist of audio information, then the audio contents should be represented visually. For a music clip, for example, this could be a picture of the composer
- For *text*, we could emphasize single words, paragraphs, or text sections of different lengths. A pointer can be positioned on a section emphasized in this way, so that the user could double-click the highlighted section to display the target node referred to by the pointer
- For *single images*, specific graphic objects or simply areas are defined as selection objects. A specific marking can occur through a color or stripe.
- For *motion pictures*, media-independent representations of anchors are the preferred method.
- For audio, we would always have to use a media-independent solution. In this case, we would use a brief descriptive text or a single image of the size of an icon.

Tools

A hypertext system consists of several required tools:

- One or more *editors* are used to edit information from various media.
- *Search tools* are used to find information.
- A *browser* is used to obtain a

[1] A micon is a reduced motion picture, representing a characteristic part of the video sequence of the target node.

short and clear representation of the nodes and edges.

- During the *navigation* through a document, a proper support of the phenomena *Getting Lost in the Hyperspace* is needed. A *backtracking* and clear representation of the whole structure with respect to the actual position should be available.

7.4 Architecture: SGML and ODA and MHEG

7.4.1 SGM Document Architecture

The *Standard Generalized Markup Language* (SGML) has been promoted mainly by US publishing companies. The authors provide text, i. e. contents. They use a uniform markup format to describe titles, tables, and other document elements, without actually describing the document's look (e. g. , fonts or line spacing). The final layout is determined by the publisher.

- The basic idea is that the author uses tags to mark specific text parts.
- SGML defines the look of these tags, but it does not define where they should occur, or what meaning they should have.
- Instead, user groups agree on the meaning of such tags.
- SGML offers a frame that can be used to describe the syntax in a way specific to a user group within an object-oriented system.
- Classes and objects, hierarchies of classes and objects, inheritance, and embedded methods (processing instructions) can be used in the specification.
- SGML defines the syntax but no semantics.

The following example shows the use of SGML in a text document:

```
<title>Multimedia Application</title>
<course>BCA</course>
<semester>VIII</semester>
<author>Jeevan Poudel</author>
<year>2077</year>
<summary>This is for BCA  ...
...
```

Using SGML

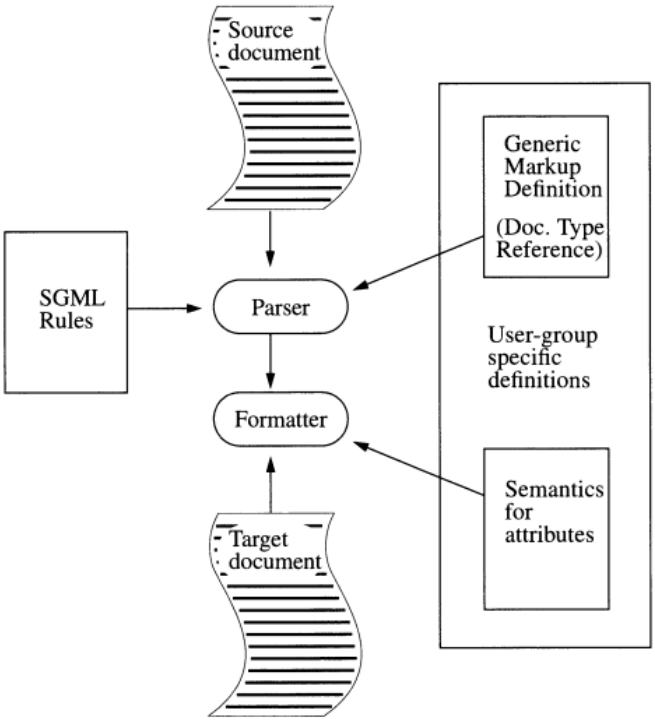

Figure 7.5: SGML document processing, from information to representation.

The handling process for an SGML document shown in Figure 7.5 splits the processing job into two processes:

- The *formatter* knows the meaning of the tags, and uses these tags to produce a formatted document.
- The *parser* uses the tags contained in the document in combination with the appropriate document type.

Tags can be used to define the structure of a document, which normally includes the association of some parts of the layout. It is based on the common context between the creator of the document and the formatting process, but not defined by SGML. There are various tag categories:

- *Descriptive markup tags* describe the actual structure in the following form:

```
<start-tag> or alternatively <end-tag>
```

One example is the definition of the beginning of a paragraph:

```
<paragraph> The text of this paragraph begins here ...
```

- An *entity reference* points to another element, which then replaces the entity reference. This could also be interpreted as an abbreviation, which is later overwritten by copying the actual content in its place. The following example shows entity reference in a mathematical context:

  &square x...should be x^2

- The *markup declarations* define the elements to which an entity reference refers. In our example of squaring a variable x, square is defined as:

```
  <!ELEMENT square (...)>
```

  A markup declaration can also be used to define rules for the structure (classes). The following example defines the structure of an article *paper*:

```
<!ELEMENT paper (preamble, body, postamble)>
<!ELEMENT preamble (title, author, side)>
<!ELEMENT title (#CDATA)> -- character data
<!ELEMENT body (...) >
```

- *Processing Instructions* are used to embed instructions for other programs into a piece of text, e. g. , instructions for the formatter. Processing instructions can also be used to insert various media, e. g. , images and video.

SGML uses a grammar for tags to define a syntax that has to be observed. It does not define the meaning of these tags.

Figure 7.6 shows the information or document architecture of SGML. Using its tags, SGML has a representation model. *Objects*, *classes*, and *inheritance* can be used to define the structure.

SGML and Multimedia

- SGML standard supports multimedia data only in the form of images.
- An image is embedded into

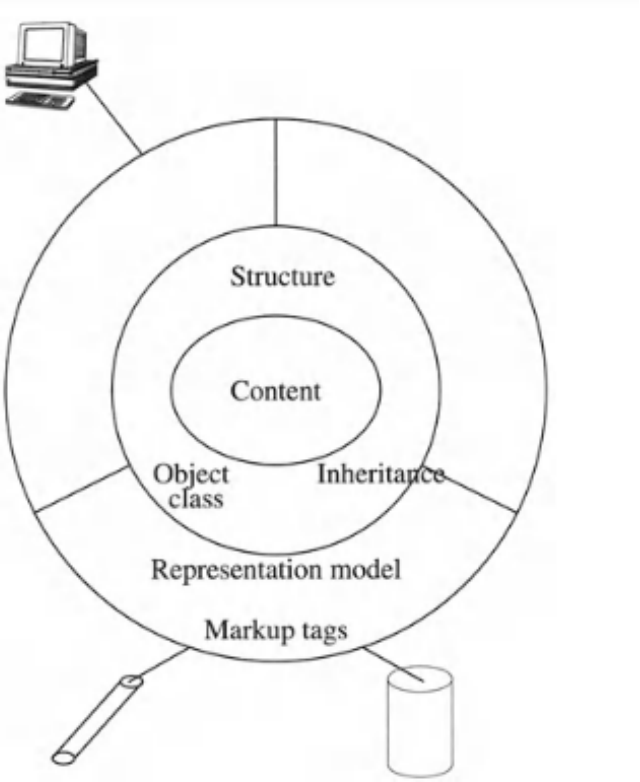

Figure 7.6: The SGML document architecture focuses on the representation model.

an SGML document in CGM (*Computer Graphics Metafile*) form.

- The standard does not include specifications or recommendations for other media.

```
<!ATTLIST video   id    ID #IMPLIED>
<!ATTLIST video   synch   synch #IMPLIED>
<!ELEMENT video   (audio, movpic)>
<!ELEMENT audio   (#NDATA)) -- non-text media
<!ELEMENT movpic  (#NDATA)) -- non-text media
...
<!ELEMENT story   (preamble, body, postamble)) :
```

`#NDATA` can be used to set a reference that points to specific data. This data is normally external, i. e. , stored in a separate file. The above example shows a video definition, consisting of audio and motion pictures.

7.4.2 Document Architecture ODA

The *Open Document Architecture (ODA)* was initially called the *Office Document Architecture* because it supports mostly office-oriented applications. The main goal of this document architecture is to support the exchange, processing and presentation of documents in open systems.

ODA

- The main property of ODA is the distinction among *content*, *logical* structure and *layout* structure.
- This is in contrast to SGML where only a logical structure and the contents are defined.
- ODA also defines semantics.

Figure 7.7 shows these three aspects linked to a document. One can imagine these aspects as three orthogonal views of the same document. Each of these views represent one aspect, together we get the actual document.

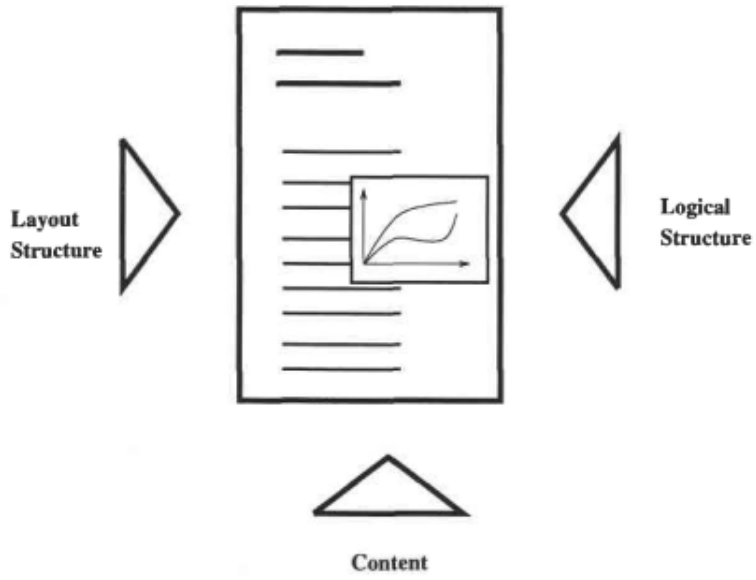

Figure 7.7: ODA: Content, layout and logical view.

Content Portions

- The content of the document consists of *Content Portions*.
- These can be manipulated according to the corresponding medium.
- A content architecture describes for each medium:
  - the specification of the elements
  - the possible access functions and
  - the data coding
- Individual elements are the Logical Data Units (LDUs), which are determined for each medium.
- The access functions serve for the manipulation of individual elements.
- The coding of the data determines the mapping with respect to bits and bytes.
- ODA has content architectures for media text, geometrical graphics and raster graphics.

Layout Structure and Logical Structure

The structure and presentation models describe — according to the information architecture — the cooperation of information units. These kinds of meta information distinguish layout and logical structure.

Layout Structure

- The *layout structure* specifies mainly the representation of a document.
- It is related to a two-dimensional representation with respect to a screen or paper.
- The presentation model is a tree.
- Using *frames* the position and size of individual layout elements is established.
- For example, the page size and type style are also determined.

Logical Structure

- The *logical structure* includes the partitioning of the content.
- Here, paragraphs and individual headings are specified according to the tree structure.

7.4.3 MHEG

The committee ISO/IEC JTC1/SC29 (*Coding of Audio, Picture, Multimedia and Hypermedia Information*) works on the standardization of the exchange format for multimedia systems. The actual standards are developed at the international level in three working groups cooperating with research and industry. The results of the working groups: the *Joint Photographic Expert Group (JPEG)* and the *Motion Picture Expert Group (MPEG)* are of special importance in the area of multimedia systems.

In a multimedia presentation, the contents, in the form of individual information objects, are described with the help of the above named standards. The structure is specified first through timely spatial relations between the information objects. The standard of this structure description is the subject of the working group WG12, which is known as the *Multimedia and Hypermedia Information Coding Expert Group (MHEG)*. The name of the developed standard is officially called *Information Technology — Coding of Multimedia and Hypermedia Information (MHEG)*.

7.4.3.1 Example of an Interactive Multimedia Presentation

Figure 7.8 presents a time diagram of an interactive multimedia presentation. The presentation starts with some music. As soon as the voice of a newsspeaker is heard in the audio sequence, a graphic should appear on the screen for a couple of seconds. After the graphic disappears, the viewer carefully reads a text. After the text presentation ends, a Stop button appears on the screen. With this button the user can abort the audio sequence. Now, using a displayed input field, the user enters the title of a desired video sequence. These video data are displayed immediately after the modification.

Content

- A presentation consists of a sequence of information representations.
- For the representation of this information, media with very different properties are available.
- Because of later *reuse*, it is useful to capture each information as an individual object.
- The contents in our example are:
  - the video sequence,
  - the audio sequence,
  - the graphics, and
  - the text.

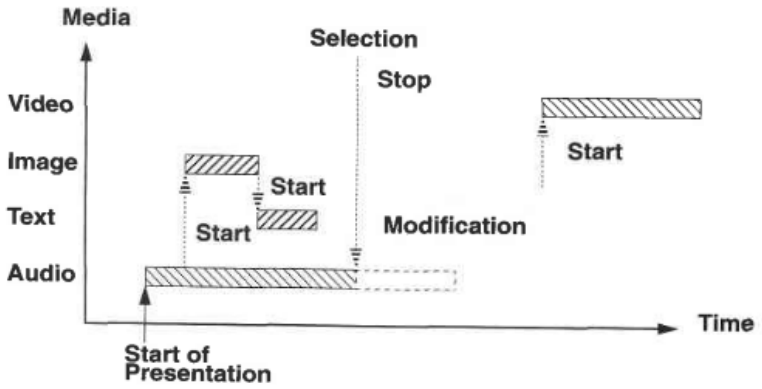

Figure 7.8: The timing diagram of an interactive presentation.

Behavior

- The notion *behavior* means all information which specifies the representation of the contents as well as defines the run of the presentation.
- The first part is controlled by the actions *start*, *set volume*, *set position*, etc.
- The last part is generated by the definition of timely, spatial and conditional links between individual elements.
- If the state of the content's presentation changes, then this may result in further commands on other objects (e. g. , the deletion of the graphic causes the display of the text).
- Another possibility, how the behavior of a presentation can be determined, is when external programs or functions (script) are called.

User Interaction

- In the discussed scenario, the running animation could be aborted by a corresponding user interaction.
- There can be two kinds of user interactions.
  - *simple selection*, which controls the run of the presentation through a prespecified choice (e. g. , push the Stop button).
  - complex *modification* which gives the user the possibility to enter data during the run of the presentation (e. g. editing of a data input field).

Container

- Merging together several elements as discussed above, a presentation, which progresses in time, can be achieved.
- To be able to exchange this presentation between the involved systems, a *composite* element is necessary.
- This element is comparable to a container.
- It links together all the objects into a unit.
- With respect to hypertext/ hypermedia documents, such containers can be ordered to a complex structure, if they are linked together through so-called hypertext pointers.

7.4.3.2 MHEG Class Hierarchy

Figure 7.9 summarizes the individual elements in the MHEG class hierarchy in the form of a tree.

- Instances can be created from all leaves(roman printed classes).
- All internal nodes, including the root (*italic* printed classes), are abstract classes, i. e. , no instances can be generated from them.
- The leaves inherit some attributes from the root of the tree as an abstract basic class.
- The internal nodes do not include any further functions.
- Their task is to unify individual classes into meaningful groups.
- The **action**, the **link** and the **script** classes are grouped under the **behavior** class, which defines the behavior in a presentation.
- The **interaction** class includes the user interaction, which is again modeled through the **selection** and **modification** class.
- All the classes together with the **content** and **composite** classes specify the individual components in the presentation and determine the **component** class.
- Some properties of the particular MHEG engine can be queried by the **descriptor** class
- The **macro** class serves as the simplification of the access, respectively reuse of objects. Both classes play a minor role.

