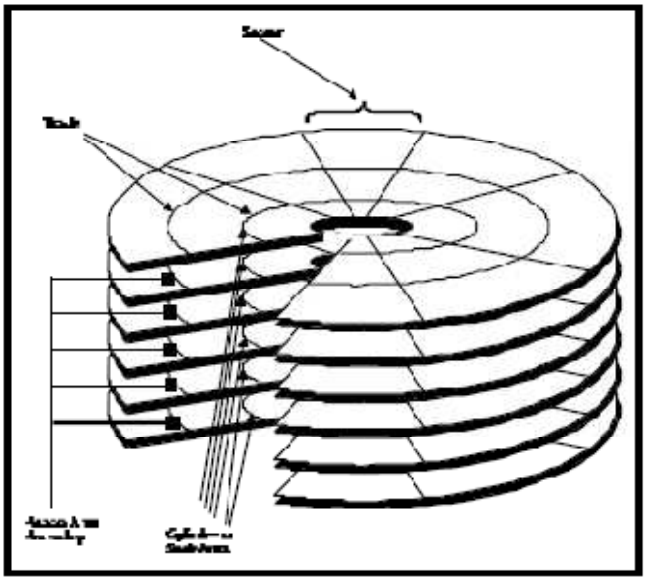
UNESCO-NIGERIA TECHNICAL & VOCATIONAL
EDUCATION REVITALISATION PROJECT-PHASE II

NATIONAL DIPLOMA IN

COMPUTER TECHNOLOGY



File Organization and Management

YEAR II- SE MESTER I

THEORY

Version 1: December 2008

Table of Contents

WEEK 1

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ The concept of file in computing
- ✓ The concept of record, field, character, byte and bits in relation to a file

File Concepts

In this section, we shall deal with the concepts of file & their relationship.

Many operating system consider files as a collection of bytes. At a higher level, where the content of the file is being considered, these binary digits may represent integer values or text characters, or anything else. It is up to the program using the file to understand the meaning and internal layout of information in the file and present it to a user as more meaningful information (like text, images, sounds).

At any instant in time, a file might have a size, normally expressed in bytes, that indicates how much storage is associated with the file. In most modern operating systems the size can be any whole number up to a system limit. However, the general definition of a file does not require that its instant size has any real meaning, unless the data within the file happens to correspond to data within a pool of persistent storage.

Information in a computer file can consist of smaller packets of information, often called "records" or "lines" that are individually different but share some trait in common. For example, a payroll file might contain information concerning all the employees in a company and their payroll details; each record in the payroll file concerns just one employee, and all the records have the common trait of being related to payroll. This is very similar to placing all payroll information into a specific filing cabinet in an office that does not have a computer. A text file may contain lines of text, corresponding to printed lines on a piece of paper. Alternatively, a file may contain an arbitrary binary image (a BLOB) or it may contain an executable.

The way information is grouped into a file is entirely up to the person designing the file. This has led to a plethora of more or less standardized file structures for all imaginable purposes, from the simplest to the most complex. Most computer files are used by computer programs. These programs create, modify and delete files for their own use on an as-needed basis. The programmers who create the programs decide what files are needed, how they are to be used and their names.

In some cases, computer programs manipulate files that are made visible to the computer user. For example, in a word-processing program, the user manipulates document files that she names herself. The content of the document file is arranged in a way that the word-processing program understands, but the user chooses the name and location of the file, and she provides the bulk of the information (such as words and text) that will be stored in the file.

Many applications pack all their data files into a single file, using internal markers to discern the different types of information contained within.

Files on a computer can be created, moved, modified, grown, shrunk and deleted. In most cases, computer programs that are executed on the computer handle these operations, but the user of a computer can also manipulate files if necessary. For instance, Microsoft Word files are normally created and modified by the Microsoft Word program in response to user commands, but the user can also move, rename, or delete these files directly by using a file manager program such as Windows Explorer.

In Unix-like systems, user-space processes do not normally deal with files at all; the operating system provides a level of abstraction which means that almost all interaction with files from user-space is through hard links. Hard links allow a name to be associated with a file (or they can be anonymous - and therefore temporary); files do not have names in the OS. For example, a user-space program cannot delete a file; it can delete a link to a file (for example, using the shell commands rm or mv or, in the anonymous case, simply by exiting), and if the kernel determines that there are no more existing links to the file, it may then delete the file. In fact, it really is only the kernel that deals with files, but it serves to handle all user-space interaction with (virtual) files in a manner that is transparent to the user-space programs.

- **Bit:** A Bit is simply defined as binary digits, that’s 0’s &1’s. The bit is the smallest unit of storage.