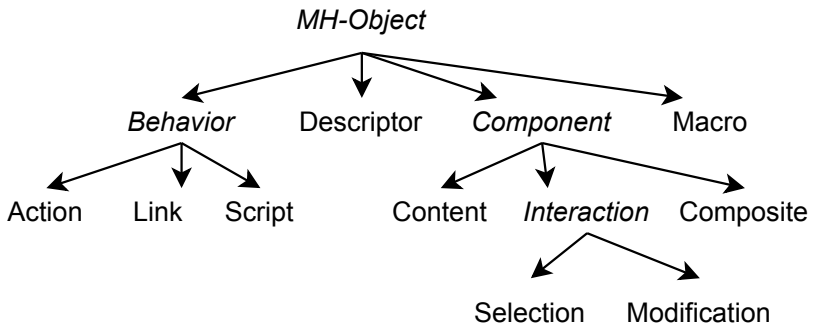

Figure 7.9: Class hierarchy of MHEG objects.

MH-Object Class The abstract MH-Object Class inherits both data structures *MHEG identifier* and *Descriptor*.

- MHEG identifier consists of the attributes *MHEG identifier* and *Object Number* and it serves as the addressing of MHEG objects.
- The first attribute identifies a specific application.
- The *Object Number* is a number which is defined only within the application.
- The data structure *Descriptor* provides the possibility to characterize more precisely each MHEG object through a number of optional attributes.
- For example, this can become meaningful if a presentation is decomposed into individual objects and the individual MHEG objects are stored in a database.
- Any author, supported by proper search functions, can reuse existing MHEG objects.

CHAPTER

8

ADVANCED TECHNOLOGIES IN MULTIMEDIA

8.1 Multimedia Operating System

8.1.1 Introduction

The operating system is the shield of the computer hardware against all software components. It provides a comfortable environment for the execution of programs, and it ensures effective utilization of the computer hardware. The operating system offers various services related to the essential resources of a computer: CPU, main memory, storage and all input and output devices.

For the processing of audio and video, multimedia application demands that humans perceive these media in a natural, error-free way. These continuous media data originate at sources like microphones, cameras and files. From these sources, the data are transferred to destinations like loudspeakers, video windows and files located at the same computer or at a remote station. On the way from source to sink, the digital data are processed by at least some type of move, copy or transmit operation. In this data manipulation process there are always many resources which are under the control of the operating system. The integration of discrete and continuous multimedia

data demands additional services from many operating system components.

8.1.2 Resource Management

Multimedia systems with their integrated audio and video processes often reach their capacity limits despite the usage of data compression and utilization of the newest computing and communication technologies. Current multimedia systems require therefore their own resource management; here the availability of resource reservation may provide an advantage for quality provision in these systems.

The actual requirements depend on the *type of media* and the *nature of the applications* supported.

In integrated distributed multimedia system, several applications compete for system resources. This shortage of resources requires careful allocation. The system management must employ adequate scheduling algorithms to serve the requirements of the applications. Thereby, the resource is first allocated and then managed

Resource management in distributed multimedia systems covers several computers and the involved communication networks. It allocates all resources involved in the data transfer process between sources and sinks

Applied to operating systems, resource management covers the CPU (including process management), memory management, the file system and device management

8.1.2.1 Resources

A resource is a system entity required by tasks for manipulating data. Each resource has a set of distinguished characteristics, and they can be classified as follows:

- A resource can be *active* or *passive*:
  - An *active resource* is the CPU or a network adapter for protocol processing; it provides a service.
  - A *passive resourc*e is the main memory, communication bandwidth or a file system; it denotes some system capability required by active resources.
- A resource can be either used *exclusively* one process at a time or *shared* between various processes
  - Active resources are often exclusive

  - passive resources can usually be shared among processes

- A resource that exists only once in the system is known as a *single*, otherwise it is a *multiple* resource.

8.1.2.2 Requirements

The requirements of multimedia applications and data streams must be served by the single components of a multimedia system. The transmission and processing requirements of local and distributed multimedia applications can be specified according to the following characteristics:

- *Throughput* can be determined from the needed data rate and the data unit size, transmitted over a connection.

- *Delay* needs to be distinguished between the local and the global (end-to-end) delay:

  - *Local delay* is the maximum time span for the completion of a certain task at this resource.
  - *End-to-end delay* is the total delay for a data unit to be transmitted from the source to its destination.

- *Jitter* (or delay jitter) determines the maximum allowed variance in the arrival of data at the destination.

- *Reliability* maps to the error handling algorithms. The reliability defines error detection and correction mechanisms used for the transmission and processing of multimedia tasks.

In accordance with communication systems, these requirements are also known as *Quality of Service parameters (QoS)*.

8.1.2.3 Allocation Scheme

Reservation of resources can be made either in a pessimistic or optimistic way:

- The *pessimistic approach* avoids resource conflicts by making reservations for the worst case, i. e. resource bandwidth for the longest processing time and the highest rate which might ever be needed by a task is reserved. Resource conflicts are therefore avoided.

- With the *optimistic approach*, resources are reserved according to an average workload only. This means that the CPU is only reserved for the average processing time.

The optimistic approach is considered to be an extension of the pessimistic approach. It requires that additional mechanisms to detect and solve resource conflicts be implemented.

8.1.2.4 Continuous Media Resource Model

To present more precisely QoS parameters and the multimedia stream characteristics at the system level, often a continuous media model is considered.

Continuous media model is frequently adopted to define QoS parameters and hence, the characteristics of the data stream. It is based on the model of *Linear Bounded Arrival Processes (LBAP)*. In this model a distributed system is decomposed into a chain of resources traversed by the messages on their end-to-end path.

LBAP model is a message arrival process at a resource defined by three parameters:

1. *Maximum message size* M, measured in bytes per message ($bytes/Msg$)
2. *Maximum message rate* R, measured in messages per second (Msg/s), and
3. *Maximum burstiness* B measured in messages (Msg).

Example The LBAP model is discussed in terms of a specific example: two workstations are interconnected by a LAN. A CD player is attached to one workstation. Single channel audio data are transferred from the CD player to this workstation over the network to the other computer. The audio signal is sampled with the frequency of $44.1kHz$. Each sample is coded with $16bits$. This results in a data rate of:

$$R_{byte} = 44,100Hz \times \frac{16bits}{8bits/byte} = 88,200bytes/s$$

The samples on a CD are assembled into frames. These frames are the audio messages to be transmitted. 75 of these audio messages are transmitted per second. Therefore, we encounter a maximum message size of:

$$M = \frac{88,200byte/s}{75Msg/s} = 1,176byte/Msg$$

Up to $12,000 bytes$ are assembled into one packet and transmitted over LAN. In a packet of $12,000 bytes$ transmitted over the LAN, we will never encounter more messages than:

$$\frac{12,000 byte}{1,176 byte/Msg} \geq 10 Msg = B$$

It obviously follows that:

- $M = 1176 bytes/Msg$
- $R = 75 Msg/s$
- $B = 10 Msg$

Burst

In the following calculation we assume that because of lower adjacent data rate, a burst never exceeds the maximum data rate. Hence, bursts do not succeed one another. During the time interval of length t, the maximum number of messages arriving at a resource must not exceed:

$$\bar{M} = B + R \times t$$

For example, if we assume $t = 1s$, then

$$\bar{M} = 10 Msg + 75 Msg/s \times 1s = 85 Msg$$

The introduction of burstiness B allows for short time violations of the rate constraint.

Maximum Average Data Rate

The maximum average data rate of the LBAP can be calculated as follows:

$$R = M \times R$$

For example:

$$R = 1,176 byte/Msg \times 75 Msg/s = 88,200 byte/s$$

8.1.3 Process Management

Process management deals with the resource *main processor*. The capacity of this resource is specified as processor capacity. The process manager maps single processes onto resources according to a specified scheduling policy such that all processes meet their requirements. In most systems, a process under control of the process manager can adopt one of the following states:

- In the **initial state**, no process is assigned to the program. The process is in the idle state.
- If a process is waiting for an event, i. e. , the process lacks one of the necessary resources for processing. It is in the **blocked state**.
- If all necessary resources are assigned to the process, it is ready to run. The process only needs the processor for the execution of the program. The process is in the **ready state**.
- A process is running as long as the system processor is assigned to it. In this case, the process is in **running state**.

The process manager is the *scheduler*. This entity transfers a process into the ready-to-run state by assigning it a position in the respective queue of the dispatcher, which is the essential part of the operating system kernel. The dispatcher manages the context switch and hence the transition from the ready-to-run to the run state. In most operating systems, the next process to run is chosen according to a priority policy. Between processes with the same priority, the one with the longest ready time is chosen.

8.1.3.1 Real-time Processing Requirement

Continuous media data processing must occur in exactly predetermined-usually periodic-intervals. Operations on these data recur over and over and must be completed at certain deadlines. The real-time process manager determines a schedule for the CPU resource CPU that allows it to make reservations and to give processing guarantees. The problem is to find a feasible scheduler which schedules all time-critical continuous media tasks in a way that each of them can meet its deadlines. This must be guaranteed for all tasks in every period for the whole run-time of the system. In a multimedia system, continuous and discrete media data are processed concurrently.

For scheduling of multimedia tasks, two conflicting goals must be considered:

- An uncritical process should not suffer from starvation because time-critical processes are executed. Multimedia applications rely as much on text and graphics as on audio and video.
- On the other hand, a time-critical process must never be subject to priority inversion.

Realtime requirements of multimedia processes can be partitioned into two groups:

1. hard-realtime requirements
2. soft-realtime requirements

Hard-realtime requirements mean that each deadline of a multimedia process must be guaranteed by the process management.

Soft-realtime requirements mean that most of the deadlines of a multimedia process are guaranteed by the process management, but some deadlines can be violated over the duration of the task without any catastrophic consequences.

Traditional Real-time Scheduling

The problem of realtime processing is widely known in computer science. Some realtime scheduling methods are employed in operations research. They differ from computer science realtime scheduling because they operate in a static environment, where no adaptation to changes of the workload is necessary.

The goal of traditional scheduling on time-sharing computers is optimal throughput, optimal resource utilization and fair queuing. In contrast, the main goal of realtime tasks is to provide a schedule that allows all, respectively, as many time-critical processes as possible, to be processed in time, according to their deadline.

There are several attempts to solve real-time scheduling problems. Many of them are just variations of basic algorithms. To find the best solutions for multimedia systems, two basic algorithms are analyzed:

1. *Earliest Deadline First* Algorithm
2. *Rate Monotonic Scheduling*

8.1.3.2 Real-time Scheduling

All scheduling algorithms to be introduced are based on the following system model for the scheduling of real-time tasks. Their essential components are the resources, tasks and scheduling goals.

A *task* is a schedulable entity of the system, and it corresponds to the notion of a thread in the previous description. In a hard real-time system, a task is characterized by its timing constraints, as well as by its resource requirements.

The time constraints of the periodic task T are characterized by the following parameters (s, e, d, p):

- s: Starting point
- e: Processing time of T
- d: Deadline of T
- p: Period of T
- r: Rate of $T (r = \frac{1}{p})$

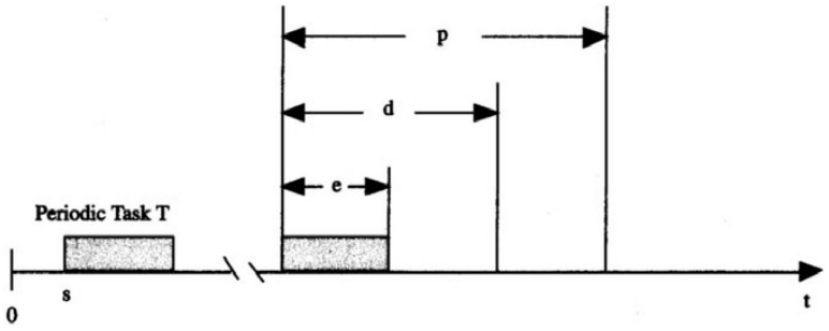

Figure 8.1: Characterization of periodic tasks.

- Whereby $0 \leq e \leq d \leq p$ (see Figure 8.1). The starting point s is the first time when the periodic task requires processing.
- Afterwards, it requires processing in every period with a processing time of e.
- At $s + (k-1)p$, the task T is ready for k-processing.
- The processing of T in period k must be finished at $s + (k-1)p + d$.
- For continuous media tasks, it is assumed that the deadline of the period $(k-1)$ is the ready time of period k.
- This is known as the congestion avoiding deadlines: the deadline for each message (d) coincides with the period of the respective periodic task (p).

Tasks can be *preemptive* or *non-preemptive*:

- A *preemptive* task can be interrupted by the request of any task with a higher priority. Processing is continued in the same state later on.
- A *non-preemptive* task cannot be interrupted until it voluntarily yields the processor. Any high-priority task must wait until the low-priority task is finished. The high-priority task is then subject to priority inversion.

A major performance metric for a real-time scheduling algorithm is the guarantee ratio. The guarantee ratio is the total number of guaranteed tasks versus the number of tasks which could be processed.

Another performance metric is the processor utilization. This is the amount of processing time used by guaranteed tasks versus the total amount of processing time:

$$U = \sum_{i=1}^{n} \frac{e_i}{p_i}$$

8.1.3.3 Earliest Deadline First (EDF) Algorithm

- One of the best-known algorithms for real-time processing.
- At every new ready state, the scheduler selects the task with the earliest deadline among the tasks.
- The requested resource is assigned to the selected task.
- At any arrival of a new task, EDF must be computed immediately leading to a new order.
- The new task is processed immediately if its deadline is earlier than that of the interrupted task.
- The processing of the interrupted task is continued according to the EDF algorithm later on.

EDF is an optimal, dynamic algorithm, i. e. it produces a valid schedule whenever one exists. With n tasks which have arbitrary ready-times and deadlines, the complexity is $\Theta(n^2)$ [यो uppercase Theta Θ हो, lowercase Theta θ होइन।].

For scheduling continuous media data on a single processor machine with priority scheduling, process priorities are likely to be rearranged quite often. A priority is assigned to each task ready for processing according to its deadline. If the computed priority of a new process is not available, the priorities of other processes must be rearranged until the required priority is free. In the worst case, the priorities of all processes must be rearranged.

8.1.3.4 Rate Monotonic Algorithm

The rate monotonic scheduling principle was introduced by Liu and Layland in 1973. It is an optimal, static, priority-driven algorithm for preemptive, periodic jobs. Optimal in this context means that there is no other static

algorithm that is able to schedule a task set which cannot be scheduled by the rate monotonic algorithm. A process is scheduled by a static algorithm at the beginning of the processing. Subsequently, each task is processed with the priority calculated at the beginning. No further scheduling is required. The following five assumptions are necessary prerequisites to apply the rate monotonic algorithm:

1. The requests for all tasks with deadlines are periodic, i. e. , have constant intervals between consecutive requests.

2. The processing of a single task must be finished before the next task of the same data stream becomes ready for execution. Deadlines consist of runability constraints only, i. e. , each task must be completed before the next request occurs.

3. All tasks are independent.

4. This means that the requests for a certain task do not depend on the initiation or completion of requests for any other task.

5. Run-time for each request of a task is constant.

6. Run-time denotes the maximum time which is required by a processor to execute the task without interruption.

7. Any non-periodic task in the system has no required deadline.

8. Typically, they initiate periodic tasks or are tasks for failure recovery.

9. They usually displace periodic tasks.

The rate monotonic algorithm is a simple method to schedule time-critical, periodic tasks on the respective resource. A task will always meet its deadline, if this can be proven to be true for the longest response time. The response time is the time span between the request and the end of processing the task. This time span is maximal when all processes with a higher priority request to be processed at the same time. This case is known as the critical instant (see Figure 8.2). In this figure, the priority of a is, according to the rate monotonic algorithm, higher than b, and b is higher than c. The critical time zone is the time interval between the critical instant and the completion of a task

8.1.4 System Architecture

The employment of continuous media in multimedia systems also imposes additional, new requirements to the system architecture. The audio and

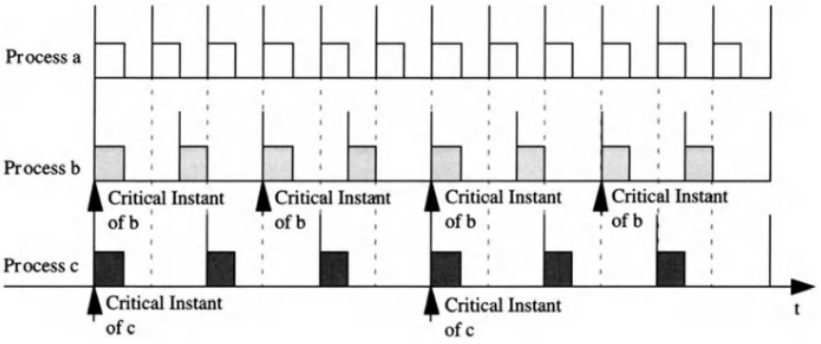

Figure 8.2: Example of critical instants.

video data need to be copied directly from adapter to adapter for acquiring the shortest possible path. A problem with direct copying from adapter to adapter is the control and the change of quality of service parameters. In multimedia systems, such an adapter to adapter connection is defined by the capabilities of the two involved adapters and the bus performance. This architecture of low-level data streaming corresponds with proposals for using additional new busses for audio and video transfer within a computer. It also enables a switch-based rather than a bus-based transfer within a computer.

The architecture of the protocol processing system is just one issue to be considered in the system architecture of multimedia supporting operating systems. Multimedia data should be delivered from the input device (e. g. CD-ROM) to an output device (e. g. a video decompression board) across the fastest possible path. The paradigm of streaming from source to sink is an appropriate way of doing this. Hence, the multimedia application opens devices, establishes a connection between them, starts the data flow and returns to other duties.

The most dominant characteristic of multimedia applications is to preserve the temporal requirement at the presentation time. Therefore, multimedia data is handled in a *Real-Time Environment (RTE)*, i. e. , its processing is scheduled according to the inherent timing requirements of multimedia data. On a multimedia computer, the RTE will usually coexist with a *Non-Real-Time Environment (NRTE)*. The NRTE deals with all data that have no timing requirements. Figure 8.3 shows the approached architecture.

Multimedia I/O devices are, in general, accessed from both environments. Data such as a video frame, for example, is passed from the RTE to the display. The RTE is controlled by related functions in the NRTE. The establishment of communication connections at the start of a stream must not obey timing requirements, but the data processing for established connections is

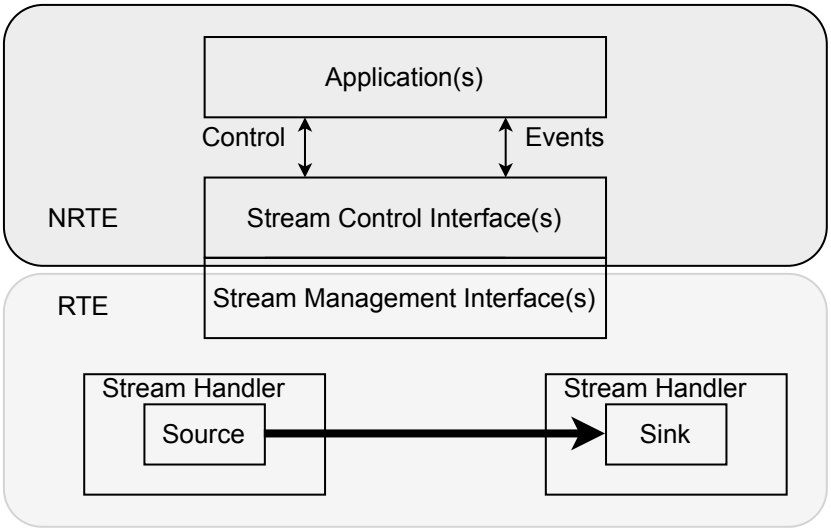

Figure 8.3: Real-time and non-real-time environments.

compelled. All control functions are performed in the NRTE.

System programs, such as communication protocol processing and database data transfer programs, make use of this programming in the RTE. Whereas applications like authoring tools and media presentation programs are relieved from the burden of programming in the RTE, they just interface and control the RTE services. Applications determine processing paths which are needed for their data processing, as well as the control devices and paths.

To reduce data copying, buffer management functions are employed in the RTE. This buffer management is located "between" the stream handlers. Stream handlers are entities in the RIE which are in charge of multimedia data.

Multimedia data usually "enters" the computer through an input device, a source and "leaves" it through an output device, a sink (where storage can serve as an I/O device in both cases). Sources and sinks are implemented by a device driver.

8.2 Multimedia Communication System

From the communication perspective, we divide the higher layers of the Multimedia Communication System (MCS) into two architectural subsystems: an *application subsystem* and a *transport subsystem*.

8.2.1 Application Subsystem

8.2.1.1 Collaborating Computing

Infrastructure of networked workstations and PCs, and the availability of audio and video at these end-points, makes it easier for people to cooperate and bridge *space* and *time*. In this way, network connectivity and end-point integration of multimedia provides users with a *collaborative computing* environment. Collaborative computing is generally known as *Computer-Supported Cooperative Work* (CSCW).

There are many tools for collaborative computing, such as:

- *electronic mail*,
- *internet relay chat (IRC)*,
- *forums*,
- *screen sharing tools*,
- *text-based conferencing systems*,
- *telephone conference systems*,
- *conference rooms* and
- *video conference systems*.

We present a framework for collaborative computing and general related issues exemplified by different systems and tools.

Collaborative Dimensions

Electronic collaboration can be categorized according to three main parameters:

- *time*
- *user scale* and
- *control*

Therefore, the collaboration space can be partitioned into a three-dimensional space (shown in Figure 8.4).

Time With respect to time, there are two modes of cooperative work: *asynchronous* and *synchronous*:

- *Asynchronous cooperative* work specifies processing activities that do not happen at the same time.
- The *synchronous cooperative* work happens at the same time.

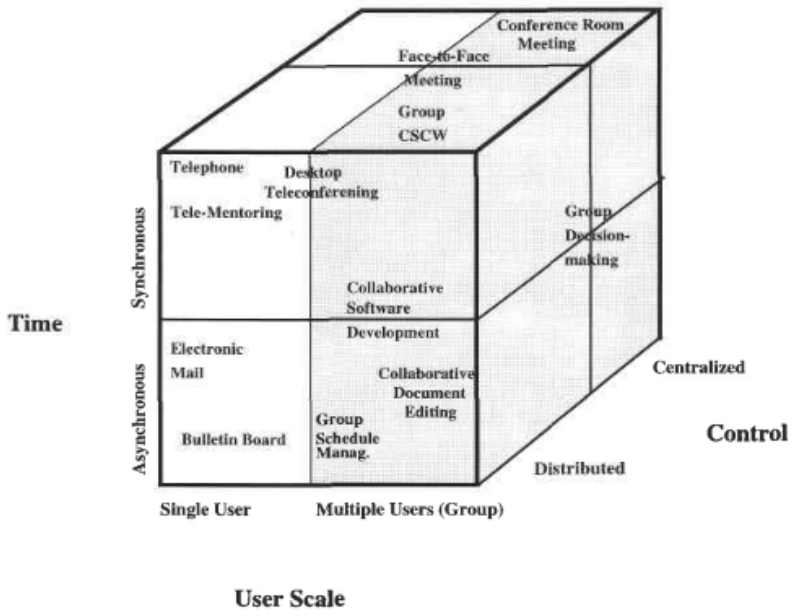

Figure 8.4: Dimensions of collaborative computing.

User Scale The user scale parameter specifies whether a *single user* collaborates with another user or a *group* of more than two users collaborate together. Groups can be further classified as follows:

- A group may be *static* or *dynamic* during its lifetime:
  - A group is *static* if its participating members are predetermined and membership does not change during the activity.
  - A group is *dynamic* if the number of group members varies during the collaborative activity, i. e. , group members can join or leave the activity at any time.
- Group members may have different roles in the Computer-Supported Cooperative Work (CSCW), e. g. a *member* of a group, a *participant* of a group activity, a *conference initiator*, a *conference chairman*, a *token holder* or an *observer*.
- Groups may consist of members which have homogeneous or heterogeneous characteristics and requirements of their collaborative environment.

Control Control during the collaboration can be *centralized* or *distributed*

- *Centralized* control means that there is a chairman (e. g. main manager) who controls the collaborative work and every group member (e. g. user agent) reports to him or her.
- *Distributed* control means that every group member has control over his/her own tasks in the collaborative work and distributed control protocols are in place to provide consistent collaboration.

8.2.1.1.1 Group Communication Architecture Group communication (GC) includes the synchronous or asynchronous communication of several users under central or distributed control.

An architectural model for this communication comprises:

- a support model,
- a system model, and
- an interface model.

The support model includes group *communication agents*, which communicate over a network (see Figure 8.5). These agents include individual components, each dedicated to special aspects:

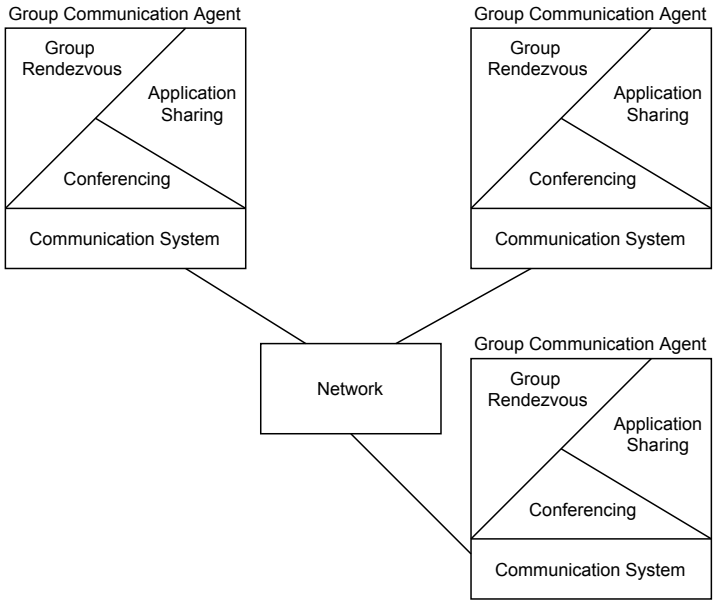

Figure 8.5: Group communication support model.

- **Group Rendezvous** Group rendezvous denotes a method which allows one to organize meetings, and to get information about the group, ongoing meetings and other static and dynamic information.
- **Shared Applications** Application sharing denotes techniques which allow one to replicate information to multiple users simultaneously. The remote users may point to interesting aspects (e. g. via telepointing) of the information and modify it so that all users can immediately see the updated information (e. g. joint editing). Shared applications mostly belong to collaboration-transparent application.
- **Conferencing** Conferencing is a simple form of collaborative computing. This service provides the management of multiple users for communicating with each other using multiple media. Conferencing applications belong to collaboration-aware applications.

The GC system model is based on a client-server model.

- **Clients** provide user interfaces for smooth interaction between group members and the system.
- **Servers** supply functions for accomplishing the group communication work, and each server specializes in its own function.

The GC interface model includes two kinds of protocols for exchanging information within the GC support model:

1. **User presentation protocols** User presentation protocols perform interactions among the clients, such as opening a conference, closing a conference, dynamic joining and leaving of a meeting and floor passing.
2. **Group work management protocols** Group work management protocols specify the communication between the clients and the servers. Services such as registration of active conferences and queries for further conference information are supported by these protocols.

8.2.1.1.1.1 Group Rendezvous Group rendezvous denotes a method which allows one to organize meetings, and to get information about the group, ongoing meetings and other static and dynamic information. There are *synchronous* and *asynchronous* methods for group rendezvous:

- **Synchronous Rendezvous Methods**
  - These methods use directory services and explicit invitations.

  - Directory services access information stored in a knowledge base about the conference, such as the name of the conference, registered participants, authorized users and name and role of the participants.
  - The explicit invitations method sends invitations either point-to-point or point-to-multipoint to conference participants.

- **Asynchronous Rendezvous Methods**
  - These methods may be implemented through e-mail or bulletin boards.
  - The e-mail based mechanism encapsulates in the body message enough information about a group session establishment.
  - Bulletin boards on the internet announces seminars, classes, conferences and other open meetings of a school or institution.

8.2.1.1.1.2 Applications Sharing Sharing applications is recognized as a vital mechanism for supporting group communication activities.

- Sharing applications means that when a shared application program (e. g. editor) executes any input from a participant, all execution results performed in the shared object (e. g. document text) are distributed among all the participants.

An important issue in application sharing is shared control. The primary design decision in sharing applications is to determine whether they should be *centralized* or *replicated*:

- **Centralized Architecture**
  - In a centralized architecture, a single copy of the shared application runs at one site.
  - All participants' input to the application is forwarded to the local site and the application's output (shared object) is then distributed to all sites.
  - The advantage of the centralized approach is *easy maintenance* because there is only one copy of the application that updates the shared object.
  - The disadvantage is *high network traffic* because the output of the application needs to be distributed every time.

- **Replicated Architecture**
  - In a replicated architecture, a copy of the shared application runs locally at each site.
  - Input events to each application are distributed to all sites and each copy of the shared application is executed locally at each site.
  - The Advantages of this architecture are:
    * *Low network traffic*, because only input events are distributed among the sites.
    * *Low response times*, since all participants get their output from local copies of the application.
  - The disadvantages are:
    * The requirement of *the same execution environment* for the application at each site, and
    * The *difficulty in maintaining consistency.*

Figure 8.6 shows centralized architecture and Figure 8.7 shows replicated architecture.

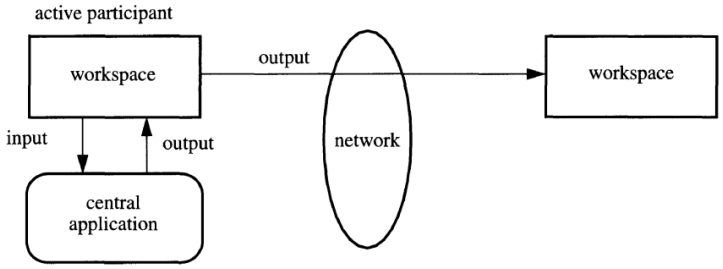

Figure 8.6: Centralized architecture.

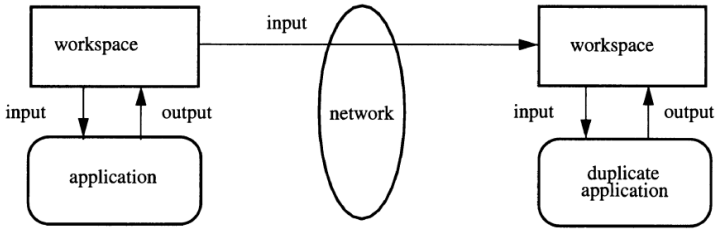

Figure 8.7: Replicated architecture.

The abstraction of a *floor* is used to ensure consistency of distributed data objects (e. g. multimedia documents), or applications (programs) shared among participants. Only one member of the group, namely the one who currently owns the floor, called the *floor holder*, has the right to manipulate distributed objects within the shared workspace. The floor holder is the only user who can create input events for the shared application, i. e. , modify data, which will then be available to all users.

A possible shared data manipulation architecture is shown in Figure 8.8.

- A CSCW control component resides at every-site and dispatches input events coming from an input device (e. g. , keyboard, mouse).
- The control component checks whether the system is currently used by the set of floor holders.
- If this is the case, then the inputs are accepted locally and processed, and then passed on to the other systems.
- If the local system is not a floor holder, then the local inputs are rejected by the control component.
- However, the control component accepts input events transmitted by other participating systems.