Quantities of bits				
SI prefixes			**Binary prefixes**	
Name (Symbol)	**Standard SI**	**Binary usage**	**Name (Symbol)**	**Value**
kilobit (kbit)	10^3	2^{10}	kibibit (Kibit)	2^{10}
megabit (Mbit)	10^6	2^{20}	mebibit (Mibit)	2^{20}
gigabit (Gbit)	10^9	2^{30}	gibibit (Gibit)	2^{30}
terabit (Tbit)	10^{12}	2^{40}	tebibit (Tibit)	2^{40}
petabit (Pbit)	10^{15}	2^{50}	pebibit (Pibit)	2^{50}
exabit (Ebit)	10^{18}	2^{60}	exbibit (Eibit)	2^{60}
zettabit (Zbit)	10^{21}	2^{70}	zebibit (Zibit)	2^{70}
yottabit (Ybit)	10^{24}	2^{80}	yobibit (Yibit)	2^{80}

.

A **bit** is a binary digit, taking a value of either 0 or 1. Binary digits are a basic unit of information storage and communication in digital computing and digital information theory. Information theory also often uses the natural digit, called either a *nit* or a *nat*. Quantum computing also uses qubits, a single piece of information with a probability of being true.

The bit is also a unit of measurement, the information capacity of one binary digit. It has the symbol **bit**, or **b** (see discussion below).

Binary digit

Claude E. Shannon first used the word ***bit*** in his 1948 paper *A Mathematical Theory of Communication*. He attributed its origin to John W. Tukey, who had written a Bell Labs

memo on 9 January 1947 in which he contracted "binary digit" to simply "bit". Interestingly, Vannevar Bush had written in 1936 of "bits of information" that could be stored on the punch cards used in the mechanical computers of that time. [1]

A bit of storage can be either on (1) or off (0). A single bit is a one or a zero, a true or a false, a "flag" which is "on" or "off", or in general, the quantity of information required to distinguish two mutually exclusive equally probable *states* from each other. Gregory Bateson defined a bit as "a difference that makes a difference".[2]

Representation

Transmission

Bits can be implemented in many forms depending on context. For example, in digital circuitry in most computing devices as well as flash memories, a bit is an electrical pulse generated by the internal clock in the control unit or data register. For devices using positive logic, a logical 1 (true value) is represented by up to 5 volts, while a logical 0 (false value) is represented by 0 volt.

Storage

Bits are manipulated in the volatile memory of a computer, and can further be kept in a persistent manner on a magnetic storage device such as magnetic tape or disc, as well as on optical discs.

Unit

It is important to differentiate between the use of "bit" in referring to a discrete storage unit and the use of "bit" in referring to a statistical unit of information. The bit, as a discrete storage unit, can by definition store only 0 or 1. A statistical bit is the amount of information that, *on average*[*citation needed*], can be stored in a discrete bit. It is thus the amount of information carried by a choice between two equally likely outcomes. One bit corresponds to about 0.693 nats ($\ln(2)$), or 0.301 hartleys ($\log_{10}(2)$).

Consider, for example, a computer file with one thousand 0s and 1s which can be losslessly compressed to a file of five hundred 0s and 1s (on average, over all files of that kind). The original file, although having 1,000 bits of storage, has at most 500 bits of information entropy, since information is not destroyed by lossless compression. A file can have no more information theoretical bits than it has storage bits. If these two ideas need to be distinguished, sometimes the name *bit* is used when discussing data storage while *shannon* is used for the statistical bit. However, most of the time, the meaning is clear from the context.

Abbreviation and symbol

No uniform agreement has been reached yet about what the official unit symbols for bit and byte should be. One commonly-quoted standard, the International Electrotechnical Commission's IEC 60027, specifies that "bit" should be the unit symbol for the unit bit (e.g. "kbit" for kilobit). In the same standard, the symbols "o" and "B" are specified for the byte.

Today the harmonized ISO/IEC IEC 80000-13:2008 standard cancels and replaces subclauses 3.8 and 3.9 of IEC 60027-2:2005 (those related to Information theory and Prefixes for binary multiples).

The other commonly-quoted relevant standard, IEEE 1541, specifies "b" to be the unit symbol for bit and "B" to be that for byte. This convention is also widely used in computing, but has so far not been considered acceptable internationally for several reasons:

- both these symbols are already used for other units: "b" for barn and "B" for bel;
- "bit" is already short for "binary digit", so there is little reason to abbreviate it any further;
- it is customary to start a unit symbol with an uppercase letter only if the unit was named after a person (see also Claude Émile Jean-Baptiste Litre);

- instead of byte, the term octet (unit symbol: "o") is used in some fields and in some Francophone countries, which adds to the difficulty of agreeing on an international symbol;
- "b" is occasionally also used for byte, along with "bit" for bit.

The unit bel is rarely used by itself (only as decibel, "dB", which is unlikely to be confused with a decibyte), so the chances of conflict with "B" for byte are quite small, even though both units are very commonly used in the same fields (e.g., telecommunication).

More than one bit

A byte is a collection of bits, originally differing in size depending on the context but now almost always eight bits. Eight-bit bytes, also known as *octets*, can represent 256 values (2^8 values, 0–255). A four-bit quantity is known as a *nibble*, and can represent 16 values (2^4 values, 0–15). A rarely used term, *crumb*, can refer to a two-bit quantity, and can represent 4 values (2^2 values, 0–3).

"Word" is a term for a slightly larger group of bits, but it has no standard size. It represents the size of one register in a Computer-CPU. In the IA-32 architecture more commonly known as x86-32, 16 bits are called a "word" (with 32 bits being a double word or dword), but other architectures have word sizes of 8, 32, 64, 80 or others.

Terms for large quantities of bits can be formed using the standard range of SI prefixes, e.g., kilobit (kbit), megabit (Mbit) and gigabit (Gbit). Note that much confusion exists regarding these units and their abbreviations (see above).

When a bit within a group of bits such as a byte or word is to be referred to, it is usually specified by a number from 0 (not 1) upwards corresponding to its position within the byte or word. However, 0 can refer to either the most significant bit or to the least significant bit depending on the context, so the convention being used must be known.

Certain bitwise computer processor instructions (such as *bit set*) operate at the level of manipulating bits rather than manipulating data interpreted as an aggregate of bits.

Telecommunications or computer network transfer rates are usually described in terms of bits per second (*bit/s*), not to be confused with baud.

Bit

Binary digit? A digit is the name given to a number used in the positional value notation when using **base 10** --18.239.6.217 (talk) 03:56, 30 May 2008 (UTC).

It is stated that decimal forms of the SI factors (10^2, 10^4..) are used when related to telecommunications. Thus we can agree that **56 kilobits per second** essentially means **56 000 bits per second**. That thus gives us **7 000 bytes per second**. However, when we carry on to convert to *kilobytes per second*, should we not use the binary form of the prefix, i.e. **7 000 bps = 7 000 / 1024 = 6.836 kilobytes per second**?

Bits, Isn't a bit also 12.5 cents? as in two bits equating a quarter? as in a shave and a haircut, two bits? if so, that needs to be expressed on this article, and the derivation should be explained to. thank! Kingturtle 23:00, 24 Apr 2005 (UTC)

Bit, trit, dontcare, what?

Two is to bit as three is to trit as five is to what? I've been calling them quints, but... -- Call it quits. Or a nybble and a bit.

> The proper name would be "quinary digits", so quints is probably as correct as anything.

RfC on trivial bits

- Nibble: Nibble is ½ a byte. Generally, it is 4 bits, e.g 0110, 1110.

Byte:

This is the collection of 2 nibbles or 8 bits. E.g, 010101010. A byte fixed number of bits that can be treated as a unit by the computer. It represents a character. It can also be described as the smallest unit of the computer memory. The word "byte" has two closely related meanings:

- A contiguous sequence of a *fixed* number of bits (binary digits). The use of a byte to mean 8 bits has become nearly ubiquitous.

- A contiguous sequence of bits within a binary computer that comprises the *smallest addressable sub-field* of the computer's natural word-size. That is, the smallest unit of binary data on which meaningful computation, or natural data boundaries, could be applied.

Decimal			Binary				
Value	SI		Value	IEC		JEDEC	
1000^1	k	kilo-	1024^1	Ki	kibi-	K	kilo-
1000^2	M	mega-	1024^2	Mi	mebi-	M	mega-
1000^3	G	giga-	1024^3	Gi	gibi-	G	giga-
1000^4	T	tera-	1024^4	Ti	tebi-		
1000^5	P	peta-	1024^5	Pi	pebi-		
1000^6	E	exa-	1024^6	Ei	exbi-		
1000^7	Z	zetta-	1024^7	Zi	zebi-		
1000^8	Y	yotta-	1024^8	Yi	yobi-		

- A **byte** is the basic unit of measurement of information storage. In many computer architectures it is a unit of memory addressing, most often consisting of eight bits. A byte is one of the basic integral data types in some programming languages, especially system programming languages.