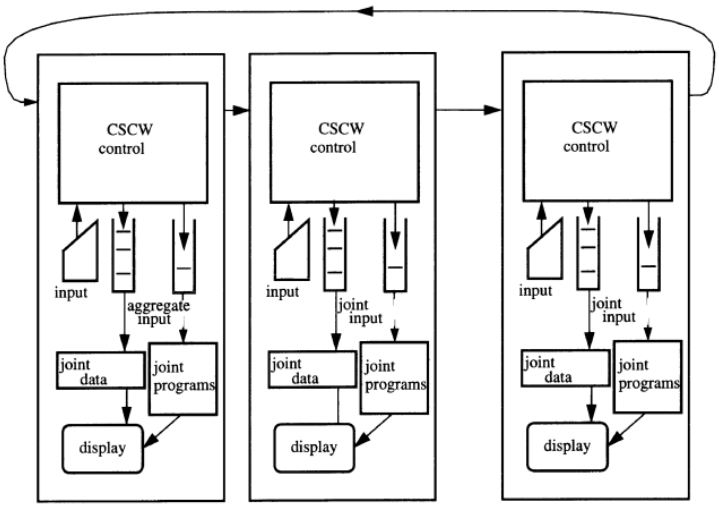

Figure 8.8: Shared data manupulation architecture.

8.2.1.1.1.3 Conferencing Conference supports collaborative computing and is also called synchronous tele-collaboration. Conferencing is a management service that controls the communication among multiple users via multiple media, such as video and audio, to achieve simultaneous face-to-face communication.

More precisely, video and audio have the following purposes in a teleconferencing system:

- *Video* is used in technical discussion to display view-graphs and to indicate how many users are still physically present at a conference. For visual support, workstations, PCs or video walls can be used.
- *Audio* is used for describing and clarifying visual information. Therefore, quality audio, with true full-duplex communication and echo cancellation, and possibly enhanced with spatial queues, is necessary.

Conference control includes several functions:

- *Establishing* a conference, where the conference participants agree upon a common state, such as identity of a chairman (moderator), access rights (floor control) and audio encoding. Conference systems may perform registration, admission, and negotiation services during the conference establishment phase.
- *Closing* a conference.
- *Adding* new users and removing users who leave the conference.

Conference states can be stored (located) either on a central machine (centralized control), where a central application acts as the repository for all information related to the conference, or in a distributed fashion. The control model follows from the location of the conference state. Accordingly, the control model may be either *centralized* or *distributed*.

- **Centralized Conference Control**
  - Centralized conference control means that the conference is set up at one central location.
  - An initiator opens the conference by selecting and explicitly inviting an initial group of participants.
  - This means that the initiator needs to know the addresses of all conference participants.

- **Distributed Conference Control**
  - Distributed conference control is based on a distributed conference state.
  - This state is achieved as follows:
    * The initiator of the conference selects one or several multicast addresses for the transmission of information to the participants, and opens the conference.
    * Conference participants can join by responding to the receipt of special multicast data.
    * The announcement information (multicast address, port) required for this purpose is previously sent or made available to the participants by use of the group rendezvous protocols.

8.2.1.2 Session Management

Session management is the core part which separates the control, needed during the transport, from the actual transport.

8.2.1.2.1 Architecture A session management architecture is built around and *entity-session* manager which separates the control from the transport. By creating a reusable session manager, which is separated from the user-interface, conference-oriented tools avoid a duplication of their effort. A possible session control architecture is shown in Figure 8.9. The session control architecture consists of the following components:

8.2.1.2.1.1 Session Manager Session managers assume *local* and *remote* functions. Local functions include:

1. membership control management, such as participant authentication or presentation of coordinated user interfaces;
2. control management for shared workspace, such as *floor control*;
3. media control management, such as intercommunication among media agents or synchronization;
4. *configuration management*, such as exchange of interrelated QoS parameters or selection of appropriate services according to QoS; and
5. *conference control management*, such as an establishment, modification and a closing of a conference.

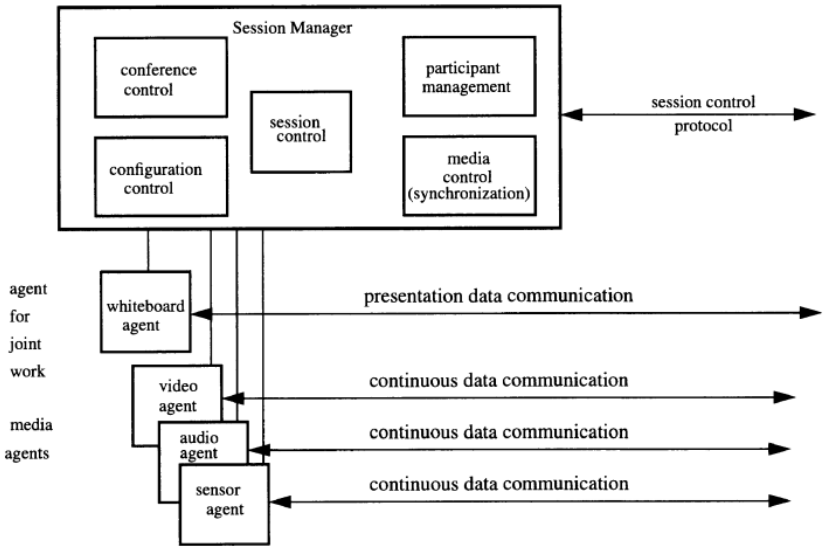

Figure 8.9: Example of session control architecture.

Remote functions include the communication with other session managers to exchange status information, which may include floor information and configuration information.

8.2.1.2.1.2 Media Agents Media agents are separate from the session manager, and they are responsible for decisions specific to each type of media.

- This modularity allows a replacement of agents.
- Each agent performs its own control mechanism over the particular medium, such as mute, unmute, change video quality, start sending, stop sending, etc.

8.2.1.2.1.3 Shared Workspace Agent The shared workspace agent transmits shared objects (e. g. telepointer coordinate, graphical or textual object) among the shared applications.

8.2.1.2.2 Session Control Each session is described by its *session state*. The state information (name of the session, start, valid policies) is either *private* (e. g. , local resources), or *shared* by all participants.

Session management includes two steps to process the session state: an *establishment* and a *modification* of the session. During the establishment,

the session manager negotiates, agrees, and sets the logical state of its own session. It negotiates and defines the transport topology with the transport subsystem.

Mechanisms embedded in session management:

8.2.1.2.2.1 Floor/Activity Control

- Within shared workspaces, the floor control is employed to provide access to the shared workspace.
- Further, the floor control in shared applications is often used to maintain data consistency.
- Floor control, applications use a *floor-passing mechanism* (gavel-passing, chalk-passing).
- The floor-passing mechanism means that at any time, only one participant has the floor.
- The floor is handed off to another participant when requested.
- To obtain the floor, the participant must explicitly take action to signal a floor change

8.2.1.2.2.2 Conference Control For conferencing applications, conference control is employed.

8.2.1.2.2.3 Media Control Media control mainly includes a functionality, such as the synchronization of media stream.

8.2.1.2.2.4 Configuration Control Configuration control includes a control of media quality, QoS handling, resource availability and other system components to provide a session according to users requirements.

8.2.1.2.2.5 Membership Control Membership control may include services, for example, *invitation*, on to a session, *registration* into a session, *modification* of the membership during the session, etc.

8.2.2 Transport Subsystem

8.2.2.1 Requirements

Networked multimedia applications by themselves impose new requirements onto data handling in computing and communications because they need:

1. substantial data throughput,
2. fast data forwarding,
3. service guarantees, and
4. multicasting.

8.2.2.1.1 Data Throughput

- Audio and video data resemble a stream-like behavior, and they demand, high *data throughput*.
- In a workstation or network, several of those streams may exist concurrently, demanding a high throughput.

8.2.2.1.2 Fast Data Forwarding Fast data forwarding imposes a problem on end-systems where different applications exist in the same end-system, and they each require data movement ranging from normal, error-free data transmission to new time-constraint traffic types transmission.

8.2.2.1.3 Service Guarantees Distributed multimedia applications need service guarantees, otherwise their acceptance does not come through as these systems, working with continuous media, compete against analog radio and television services.

8.2.2.1.4 Multicasting Multicast is important for multimedia-distributed applications in terms of sharing resources like the network bandwidth and the communication protocol processing at end-systems.

8.2.2.2 Transport Layer

Transport protocols serve to support addressing of communication end-points within end-systems,

- *fragmentation* and *reassembly* of data,
- *flow* and *congestion control*,*error control*, as well as
- *connection establishment* and *closure*.

Transport protocols, to support multimedia transmission, need to have new features and provide the following functions:

- timing information
- semi-reliability
- multicasting
- NAK (None-AcKnowledgment)-

based *error recovery mechanism* and

- rate control

The Internet protocol stack includes two types of transport protocols:

8.2.2.2.1 Transmission Control Protocol (TCP) Early implementations of video conferencing applications were implemented on top of the TCP protocol.

- TCP provides a reliable, serial communication path, or virtual circuit, between processes exchanging a full-duplex stream of bytes.
- Each process is assumed to reside in an Internet host that is identified by an IP address.
- Each process has a number of logical, full-duplex ports through which it can set up and use full-duplex TCP connections.
- During the data transmission over the TCP connection, TCP must achieve *reliable*, *sequenced delivery* of a stream of bytes by means of an underlying, unreliable IP datagram service.
- To achieve this, TCP makes use of re-transmission on time-outs and positive acknowledgments upon receipt of information.
- Because re-transmission can cause both out-of-order arrival and duplication of data, sequence numbering is crucial.

Flow control in TCP makes use of a window technique in which the receiving side of the connection reports to the sending side the sequence numbers it may transmit at any time and those it has received contiguously thus far.

For multimedia, the *positive acknowledgment* causes substantial overhead as all packets are sent with a fixed rate. *Negative acknowledgment* would be a better strategy. Further, TCP is not suitable for real-time video and audio transmission because its re-transmission mechanism may cause a violation of deadlines which disrupt the continuity of the continuous media streams. TCP was designed as a transport protocol suitable for non-real-time reliable applications, such as file transfer, where it performs the best.

8.2.2.2.2 User Datagram Protocol (UDP) UDP is a simple extension to the Internet network protocol IP that supports multiplexing of datagrams exchanged between pairs of Internet hosts. It offers only *multiplexing* and *checksums*, nothing else. Higher-level protocols using UDP must provide their own retransmission, packetization, reassembly, flow control, congestion avoidance, etc.

Many multimedia applications use this protocol because it provides some degree of real-time transport property, although loss of PDUs may occur. For experimental purposes, UDP above IP can be used as a simple, unreliable connection for medium transport.

In general, UDP is not suitable for continuous media streams because it does not provide the notion of connections, at least at the transport layer; therefore, different service guarantees cannot be provided.

8.2.2.2.3 Real-time Transport Protocol (RTP) RTP is an end-to-end protocol providing network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data over multicast or unicast network services. RTP is primarily designed to satisfy the needs of multi-party multimedia conferences, but it is not limited to that particular application. RTP provides functions, such as:

- determination of media encoding
- synchronization, framing
- error detection
- encryption
- timing and
- source identification

8.2.2.2.4 Xpress Transport Protocol (XTP) XTP was designed to be an efficient protocol, taking into account the low error ratios and higher speeds of current networks. XTP integrates transport and network protocol functionalities to have more control over the environment in which it operates.

It defines six service types:

1. connection
2. transaction
3. unacknowledged datagram
4. acknowledged datagram
5. isochronous stream and
6. bulk data

8.2.2.3 Network Layer

The requirements on the network layer for multimedia transmission are a provision of *high bandwidth*, *multicasting*, *resource reservation* and *QoS guarantees*, *new routing protocols* with support for streaming capabilities and new *higher-capacity routers* with support of integrated services.

IP

8.2.2.3.1 Internet Protocol Version 4 (IPv4) IP provides for the unreliable transfer of datagrams from a source host to destination hosts, possibly passing through one or more gateways (routers) and networks in the process. Following are the IP properties:

1. *Types of Service*
2. *Addressing and Multicasting*
3. *Interconnectivity Between Internet Protocol and Underlying Networks*
4. *Routing*

8.2.2.3.1.1 Types of Service IP includes identification of the service quality through the Type of Service (TOS) specification. TOS specifies:

1. precedence relation and
2. services such as *minimize delay*, *maximize throughput*, *maximize reliability*, *minimize monetary cost* and *normal service*.

8.2.2.3.1.2 Addressing and Multicasting One of the most critical functions of the IP is the *addressing*, i. e. , to establish a global address space that allows every network on the Internet to be uniquely identified. The IP addressing structure is shown in Figure 8.10.

The network addressing structure was revised to accommodate five classes of address: A, B, C, D and E

- Class A retained the $24bits$ host identifier field, but only $7bits$ for network number. This address space covers a small number of class A networks.
- Class B with $16bits$ for the host identifier and $14bits$ for the network number allows a larger number of Class B networks.

- Class C allocates $21bits$ for network number and $8bits$ for host identifier, therefore many more Class C networks are available.
- Class D addresses, are used for *multicasting*.
- Class E addresses have been reserved for future extensions.

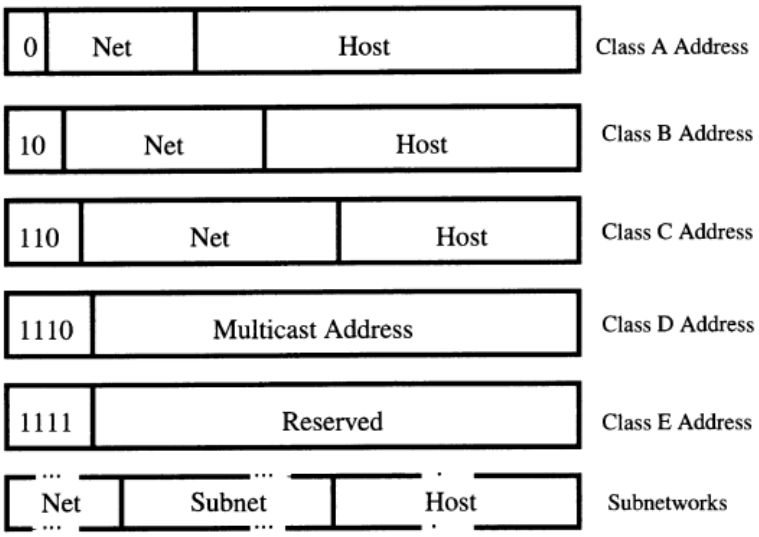

Figure 8.10: IPv4 addressing structure.

8.2.2.3.1.3 Interconnectivity Between Internet Protocol and Underlying Networks The Internet family of protocols is one of today's most widespread protocol stacks in computer networking. There is a strong interest in transporting the IP datagrams, which may carry multimedia traffic, over different networks, for example, Ethernet, ATM B-ISDN, MPLS, or `IEEE 802.11` wireless networks. Hence, the mapping between the Internet protocol and the underlying layers is of importance. Another important function in this task is the binding of IP addresses to lower-level network addresses.

8.2.2.3.1.4 Routing A major subject in Internet architecture is the routing of IP packets because the basic model of Internet consists of networks connected by routers. To create an opportunity for further experimental exploration of different routing protocols for global networking, the concept of *Autonomous Systems* (AS) was developed. ASs are collections of routers

falling under a common administrative authority. In theory, the routers commonly use the same routing protocol- Interior Gateway Protocol (IGP), within the AS. AS of gateways (routers) exchange reachability information by means of an Exterior Gateway Protocol (EGP).

As the common IGP for the Internet, the *Open Shortest Path First* (OSPF) has been adopted.

Another IGP protocol is the *Routing Information Protocol* (RIP). RIP was one of the first routing protocols used with IP and was implemented by the program routed that comes with most UNIX systems.

RIP uses a *distance vector* algorithm to propagate routing information. A router running RIP advertises the destinations it can reach along with a distance to each destination; adjacent routers receive the information and update heir routing tables.

For EGP, the *Border Gateway Protocol* (BGP) was developed. BGP allows the sender and receiver to enforce policies, to provide facilities for transit routing, and it uses TCP for all communication to ensure reliable transport.

8.2.2.3.2 Internet Protocol Version 6 (IPv6)

- One of the major reasons why a new version of IP protocol is considered is the *limited address space* of IPv4.
- Second major motivation for changes in IP have arisen from new Internet applications such as audio and video applications.
- Third motivation is that IPv4 is fully missing security functions.

IPv6 retains many of the design features that have made IPv4 successful. Hence, IPv6 is also connectionless protocol, where each datagram contains a destination address, and each datagram is routed independently. Despite retaining the basic concepts from the current version, IPv6 changes can be categorized as follows:

1. enhanced addressing and improved routing,
2. simplification of IP header format,
3. improved support of IP options,
4. support of QoS,
5. support of security, and

6. fragmentation of data.

- IPv6 utilizes 128-bit Internet addresses. Therefore, it can support 2^{128} Internet addresses.
- IPv6 is represented by 8 sets of 4 hexadecimal digits separated by colon. Example: `2001:db8:85a3:0000:0000:8a2e:370:7334`
- For example: the above IPv6 address can be represented as `2001:db8:85a3::8a2e:370:7334`

  Here are 6 sets of hexadecimal 4 digits number so :: represents two sets of `0000` within it.

8.2.2.3.3 Internet Group Management Protocol (IGMP) IGMP is a protocol for managing Internet multicasting groups. It is used by conferencing applications to join and leave particular multicast group. The basic service permits a source to send datagrams to all members of a multicast group. There are no guarantees of the delivery to any or all targets in the group.

A multicast router periodically sends queries (Host Membership Query messages) with a Time-to-Live (TTL) value of 1 (in order not to leave the connected LAN segment) to refresh their knowledge of memberships present on a particular network. If there exist multiple multicast routers on that network segment, one of them will be responsible to respond to the query

8.2.2.3.4 Resource reServation Protocol (RSPV) RSVP is a protocol which transfers reservations and keeps a state at the intermediate nodes. It does not have a data transfer component. RSVP messages are sent as IP datagrams, and the router keeps "soft state", which is refreshed by periodic reservation messages. In the absence of the refresh messages, the routers delete the reservation after a certain timeout.

This protocol was specified by IETF to provide one of the components for integrated services on the Internet. To implement integrated services, four components need to be implemented:

1. the packet scheduler,
2. admission control routine,

3. classifier, and

4. the reservation setup protocol.

RSVP reservations[1] are receiver-oriented, which means that the sender starts, but the actual reservation of resources is performed by the receiver. This is done to support heterogeneous receivers in a multicast group.

8.2.3 Quality of Service and Resource Management

Quality of service is the ability to provide different priorities to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow.

The user/application requirements on multimedia systems need to be mapped into services which then make the effort to satisfy the requirements. Due to the heterogeneity of requirements, services in multimedia systems must be parameterized. The result is the *"quality-controllable services"* which then allow to classify and differentiate system and communication services.

Parameterization of services was defined first in ISO (International Standard Organization) standards through the notion of Quality of Service (QoS). The ISO standard defined QoS as a concept for specifying how *"good"* the offered networking services are.

Quality of Service indicates the defined and controlling behavior of a service expressed through quantitative measurable parameter(s).

QoS Layering

Traditional QoS (ISO standards) was defined by the network layer of the communication system. An enhancement of QoS was achieved through inducing QoS into transport services. For Multimedia Communication System (MCS), the QoS notion must be extended because many other services contribute to the end-to-end service quality.

The MCS consists of three layers:

- *application*,
- *system* (including communication services and operating system services), and
- *devices* (network and Multimedia (MM) devices).

[1] A reservation specifies the amount of resources to be reserved for all, or some subset of the packets in a particular session.

Above the application may or may not reside a human user. This implies the introduction of QoS in the application (application QoS), in the system (system QoS) and in the network (network QoS). In the case of having a human user, the MCS may also have a user QoS specification. We concentrate in the network layer on the network device and its QoS because it is of interest to us in the MCS. The MM devices find their representation (partially) in application QoS.

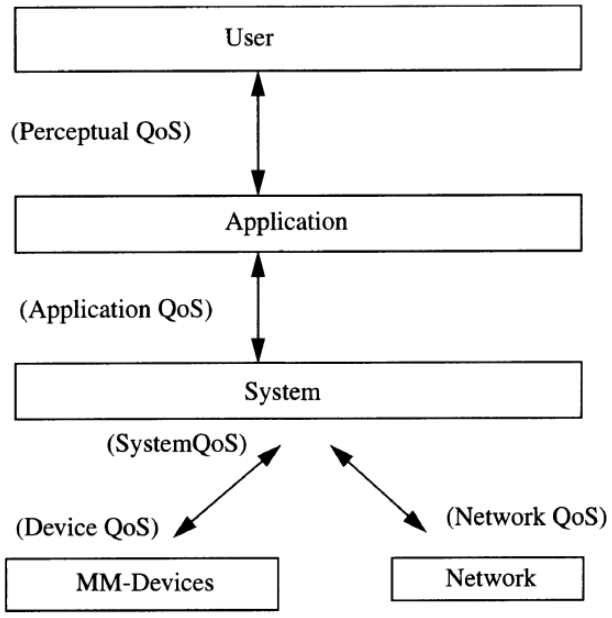

Figure 8.11: QoS-layered model for the MCS.

Perceptual/User QoS The *perceptual QoS* parameters need to allow for description of two major requirements:

- description of **perceptive qualities** such as media quality (e. g. , excellent, good, or bad quality), windows size (e. g. , big, small), response time (e. g. , interactive, batch), security (e. g. , high, low), and
- description of **pricing choices**, i. e. users should be able to specify the range of price they are willing to pay for the desired service.

Application QoS Application QoS parameters describe requirements for application services in terms of:

- **media quality**, including media characteristics (e. g. , frame rate, frame resolution), and their transmission characteristics (e. g. , end-to-end delay, jitter);

- **media relations**, specifying relations among media (e.g., media transformation, inter and intra frame synchronization skews); and

- **adaptation rule**s (e. g. , actions if network bandwidth is scare).

System QoS System QoS parameters describe requirements, placed on communication and computing services, derived from application QoS. They may be specified in terms of *quantitative* and *qualitative* criteria.

- **Quantitative criteria** represent concrete measures such as *bit per second*, *number of errors*, *task processing time*, *task period*.

  - The QoS parameters include then *throughput*, *delay*, *response time*, *data rate*, *data corruption* at the system level, task and *buffer specifications*.

- **Qualitative criteria** specify the expected functions needed for provision of QoS such as *interstream synchronization*, *ordered data delivery*, *error-recovery mechanisms*, *scheduling mechanisms* and others.

Network QoS Network QoS parameters describe requirements, placed on low level network services. They may be specified in terms of:

- **network load**, describing the ongoing network traffic and characterized through average/minimal interarrival time on the network connection, burstiness, packet/cell size and service time in the node for a connection's packet/cell and

- **network performance**, describing network service guarantees. Performance might be expressed through a source to destination delay bound for a packet, or packet loss rate, or others.

Device QoS Device QoS parameters typically specify timing and throughput demands for media data units given by audio/video devices.

QoS Parameter Values and Types of Service

The specification of QoS parameter values determines the type of service, called *service class*. There are at least three service classes:

1. guaranteed,
2. predictive and best
3. effort services.

Guaranteed Services Guaranteed services provide QoS guarantees, as specified through the QoS parameter values either in *deterministic* or *statistical* representation.

- The *deterministic* values can be given through a single value (e. g. average value, threshold value, target value), a pair of values (e. g. minimum and average value, lowest quality and target quality) or an interval of values (lower bound and upper bound).
- *Statistical* value specifies statistical bound on error rate etc.

Predictable Services A predictive service is based on past network behavior, hence the QoS parameters are estimates of past behavior which the service tries to match.

Best Effort Services Best-effort services are based on either no guarantees, or on partial guarantees. There is either no specification of QoS parameters required, or some bounds in deterministic or statistical forms are given.

8.2.3.1 Resource Management Architecture

Resources are managed by various components of a resource management subsystem in a networked multimedia system (Figure 8.12). The main goal of resource management is to provide guaranteed delivery of multimedia data. This goal implies three main actions:

1. to *reserve* and allocate resources (end-to-end) during multimedia call establishment so that traffic can flow according to the QoS specification, which means distribution and negotiation of the QoS specification for system components involved in the data transfer from the source(s) to the sink(s);

2. to *provide* resources according to the QoS specification, which means adhering to resource allocation during multimedia delivery using proper service disciplines; and

3. to *adapt* to resource changes during ongoing multimedia data processing.

The resource management subsystem includes *resource managers* at the hosts as well as the network nodes. *Resource management protocols* are used to exchange information about resources among the resource management.

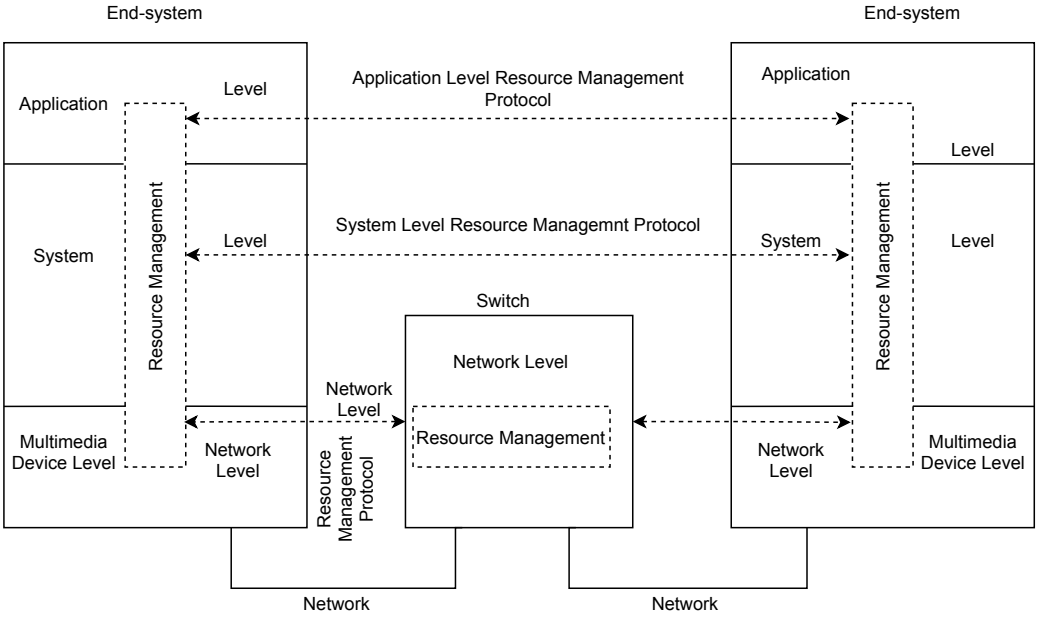

Figure 8.12: Resource management in MCSs.

Relation Between QoS and Resources

The requested output QoS parameters are dependent on:

- the *input quality*,
- the *resource capacity* allocated to services (processes), and
- the *scheduling algorithms* managing the shared resources for the distributed multimedia system.

According to the requested output QoS parameters, we can determine how much resources are required to achieve it.

- For example, the requested end-to-end delay parameter determines the behavior of transmission services along the path between media source and sink with respect to:
- packet scheduling (bandwidth allocation),
- queueing (buffer allocation) and
- task scheduling (CPU allocation).

The above described relation between QoS and resources can be expressed in the form of different mappings, service curves, and profiles. The relation between QoS and resources is established in two phases:

1. *Establishment Phase* (Setup) and
2. *Runtime Phase* (Enforcement)

Establishment Phase As shown in Figure 8.13, resources are reserved (planned) and allocated during the connection setup according to the QoS specification.

1. This means that during the establishment, an application client requests a resource allocation by specifying its requirements through an application QoS specification.
2. This QoS specification is then translated by a QoS translator entity (e. g. , QoS compiler) into the system QoS specification and their required resources.
3. The required resources represent then the reservation request to the resource management.
4. The resource management checks the resource utilization and decides if the reservation request can be served.
5. If the reservation can be granted along the end-to-end path, a QoS contract is provided to the application/user.
6. If reservation cannot be granted, the request is rejected.

Runtime Phase As shown in Figure 8.14, resources must be provided according to QoS specifications during the lifetime of an application.

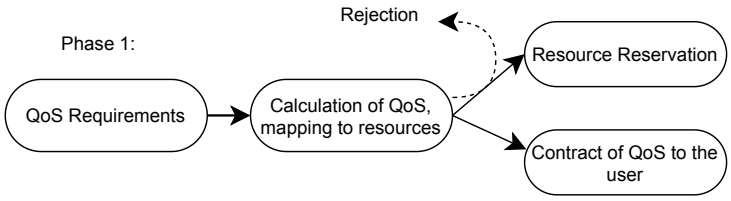

Figure 8.13: Establishment phase.

1. This means, that once the resource reservation and allocation are performed during the establishment phase, the QoS enforcement must occur during the runtime phase.
2. The resource allocation must be enforced through mechanisms such as traffic shaping and appropriate scheduling mechanisms.
3. If any changes in resource allocation occur, an adaptation needs to be in place to adjust the data transfer if necessary.

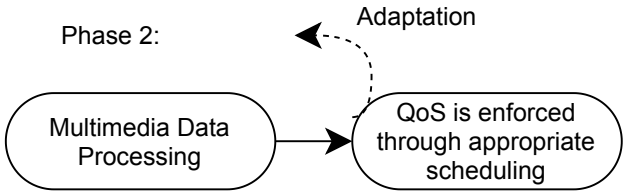

Figure 8.14: Runtime phase.

8.3 Abstraction of Programming

8.3.1 Abstraction Levels

Abstraction levels in programming define different approaches with a varying degree of detail for representing, accessing and manipulating data. A multimedia application may access each level.

A *device* for processing continuous media can exist as a separate component in a computer. In this case, a device is not part of the operating system, but is directly accessible to every component and application.

A *library*, the simplest abstraction level, includes the necessary functions for controlling the corresponding hardware with specific device ac-

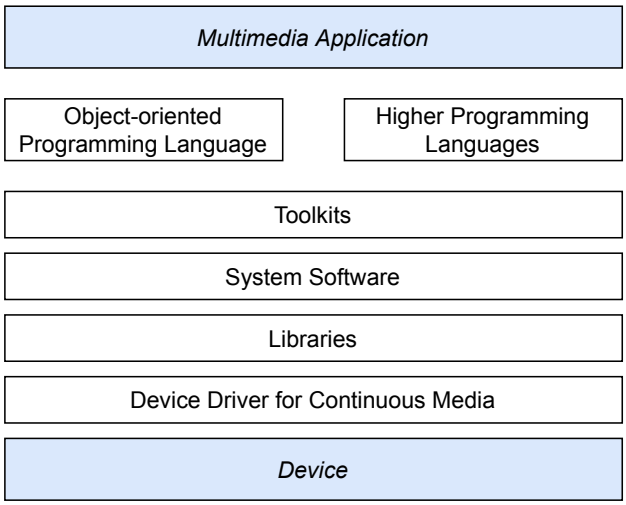

Figure 8.15: Abstraction levels of the programming of multimedia systems.

cess operations.

As with any device, multimedia devices can be bound through a device driver, respectively the operating system. Hence, the processing of the continuous data becomes part of the *system software*.

Multimedia *device drivers* embedded in operating systems simplify considerably the implementation of device access and scheduling.

Higher procedural programming languages build the next abstraction level. They are the languages most often used to implement commercial multimedia applications. Further, they can contain abstractions of multimedia data. The code generated from the compiler can be processed through libraries, as well as through a system interface for continuous data.

More flexibility for the programmer is provided via the abstraction level - *an object-oriented environment*. This environment provides the application with a class hierarchy for the manipulation of multimedia. Also in this case, the generated or interpreted code can be processed and controlled through libraries, as well as through a system interface for continuous media (see Figure 8.15).

8.3.2 Libraries

- The processing of continuous media is based on a set of functions which are embedded into libraries.
- This is the usual solution for programming multimedia data.
- These libraries are provided together with the corresponding hardware.

- Some libraries can be considered as extensions of the graphical user interface, whereas other libraries consist of control instructions passed as control blocks to the corresponding driver.

8.3.3 System Software

Instead of implementing access to multimedia devices through individual libraries, the device access can become part of the operating system. An example of access to multimedia devices and support for continuous media processing implemented in operating system is the experimental *Nemo* system from the University of Cambridge.

8.3.4 Toolkits

A simpler approach for control of the audio and video data processing can be taken by using *toolkits*. These toolkits are used to:

- Abstract from the actual physical layer.
- Allow a uniform interface for communication with all different devices of continuous media.
- Introduce the client-server paradigm.

Toolkits can also hide process-structures. It is possible to embed toolkits into the programming languages or object-oriented environment.

8.3.5 Higher Programming Languages

- In HLL, the processing of continuous media data is influenced by a group of similar constructed functions.
- These calls are mostly hardware and driver-independent.
- Hence, their integration in HLLs leads to a wishful abstraction, supports a better programming style and increases the productivity.
- The programs in an HLL either directly access multimedia data structures, or communicate directly with the active processes in a real-time environment.
- The processing devices are controlled through corresponding device drivers.
- Compiler, linker and/or loader

provide the required communication between the application program and the processing of continuous data.

Media can be considered differently inside a programming language.

8.3.5.1 Media as Types

- One of the alternatives to programming in an HLL with libraries is the concept of *media as types*.
- Here, the data types for video and audio are defined.
- In the case of text, character is the type (the smallest addressable element).
- A program can address such characters through functions and sometimes directly through operators.
- They can be copied, compared with other characters, deleted, created, read from a file or stored.
- Further, they can be displayed, be part of other data structures, etc.

.

8.3.5.2 Media as Files

Another possibility of programming continuous media data is the consideration of continuous media streams as *files instead* of data types.

```
file_h1 = open(MICROPHONE_1,...)
file_h2 = open(MICROPHONE_2,...)
file_h3 = open(SPEAKER, ...)
...
read(file_h1)
read(file_h2)
mix(file_h3, file_h1, file_h2)
activate(file_h1, file_h2, file_h3)
...
deactivate(file_h1, file_h2, file_h3)
...
rc1 = close(file_h1)
rc2 = close(file_h2)
rc3 = close(file_h3)
```

The example describes the merging of two audio streams. The physical file is associated during the open process of a file with a corresponding file name. The program receives a *file descriptor* through which the file is accessed. In this case, a device unit, which creates or processes continuous data streams, can be associated with a file name.

Read and *write functions* are based on continuous data stream behavior. Therefore, a new value is assigned continuously to a specific variable which is connected, for example, with one read function. On the other hand, the read and write functions of discrete data occur in separate steps. For each assignment of a new value from a file to the corresponding variable, the read function is called again.

An *activate function* means that the actual data transmission starts and a deactivate function means that the transmission stops.

8.3.5.3 Media as Processes

The processing of continuous data contains a time-dependency because the life span of a process equals to the life span of a connection(s) between source(s) and destination(s). A connection can exist locally, as well as remotely.

```
1  PROCESS cont_process_a;
2    ...
3    On_message_do
4      set_volume ...
5      set_loudness ...
6      ...
7    ...
8    [main]
9  pid = create (cont_process_a)
10 send(pid, set_volume, 3)
11 send(pid, set_loudness)
12 ...
```

In the above example, the process `cont_process_a` implements a set of *actions* (*functions*) which apply to a continuous data stream, Two of them are the modification of the volume `set_volume` and the process of setting a volume, dependent from a band filter, `set_loudness`.

8.3.5.4 Programming Language Requirements

The processing of continuous data is:

- Controlled by the HLL through pure asynchronous instructions.
- An integral part of a program through the identification of the media, respectively data streams with data types, variables, files or processes.

Therefore, the HLL should support a *parallel processing*.

8.3.6 Object-oriented Approaches

- The object-oriented approach was first introduced as a method for the reduction of complexity in the software development and it is used mainly with this goal today.
- Further, the reuse of software components is a main advantage of this paradigm.
- The basic ideas of object-oriented programming are: *data encapsulation* and *inheritance*, in connection with *class* and *object* definitions.
- Instead of using functions and data structures, programs are implemented by using classes, objects, and methods.

8.3.6.1 Abstract Type Definition

- The definition of data types through abstract interfaces is called *abstract type definitions*.
- The abstract type definition is understood as an interface specification without a knowledge and implementation of internal algorithms.
- This data abstraction hides the used algorithm.
- In a distributed multimedia system, abstract data types are assumed for virtual and real device units such as cameras and monitors.

8.3.6.2 Class

- The implementation of abstract data types is done through classes.
- A class specification includes an interface provided to the outside world.
- For example, in a class *professional camera*, the operations *zoom* and *set back-light* are defined and implemented.
- If the objects, which represent a closed class, use only relative position entries, the implementation of the *zoom* operation needs to transform the absolute values into the necessary relative parameters.

8.3.6.3 Object

- An *object* is the instance of the class.
- Therefore, all objects, derived from the same class include the same operations as an interface to the outside world.
- An object is created at run-time of the system.
- It includes a set of operations,

which are called *methods*.

- Additionally, each object has an *internal state*, which exists during the life span of the object, but it can only be accessed using the methods associated with this object.
- It can be compared with a global variable assigned to a process, but not with local variables of functions and procedures.
- Objects communicate among each other through the exchange of messages.
- Thus, a *message* calls the corresponding method of the target object.

In a distributed multimedia environment, virtual units are considered to be objects. Multimedia data units (the LDU's images, audio and video clips) can also be considered objects.

8.3.6.4 Inheritance

One of the most important properties of object-oriented systems is inheritance. Classes contain, besides the root and leaves of the hierarchy, super classes and subclasses (parent and child).

- For example, let the class *professional-camera* be a subclass of the class *camera*.
- Methods such as *autofocus-on* and *focus* are defined in the class *camera*.
- The *professional-camera* class also has the method *zoom*.
- An object, which is derived from the *professional-camera*, can use the method *zoom*, as well as the operations *focus* and, *autofocus-on*.

The main problem has been and remains to be the design of a clear and uniform class hierarchy for a multimedia system.

8.3.6.5 Polymorphism

Polymorphism is related to the property of inheritance indicating when the same name of a method is defined in several classes (and objects) with different implementations and functionalities.

- For example, the function play is used with audio and video data.
- It uses different device units for each medium.

- The data can come either from a file of a local file environment or from an audio-video sequence of an external device.
- Inside the object-oriented approach, for example, play is defined in different classes.
- According to which object must perform the operation, the corresponding method is chosen.

The complexity of different types and device units is reduced and there is a common set of method names for classes and objects of different media. On the other hand, polymorphism can also very easily cause programming errors that are difficult to find. Hence, this abstraction strongly complicates the implementation. This can occur easily through unwanted, multiple identical method names.

8.4 Synchronization

8.4.1 Introduction

Synchronization is the coordination of the events to operate a system in unison. Systems operating with all their parts in synchrony are said to be synchronous or in sync. Synchronization in multimedia systems refers to the temporal relations between media objects in the multimedia system. Synchronization between media objects comprises relations between time-dependent media objects and time-independent media objects. It is addressed and supported by many system components including the operating system, communication system, databases, and documents and even often by applications.

8.4.2 Notion of Synchronization

The Merriam-Webster Dictionary defines the term *synchronization* as follows:

Synchronization = the act or result of synchronizing; the state of being synchronous.

Synchronize = to happen at the same time; to represent or arrange (events) to indicate coincidence or coexistence; to make synchronous in operation; to make (motion picture sound) exactly simultaneous with the action.

Synchronization creates a relationship between independent objects (pieces of information, media, processes, data streams, LDUs). Synchronization between media objects includes relationships between time-dependent and time-independent objects.

Three criteria for the classification of a system as a multimedia system can be distinguished:

- the *number of media*
- the *types of supported media* and
- the *degree of media integration*

The Simplest criterion is the number of media used in an application. Using only this criterion, even a document processing application that supports text and graphics can be regarded as a multimedia system.

The types of supported media are an additional criterion. In this case we distinguish between time-dependent and time-independent media.

- A *time-independent* media object is usually presented using one presentation unit (e. g. bitmap graphics).
- *Time-dependent* media objects are presented by a sequence of presentation units (e. g. a video sequence) presented frame after frame.

The degree of media integration is the third criterion. In this case, integration means that the different types of media remain independent but can be processed and presented together.

Combining all three criteria, we propose the following definition of a multimedia system: *a system or application that supports the integrated processing of several media types with at least one time-dependent medium.*

Figure 8.16 classifies applications according to the three criteria. The arrows indicate the increasing degree of multimedia capability for each criterion.

Integral digital systems can support all types of media, and due to digital processing, may provide a high degree of media integration. Systems that handle time-dependent analog media objects and time-independent digital media objects are called *hybrid systems*. The disadvantages of hybrid systems is that they are restricted with regard to the integration of time-dependent and time-independent media, for example, audio and video are stored on different devices than time-independent media objects.

The audio/video applications are time-dependent media objects. In addition, they often do not support the separate handling of audio and video media objects. They are supported by audio and video server.

Traditional desktop-publishing systems are examples of integrated processing of time-independent media objects supported by the text editors.

Basic Synchronization Issues

The word synchronization refers to time. In a more general and wide sense, we use synchronization in multimedia systems as comprising:

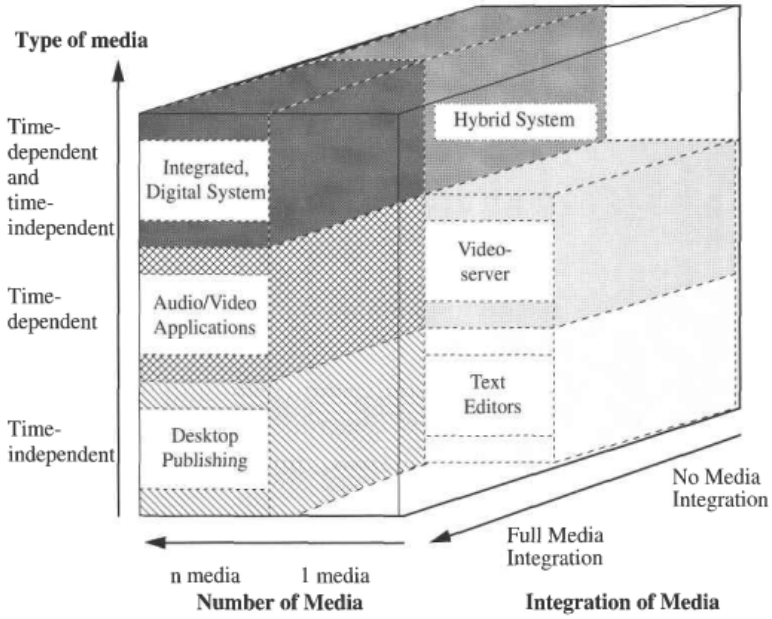

Figure 8.16: Classification of media use in multimedia systems.

- *content*
- *spatial* and
- *temporal* relations between media objects

Content Relations

- Content relations define a dependency of media objects from some data.
- Example: dependency between spreadsheet and graphics that represent data listed in spreadsheet. In this case, the same data are represented in two different ways.
- Another example is two graphics that are based on the same data but different interpretations of the data.
- In general, the implementation of content relations in multimedia systems is based on the use of common data structures or object interfaces that are used to present objects using different media.

Spatial Relations

- The spatial relations that are usually known as *layout relationship* define the space used for the presentation of a media object on an output device at a certain point of time in a multimedia presentation.
- In desktop-publishing applications, this is usually expressed using *layout frames*.
- A layout frame is placed and a content is assigned to this frame.
- The positioning of a layout frame in a document may be fixed to a position in a document, to a position on a page or it may be relative to the positioning of other frames.

Temporal Relations

- Temporal relations define the temporal (time) dependencies between media objects.
- An example of temporal relations is the relation between a video and an audio object that are recorded during a concert.
- If these objects are presented, the temporal relation during the presentations of the two media objects must correspond to the temporal relation at the recording moment.

These time relations are synchronization in multimedia systems.

8.4.2.1 Intra- and Inter-object Synchronization

Intra-object Synchronization

- Intra-object synchronization refers to the time relation between various presentation units of one time-dependent media object.
- An example is the time relation between the single frames of a video sequence.
- For a video with a rate of $25 frames$ per second, each of the frames must be displayed for $40ms$.
- Figure 8.17 shows this for a video sequence presenting a bouncing ball.

Inter-object Synchronization

- Inter-object synchronization refers to the synchronization between media objects.

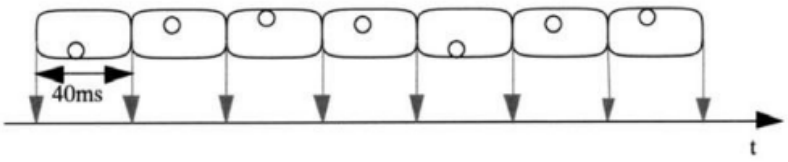

Figure 8.17: Video sequence showing a bouncing ball.

- Figure 8.18 shows an example of the time relations of a multimedia synchronization that starts with an audio/video sequence, followed by several pictures and an animation that is commented by an audio sequence.

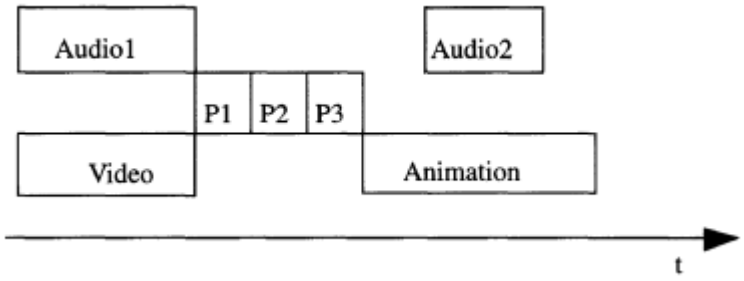

Figure 8.18: Inter-object synchronization of images, one animation, and audiovisual sequences.

8.4.3 Presentation Requirements

- The correct transmission of multimedia data at the user interface requires synchronization.
- It is impossible to state an objective measure for the synchronization from the view of the subjective human perception.
- As the human perception differs from one person to another, only heuristic criteria can be defined whether a presentation stream is correct.

Requirements to the presentation include the accuracy with regard to the presentation of LDUs for intra-object synchronization, and the accuracy with

regard to the parallelism of the presentation of media objects for inter-object synchronization.

In intra-object synchronization, we should attempt to avoid jitter or variance in an two consecutive LDUs.

Live and Synthetic Synchronization

The live and synthetic synchronization distinction refers to the type of the determination of temporal relations. In the case of live synchronization, the goal of the synchronization is to exactly reproduce at a presentation the temporal relations as they existed during the capturing process. In the case of synthetic synchronization, the temporal relations are artificially specified.

Live Synchronization

- A typical application of live synchronization is conversational services.
- In the scope of a source/sink scenario, at the source, volatile data streams (i. e. , data being captured from the environment) are created which are presented at the sink.
- The common context of several streams on the source site must be preserved at the sink.
- The source may be comprised of acoustic and optical sensors, as well as media conversion units.
- The connection offers a data path between source and sink.
- The sink presents the units to the user.
- A source and sink may be located at different sites.
- The goal of synchronization in such a scenario is to reproduce at the sink the signals in the same way as they appeared at the source.
- A possible manipulation by the sink is to adapt the presentation to the available resources.
- This may be, for example, a change of resolution or a lower frame rate.

Synthetic Synchronization

The emphasis of synthetic synchronization is to support flexible synchronization relations between media. In synthetic synchronization, two phases can be distinguished:

- In the specification phase, temporal relations between the media objects are defined.
- In the presentation phase, a run-time system presents data in a synchronized mode.

8.4.4 Reference Model for Multimedia Synchronization

A four-layer synchronization reference model is shown in Figure 8.19.

- A reference model helps to understand the large number of requirements to multimedia synchronization.
- Each layer implements synchronization mechanisms which are provided by an appropriate interface.
- These interfaces can be used to specify and/or enforce the temporal relationships.
- Each interface defines services, i. e. , offering the user a means to define his/her requirements.
- Each interface can be used by an application directly, or by the next higher layer to implement an interface.
- Higher layers offer higher programming and Quality of Service (QoS) abstractions.

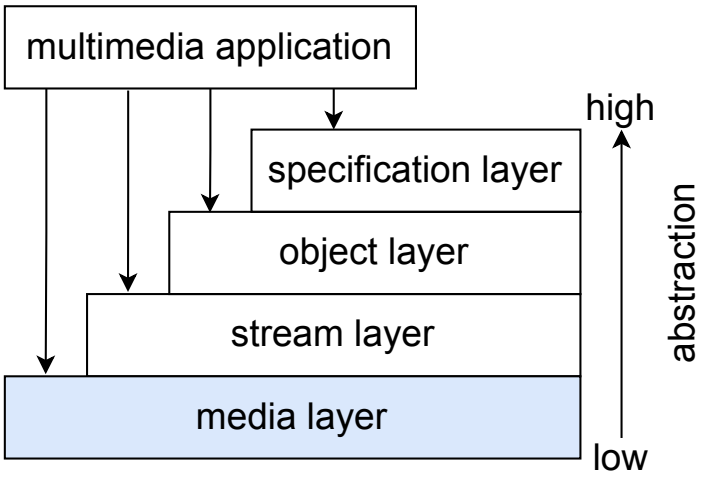

Figure 8.19: A four-layer reference model.

8.4.4.1 Media Layer

- On the media layer, an application handles one single continuous media stream as an LDU sequence.
- The abstraction offered by this layer is a device-independent interface with operations like `read (device-handle, LOU)` and `write (device-handle, LOU)`.
- To build a continuous media stream that uses the abstractions supplied by the media layer, an application runs one process for each stream in the form shown in the following example:

```
window = open("Videodevice"); \\ Create a video output window
movie = open("File"); \\ Open the video file
while (not eof(movie)) { \\ Loop
  read (movie, &ldu); \\ Read LOU
  if (ldu.time == 20) \\ Start the presentation
    print("Subtitle 1"); \\ of the synchronized subtitles
  else if (ldu.time == 26)
    print ("Subtitle 2");
  write (window, ldu); \\ Present LOU
}
close(window);\\ Close window
close(movie); \\ Close file
```

8.4.4.2 Stream Layer

- The stream layer deals with continuous media streams and media stream groups.
- A group presents all streams in parallel by using mechanisms for inter-stream synchronization.
- The abstraction supplied by the stream layer is called *streams with time parameters*.
- The quality of service relates to parameters for inter-stream synchronization within a stream and inter-stream synchronization between the streams of the group.
- Continuous media in the stream layer are referred to as a data flow.

8.4.4.3 Object Layer

- The object layer deals with all kinds of media and hides the differences between discrete and continuous media from the user.

- The abstraction given by an application is that of a complete and synchronized presentation.
- This layer accepts a synchronization specification as its input and is responsible for the correct scheduling (time plan) of the entire presentation.
- The object layer offers functions to calculate and execute complete presentation sequence schedules, including the presentation of non-continuous media objects.
- The object layer initiates preparatory actions required to achieve a correctly synchronized presentation.
- One example for the integration of this layer is the MHEG specification.

The objective of the MHEG standards is to encode multimedia and hypermedia information objects for presentation. Our example below uses a simple notation to demonstrate the basics of this reference model.

```
Composite { \\ Composite object
    start-up link \\ How to start the presentation
    viewer start-up
    viewer-list \\ Virtual views on
    Viewerl: reference to Componentl \\ component objects
    Viewer2: reference to Component2
    Viewer3: reference to Component3
    Componentl \\ Component objects
      reference to content "movie.avs" \\ of the composite
    Component2
      reference to content "subtitle1"
    Component3
      reference to content "Subtitle2"
    Link1 \\ Temporal relations
      "when timestone status of Viewer1 becomes 20 then start
   Viewer2"
    Link2
      "when timestone status of Viewer1 becomes 26 then start
   Viewer3"
}
```

One possible implementation of the object layer is the MHEG runtime system called *MHEG Engine*. This engine evaluates the status of the objects and executes operations (actions), e. g. , to prepare, start, stop, or destroy objects.

8.4.4.4 Specification Layer

- The specification layer is an "open" layer.
- It does not provide for an explicit interface.
- This layer includes applications and tools that can be used to generate synchronization specifications.
- Such tools include editors for synchronization and multimedia document and authoring systems, and tools used to convert specifications into an object-layer format.
- The specification layer is also responsible for mapping QoS requirements of the user level to the qualities offered at the object layer interface.

The methods used to specify synchronization can be grouped into the following main categories:

- *Interval-based specifications*, which allow the specification of temporal relationships between the time intervals of the presentation of media objects.
- *Axis-based specifications*, which set the presentation results in relation to the axes, which are used jointly by the presentation objects.
- *Control-flow-based specifications*, where the presentation flow is synchronized at specific synchronization points.
- *Events-based specifications*, where the events of the presentation of media are stated to trigger presentation actions.

Table 8.1: Overview on the layers of the synchronization reference model.

Layer	**Interface Abstraction**	**Tasks**
Specification	The tools performing the tasks of this layer have interfaces; the layer itself has no upper interface	Editing Formatting Mapping user-oriented QoS to the Qos abstraction at the object layer
Object	Synchronization specification Objects hiding the types of integrated media Media-oriented QoS (with regard to the acceptable skew and jitter)	Plan and coordinate presentation schedules Initiate the presentation of continuous media objects in the stream layer Initiate the presentation of discrete media objects Initiate presentation preparation actions

Table 8.1 continued from previous page

Layer	Interface Abstraction	Tasks
Stream	Arrange streams and stream groups Supply guarantees for intra-stream synchronization Supply guarantees for interstream synchronization	Reserve resources and schedule LDU processing
Media	Provide device-independent access to LDUs Supply guarantees for single LDU processing	File and device access

8.4.5 Synchronization Specification

- The synchronization specification of a multimedia object describes all *temporal dependencies* of the embedded objects.
- It is generated by means of tools offered in the specification layer and used at the interface to the object layer.
- The synchronization defines the entire presentation, so that it represents a central task in multimedia systems.
- The following list describes and evaluates the requirements to the synchronization specification and specification methods.

A synchronization specification should consist of the following components:

- Intra-object specification for the media objects embedded in the presentation.
- Specification of the inter-object synchronization for media objects embedded in the presentation.
- Description of the QoS parameters for intra-object synchronization.
- Description of the QoS parameters for inter-object synchronization.

The synchronization specification is part of the description of a multimedia object. In addition, it can describe the form to be used to represent a media object. For example, a piece of text could be written in the form of characters on the screen or generated as audio sequence. A specification can allow either of these two options or a choice of presentation forms at runtime.

For live synchronization, the temporal relationships are defined implicitly during the recording phase. The QoS requirements of each of the media involved are determined at the beginning of the recording phase.

If synthetic synchronization is applied, then the specification has to be explicitly created.

Specification Methods for Multimedia Synchronization

Complex specifications for multiple object synchronizations, including user interaction, require highly developed specification methods. The following requirements should be met by such a specification method:

- Should support object consistency and maintenance of synchronization specifications.
- Should supply an abstraction of the contents of each media object.
- Should allow easy description of all types of synchronization relationships.
- Should support the integration of continuous and discrete media objects.
- Should support the definition of QoS requirements.
- In addition, the method should support hierarchical synchronization levels to facilitate the processing of large and complex synchronization scenarios.

Following are the specification methods based on the above criteria.

8.4.5.1 Interval-based Specification

An interval-based synchronization specification considers the duration of the presentation of an object as an interval. Two time intervals can be synchronized in 13 different types, where some of these types can be inverted.

Figure 8.20 shows section of seven non-inverted rates. A simple method for synchronization specification of two media objects is able to use these seven types.

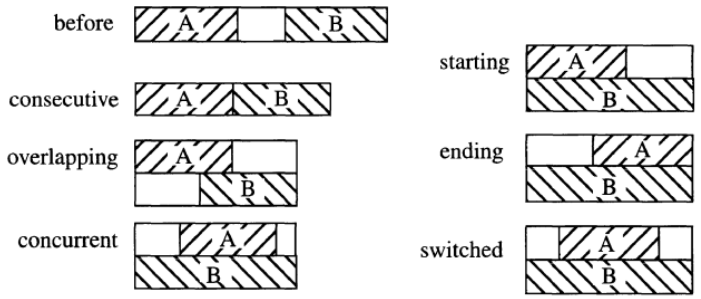

Figure 8.20: Types of temporal relationships between two objects.

Benefits

- Logical objects can be maintained.
- Good abstraction of media contents.
- Easy integration of discrete objects.
- Easy integration of interactive objects.

- Supports the specification of unknown temporal relations.

Drawbacks

- The specification is complex.
- Allows direct specification of temporal relationships between media objects, but not for subunits of media objects.
- Resolution of unknown relations at runtime can cause inconsistencies.

8.4.5.2 Axis-based Synchronization

Axis-based synchronization means that presentation events, such as start and end of a presentation, are mapped to axes, which are common to all objects of a presentation.

Synchronization Based on a Global Timer

- To implement a synchronization based on a global timer, all individual media objects are bound to an axis, representing an abstraction of the real world.
- This method describes the synchronization by linking all objects.
- They are mapped independently on a time axis.
- This means that the removal of one object does not influence the synchronization of the other objects.

Benefits

- Easy to understand.
- Supports easy implementation of hierarchies.
- Easy integration of discrete objects.
- Easy handling of objects thanks to mutual independence.
- Good abstraction of media contents.

Drawbacks

- Objects with unknown duration cannot be integrated; expansions to this model are required.
- The QoS skew has to be defined indirectly by a common time axis or additional QoS specification.

Synchronization Based on Virtual Axes We can use this specification method to specify coordinated systems with user-defined measurement units.

8.4.5.3 Control-flow-based Specification

Specifications based on the control flow synchronize the flow of coinciding presentation processes at predefined points within the presentation

Basic Hierarchical Specification Hierarchical synchronization descriptions are based on two main synchronization operations: the serial synchronization and the parallel synchronization.

Reference Points Specification In synchronization over reference points, continuous single media objects are handled as sequences of closed LDUs.

Time-specific Petri Networks Another type of specification is based on Petri networks, which expands the time specifications at various points.

8.4.5.4 Events-based Synchronization

In an events-based synchronization, presentation actions are initiated by synchronization events. Typical presentation actions are:

- Start a presentation.
- Stop a presentation.
- Prepare a presentation.

CHAPTER

9

MULTIMEDIA APPLICATION

The availability of multimedia hardware and software components has driven the enhancement of existing applications towards being more user-friendly. It has also initiated the continuous development of new multimedia applications.

9.1 Video-On-Demand

Video-On demand (VOD) services represent a class of applications where video information is accessed from one or more video servers.

VOD systems include many more components that are necessary for the provision of a complete service, such as video server(s), administration and maintenance systems, networking services, backbone networks for linking geographically distributed video servers and set-top units for receiving, demodulating, decoding and converting video for television playback.

Elements of a VOD system are shown in Figure 9.1.

The best known application of VOD is the *video library* which uses *Interactive VOD*. Interactive VOD allows a user to gain access to a movie via point-to-point connection. This connection allows the user individual and instantaneous control of storage medium in terms of *start*, *fast-forward*, *pause* and fast *rewind*.

There are two basic types of interactive VOD service:

- *Interactive VOD with instantaneous access*

  User can instantly retrieve and individually control program information from a library instantly, with instant control response.

- *Interactive VOD with delayed access*

  User retrieve and individually control program information from a library, but there is a waiting time depending on the available bandwidth resources in the network.

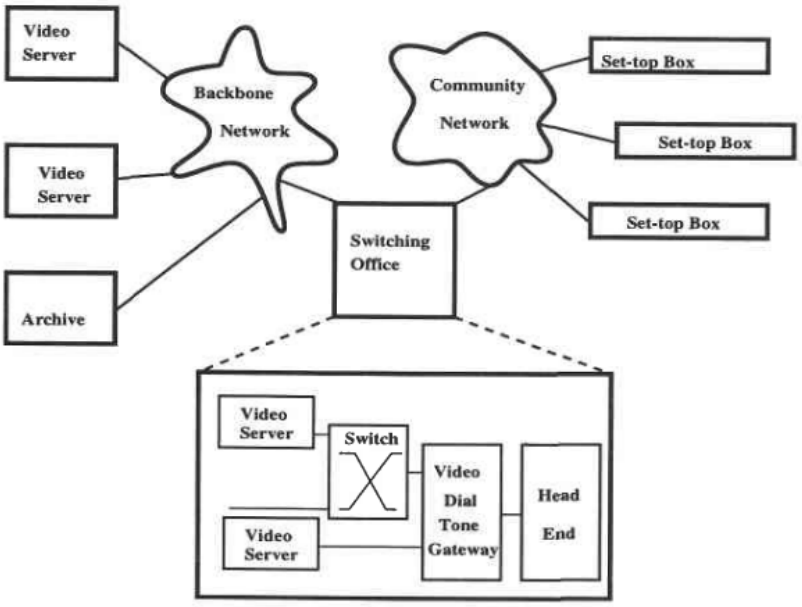

Figure 9.1: Video-On-Demand system.

9.2 Video Conferencing

Teleconferencing systems allow the user to achieve most of the efficiency and productivity of traditional meetings with one main difference: the user can stay at his/her desk as can the remote conference participants.

A multimedia conferencing system enables people to work together across geographically distant locations without the need to meet at one site. They communicate among each other in multi-party or face-to-face mode using motion video, audio and textual information in each direction. The audio

and video quality heavily depends on the platform. Therefore, a big factor in the success of a teleconferencing system is to achieve high media quality over any platform and interconnectivity among various platforms and vendors. A possible setup of a video conferencing system is shown in Figure 9.2.

Video conferencing allows participants in a live session to see each other; the video is transmitted over the network between users, live and in real-time.

Video conferencing is used either in an office environment, where the video is displayed on a PC or workstation screen, or in a conference room, where the video is displayed on a *video wall* (large TV screen).

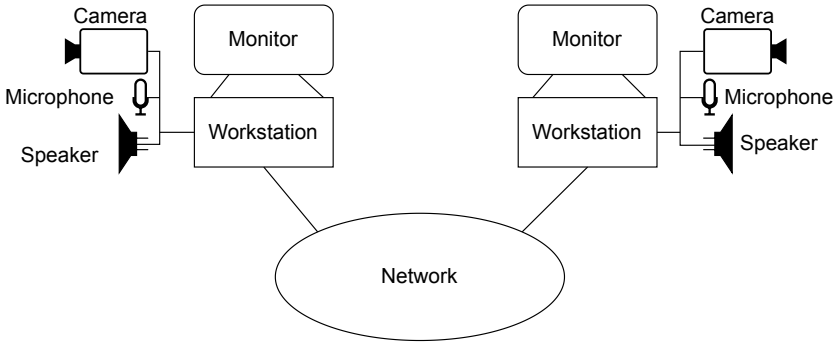

Figure 9.2: Video conferencing system.

Desktop video conferencing systems often include a dedicated shared white-board application. Application sharing denotes techniques which replicate the user interface of the particular software (e. g. , the user's favorite text processor) so that the software can be used simultaneously by all participants of a conference. The concurrency in the activities underlies the mechanisms of "floor passing" (also called "chalk passing") to determine which one of the users may actually interact with the software at a given time.

9.3 Educational Application, Industrial Application

9.3.1 Educational Application

The world in which we live is changing rapidly and the field of education is experiencing these changes in particular as it applies to Media Services. The old days of an educational institution having an isolated audio-visual department are long gone! The growth in use of multimedia within the education sector has accelerated in recent years, and looks set for continued expansion in the future.

Teachers primarily require access to learning resources, which can support concept development by learners in a variety of ways to meet individual learning needs. The development of multimedia technologies for learning offers new ways in which learning can take place in schools and the home. Enabling teachers to have access to multimedia learning resources, which support constructive concept development, allows the teacher to focus more on being a facilitator of learning while working with individual students. Extending the use of multimedia learning resources to the home represents an educational opportunity with the potential to improve student learning.

The elements used in multimedia have all existed before. Multimedia simply combines these elements into a powerful new tool, especially in the hands of teachers and students. Interactive multimedia weaves five basic types of media into the learning environment: text, video, sound, graphics and animation. Since the mode of learning is interactive and not linear, a student or teacher can choose what to investigate next. For example, one does not start on the first page of a linear document and read to the end. Interactive multimedia learning mode is more like constructing a spider's web, with one idea linked to another, allowing choices in the learner's path.

The multimedia technologies that have had the greatest impact in education are those that augment the existing curriculum, allowing both immediate enhancement and encouraging further curriculum development. For example, the WWW serves as a storehouse of information that individual learners can search for subject matter content that specifically fits their learning agendas. Multimedia applications for computers have been developed for single computing platforms such as the PC, Apple Mac and games machines.

9.3.2 Industrial Application

- Advertising
- Gaming industry
- Entertainment industry
- Fine arts
- Engineering
- Fashion design
- AI

9.4 Information System, Multimedia Archives & Digital Libraries, Media Editors

9.4.1 Information Systems

Multimedia Information Systems (MIS) couple data management capabilities (effective and efficient methods for storage, indexing, querying, and retrieval) with media management (media representation, data compression, standardization, and transmission).

Multimedia applications require representation and management of non-traditional data, such as text documents, images, audio and video data, possibly together with traditional (e. g. , relational or object-oriented) data. In particular, in multimedia applications, different data types are often coexisting within the same application domain. Hence, data management systems which are able to treat, in an integrated and uniform way, different types of data are needed.

Multimedia information management systems, by definition, are not dealing with only a single media data type, but in fact need to be designed with the semantics and requirements of multiple media and modalities in mind.

A multimedia information system (MIS) architecture, consists of the following sub-systems:

- a multimedia authoring system (MAS),
- a media sensing system (MSS),
- media processing system (MPS),
- multimedia communication system (MCS),
- a multimedia visualization and interaction system
- a multimedia object database (MODB)
- a profile and context manager (PCM), and
- a digital rights management system (DRMS).

1. MAS is responsible for authoring of multimedia objects. This involves describing visual, temporal, spatial, hierarchical, and interactive properties of a complex multimedia object.

2. MSS uses environmentally distributed sensing devices to collect multimedia data relevant for a given application.

3. MCS communicates media and multimedia objects between the various components of a multimedia information system, providing appropriate quality of service (QoS) guarantees.

4. Since row, sensed data is often not directly usable, MPS processes media objects for the benefit of the other components. For example, multiple media objects may be fused to obtain a composite object, a media object may be processed to adjust its quality, or certain features may be extracted from a multimedia object for indexing purposes.

5. MVIS benefits from the user specifications to create presentation schedules which maximize the utilization of available resources. It also enables the user to interact with the media objects and explore the multimedia information space. It performs document and media scaling to match the resource requirements to the resource availabilities. It also benefits from document structure, priorities, user preferences, and quality/cost trade-offs to develop object prefetching and caching strategies for document presentation.

6. MODB is responsible for storage and retrieval of media objects and multimedia documents. MODB enables multimedia information to be queried and retrieved, efficiently and effectively. For this purpose it maintains appropriate index structures to support queries. Since multimedia retrieval is subjective, unlike traditional databases, MODB employs fuzzy or probabilistic query processing techniques and ranking algorithms to present the query results according to their relevance

7. PCM is responsible for keeping the user, context, and task profiles to improve the processing and presentation of multimedia information.

8. DRSM ensures the intellectual property rights of the users who contribute multimedia objects into the multimedia information system by providing digital signatures and copy detection mechanisms.

9.4.2 Multimedia Archives

The integration of multimedia archives through a common, unified access point for end users, always considering their particular copyright and access policies, emerges as a necessary step for the preservation of their content and their financial viability.

During the last decade, the cost of storage and wide area communication services has decreased, while their capacity increased dramatically. This fact, along with the increasing penetration of e-commerce applications, has made digital storage, annotation and access of multimedia information a mature and viable choice for content holding organizations and individuals. Numerous multimedia archives have, either totally or incrementally, turned to the utilization of digitized archive technologies. The content of these archives can be made accessible, depending upon copyright, policy and security decisions, over the Internet in a cost-efficient, time efficient, anyplace, anytime fashion.

However, one of the main problems of multimedia archiving has been inherited to their digital descendants. For traditional archives, where raw media were stored in the form of analog hard copies, search was not an easy task as a human had to either go through a separate annotation archive, or, ideally, search using keywords in a custom, proprietary, metadata database. Much similarly to the case of books in a library that have not been indexed, information stored in a multimedia archive that cannot be searched, identified and accessed easily is practically unavailable.

9.4.3 Digital Libraries

Digital Library is an information system targeted towards a specific community, where content from different sources is collected and managed, content is structured and enriched with metadata, and a set of services is offered that makes the content available to a user community via a communication network, typically the Internet.

Digital Libraries are the electronic counterparts of traditional paper libraries, where the digital medium opens new opportunities, especially in the area of improved access support, increased content availability, powerful content interlinking, and reduced costs, but also imposes new challenges like long-term preservation in the context of fast changing storage technologies. Further important challenges are issues of copyright and digital rights management and the cost of digitization for not digitally-born content.

A Digital Library is an information system targeted towards a specific community, where content from different sources is collected and managed, content is structured and enriched with metadata, and a set of services is offered that makes the content available to a user community via a communication network, typically the Internet. Multimedia Libraries are Digital Libraries, where the managed content is not restricted to the usually mainly textual documents. Such libraries contain, next to the ‘‘textual’’ contents, media types like music, videos, images, maps, and mixtures of different con-

tent types (multimedia objects) as they are, for example used in e-Learning or in the documentation of history.

Multimedia libraries may also contain content types that were not supported in traditional libraries at all like 3D objects, executable software (e. g. , computer games) or callable services. One of the main challenges for a multimedia library is to provide effective access to these types of context (based on adequate indexing) and to provide support for the ‘‘real-time’’ integration of different content types. Some challenges of multimedia libraries are closely related to those of museums and archives that make multimedia representations of their artifacts available online.

9.4.4 Media Editors

Media composition involves editing single media, i. e. changing its objects, such as characters, audio sentences, video frames and attributes such as the font of a character, recording speed of an audio sentence or color of an image.

- Text and Graphics editors
- Image editors
- Animation editors
- Sound editors
- Video editors

9.4.4.1 Text Editors

Text editors provide writing and modifying facilities to compose text in a document. There are either separate text editors (e. g. , Emacs, Vim, Nano text editor in combination with the LaTeX[1] document preparation tool on workstations, LibreWriter on PCs) or text is embedded in graphical tools such as drawing programs CorelDRAW.

An example of an advanced word processor with graphical capabilities is *Microsoft Word*, *Libre Writer*. These tools provides, in addition to text capabilities, a new toolbar and ribbon that can be customized for the creation oftables, envelopes, bullets and more.

[1]यो पूरै नोट तयार पार्न पनि LaTeX नै प्रयोग गरिएको छ। https://www.latex-project.org/

9.4.4.2 Graphics Editors

Graphics editors use facilities at the user interface for editing structural representations of graphical objects (structure-level editing) and for modifying higher-level operations on graphical objects (object-level editing). These two levels of editing are possible because the graphical system stores object primitives and their structural representations, which can be manipulated. These drawing application, are also called *layout editor* or *graphical illustrator*. Example Adobe Illustrator, CorelDRAW, etc.

9.4.4.3 Image Editors

Image editors are suitable for applications when neither the application nor the underlying software package keeps a record of the primitives (as is typical in most painting programs).

An example of a graphics/image editor is *Adobe Photoshop*. This tool allows one to draw, edit and paste objects on several layers. Experimenting with different combinations of graphics, text and special effects without altering the original background image is possible. Furthermore, Filters let the user create 3D lighting effects, and remove dust and scratches from scanned images.

9.4.4.4 Animation Editors

Animation editing is based on graphical editors with respect to 2D or 3D spatial graphic objects. The additional component in animation is time, which can also be edited (4D editing). The functionalities of such editors include c*utting frames from an animation clip*, *adding new frames to an animation clip*, etc.

The animation tools provide the animator with the capability to draw only the key frame. The intermediate frames are then drawn by the computer animation program. This process is called *tweening*. Further, some animation tools include morphing (polymorphic tweening), which is a transformation from one shape to another. With this technique, many special effects can be created.

9.4.4.5 Sound Editors

Sound tools support a number of operations that let the user access, modify and play sound data. The operations fall into following categories:

1. *Locating and Storing Sounds*

2. *Recording and Playback*

3. *Editing*

Locating and Storing Sounds Location and storage of sounds can be done in four ways:

1. *record* a sound using an A/D audio device (analog-to-digital converter),

2. *read* sound data from a sound file,

3. *retrieve* a sound from a pasteboard, and

4. *create* sound data algorithmically.

Recording and Playback The record operation continuously records sound from a microphone input until it is stopped or paused. Recorded sound is *m-law encoded*. The data are sampled at a rate of 8000 samples per second with 12-bit precision, but if the digitized sound is being compressed, then 8-bit precision per sample is achieved.

The playback operation plays a sound using a D/A audio device (digital-to-analog converter) speaker output.

Editing The editing operation allows one to copy/paste, cut, delete, insert or replace sampled sound data. One problem ought to be pointed out here: audio data are normally contiguous in memory. However, when a sound object is edited, its data can become fragmented or discontiguous. Fragmented sounds are played less efficiently. Hence, it is important to have an operation which compacts the samples into a contiguous object.

9.4.4.6 Video Editors

Video editors are based on image editors for editing individual frames, but as in the case of animation editing, temporal considerations are important. Therefore, time resolution (time aliasing) is solved if frames are deleted, added or replaced. Editing functionalities of video editors may combine several cuts into one sequence, adjust audio separately from video and add video transition effects. An example of such a motion video editor is *Adobe Premiere*, *Edius*, *Final Cut*, *Kdenlive* etc.

In the case of a conventional videotape, the edited sequence of video frames must be recorded to a new tape to view the new video clip. This kind of editing is called *linear editing*. Video editor provides an *Edit Decision*

List (EDL) from which the final video can be reconstructed, i. e. , the edited video does not have to be recorded to a new tape because it can be played continuously using the EDL. This is called *non-linear editing*. Such tools have further editing capabilities, e. g. adding dynamic transitions such as wipes, dissolves or fades between cuts.

Video editors include several editors for editing video, sound and music in an integrated fashion.

PURBANCHAL UNIVERSITY

2018/ २०७५

4 Years Bachelor of Computer Application (BCA/Eighth Semester/Final)

Time: 3.00 hrs. Full Marks: 60 / Pass Marks: 24

BCA452CO, Multimedia Application

Candidates are required to give their own answers in their own words as far as practicable. Figure in the margin indicate full marks.

Group A

Answer TWO questions. **$2 \times 12 = 24$**

1. How image and graphics is represented in PC? Describe image synthesis and image analysis. Explain image recognition steps with the help of block diagram.

2. (a) Explain importance of data compression with its issues. Compare source coding, entropy coding, and hybrid coding.

   (b) Describe video compression techniques used in MPEG system.

3. What is data conversion? Explain with example. Briefly discuss the concept of interface.

Group B

Answer SIX questions. **$6 \times 6 = 36$**

4. Explain multimedia and multimedia system. Explain data stream characteristics with an appropriate figure.

5. Compare audio, music and sound. Explain the components of speech recognition and understanding.

6. Explain how text is used in multimedia application. What is bit map image?

7. Define resource management architecture. Discuss the relation between QoS and resource management.

8. Why is the reference model for synchronization required and how is it defined? Explain.

9. Define basic technology of optical storage media. Explain CD- ROM extended architecture.

10. Write short notes on any TWO:
    (a) Nyquist's sampling theorem
    (b) Libraries and Toolkit
    (c) Lossy compression algorithm

PURBANCHAL UNIVERSITY

2019/ २०७६

4 Years Bachelor of Computer Application (BCA/Eighth Semester/Final)

Time: 3.00 hrs. Full Marks: 60 / Pass Marks: 24

BCA452CO, Multimedia Application

Candidates are required to give their own answers in their own words as far as practicable. Figure in the margin indicate full marks.

Group A

Answer TWO questions. **$2 \times 12 = 24$**

1. (a) How multimedia is necessary in real world? Explain multimedia with its classification.

   (b) What is data stream characteristics? Explain traditional data stream characteristic in multimedia.

2. Explain the types of image format in brief. Explain Image recognition steps with the help of block diagram.

3. Why synchronization is required in multimedia? Explain with the help of example. How four layer referenced model explains the concept of synchronization?

Group B

Answer SIX questions. **$6 \times 6 = 36$**

4. Define MIDI. Explain MIDI channel message and MIDI system message.

5. Explain basic concepts of animation. Explain how animation can be controlled?

6. Why do we need compression? Explain JPEG Audio Encoding in brief.

7. What is multimedia operating system? Explain resource management techniques in multimedia operating system.

8. Explain Abstraction level of programming with the help of the block diagram.

9. Explain Group Communication Architecture used in multimedia communication system.

10. Write short notes on any TWO:

(a) Video on demand

(b) Entropy Encoding

(c) Document Architecture

BIBLIOGRAPHY

Kayastha, S. (2020). Multimedia Application: BCA-VIII. *Gomendra Multiple College*.

Defining Multimedia. (n.d.). https://www1.udel.edu/edtech/multimedia/index.html

Ralf Steinmetz and Klara Nahrstedt. (1995). *Multimedia: Computing, Communications and Applications*. Prentice Hall PTR Prentice-Hall, Inc.

Prof David Marshall and Dr Kirill Sidorov. (n.d.). CM3106 Multimedia. *School of Computer Science & Informatics*. https://users.cs.cf.ac.uk/Dave.Marshall/Multimedia

Ralf Steinmetz and Klara Nahrstedt. (2004). *Multimedia Applications*. Springer.

Klara Nahrstedt and Ralf Steinmetz. (2004). *Multimedia Systems*. Springer.

Ralf Steinmetz and Klara Nahrstedt. (2002). *Multimedia Fundamentals Volume 1: Media Coding and Content Processing*. Prentice-Hall, Inc.

JPEG Compression - Javatpoint. (n.d.). Retrieved February 15, 2021, from https://www.javatpoint.com/jpeg-compression

Process Of JPEG Data compression. (2020, April). Retrieved February 15, 2021, from https://www.geeksforgeeks.org/process-of-jpeg-data-compression/

JPEG Image Compression. (2016, January). Retrieved February 15, 2021, from https://www.slideshare.net/AishwaryaKM1/jpeg-image-compression-56894348

Asthana, A. (2006). Multimedia in education. In B. Furht (Ed.), *Encyclopedia of multimedia* (pp. 533–540). Springer US. https://doi.org/10.1007/0-387-30038-4_154

Wallace, M., Avrithis, Y., & Kollias, S. (2006). Multimedia archives and mediators. In B. Furht (Ed.), *Encyclopedia of multimedia* (pp. 452–460). Springer US. https://doi.org/10.1007/0-387-30038-4_140

Neuhold, E., & Nieder'e, C. (2006). Multimedia libraries. In B. Furht (Ed.), *Encyclopedia of multimedia* (pp. 541–547). Springer US. https://doi.org/10.1007/0-387-30038-4_155