- A byte is an ordered collection of bits, with each bit denoting a single binary value of 1 or 0. The size of a byte can vary and is generally determined by the underlying computer operating system or hardware, although the 8-bit byte is the standard in modern systems. Historically, byte size was determined by the number of bits required to represent a single character from a Western character set. Its size was generally determined by the number of possible characters in the supported character set and was chosen to be a divisor of the computer's word size. Historically bytes have ranged from five to twelve bits.
- The popularity of IBM's System/360 architecture starting in the 1960s and the explosion of microcomputers based on 8-bit microprocessors in the 1980s has made eight bits by far the most common size for a byte. The term octet is widely used as a more precise synonym where ambiguity is undesirable (for example, in protocol definitions).
- There has been considerable confusion about the meanings of metric -- or SI prefixes -- used with the word "byte", especially concerning prefixes such as kilo- (k or K) and mega- (M) as shown in the chart *Prefixes for bit and byte*. Since computer memory comes in a Power of two rather than 10, a large portion of the software and computer industry use binary estimates of the SI-prefixed quantities, while producers of computer storage devices prefer the SI values. This is why a computer hard drive advertised with a "100 GB" decimal storage capacity actually contains no more than 93 GB of 8-bit (power of 2) addressable storage. Because of the confusion, a contract specifying a quantity of bytes must define what the prefixes mean in terms of the contract (i.e., the alternative binary equivalents or the actual decimal values, or a binary estimate based on the actual values).
- To make the meaning of the table absolutely clear: A kibibyte (KiB) is made up of 1,024 bytes. A mebibyte (MiB) is made up of 1,024 × 1,024 i.e. 1,048,576 bytes. The figures in the column using 1,024 raised to powers of 1, 2, 3, 4 and so on are in units of bytes.

Alternative words

Following "bit," "byte," and "nybble," there have been some analogical attempts to construct unambiguous terms for bit blocks of other sizes.[6] All of these are strictly jargon, and not very common.

- 2 bits: crumb, quad, quarter, tayste, tydbit
- 4 bits: nibble, nybble
- 5 bits: nickel, nyckle
- 10 bits: deckle
- 16 bits: plate, playte, chomp, chawmp (on a 32-bit machine)
- 18 bits: chomp, chawmp (on a 36-bit machine)
- 32 bits: dinner, dynner, gawble (on a 32-bit machine)
- 48 bits: gobble, gawble (under circumstances that remain obscure)

- Character: A character is the smallest unit of information. It includes letters, digits and special characters such as; +, -, = *, %, &, etc.
- Field: A field is made up of characters, or simply defined as a combination of characters. It's an attribute that can be assigned values. For example, First name, Age, Gender, Date of birth, etc.
- Record: Record is the collection of organizes and related fields.

File

File is the collection of organized and related records.

Identifying and organizing files

- File is the collection of organized and related records. Files and folders arranged in a hierarchy. Files are typically accessed using names (filenames). In some operating systems, the name is associated with the file itself. In others, the file is anonymous, and is pointed to by links that have names. In the latter case, a user can identify the name of the link with the file itself, but this is a false analogue, especially where there exists more than one link to the same file.
- Files or links to files can be located in directories. However, more generally, a directory can contain either a list of files or a list of links to files. Within this definition, it is of important that the term "file" includes directories. This permits the existence of directory hierarchies, i.e., directories containing subdirectories. A name that refers to a file within a directory must be unique. In other words, there must be no identical names within a directory. However, in some operating systems, a name may include a specification of type that means a directory can contain an identical name for more than one type of object such as a directory and a file.
- In environments in which a file is named, a file's name and the path to the file's directory must uniquely identify it among all other files in the computer system. No two files can have the same name and path. Where a file is anonymous, named references to it will exist within a namespace.
- Any string of characters may or may not be a well-formed name for a file or a link depending upon the context of application. Whether or not a name is well-formed depends on the type of computer system being used. Early computers permitted only a few letters or digits in the name of a file, but modern computers allow long names (some up to 255) containing almost any combination of unicode letters or unicode digits, making it easier to understand the purpose of a file at a glance.
- Some computer systems allow file names to contain spaces; others do not. Case-sensitivity of file names is determined by the file system. Unix file systems are usually case sensitive and allow user-level applications to create files whose names differ only in the case of characters.

- Microsoft Windows supports multiple file systems, each with different policies regarding case-sensitivity. The common FAT file system can have multiple files whose names differ only in case if the user uses a disk editor to edit the file names in the directory entries. User applications, however, will usually not allow the user to create multiple files with the same name but differing in case.
- Most computers organize files into hierarchies using folders, directories, or catalogs. The concept is the same irrespective of the terminology used. Each folder can contain an arbitrary number of files, and it can also contain other folders. These other folders are referred to as subfolders. Subfolders can contain still more files and folders and so on, thus building a tree-like structure in which one "master folder" (or "root folder" — the name varies from one operating system to another) can contain any number of levels of other folders and files. Folders can be named just as files can (except for the root folder, which often does not have a name). The use of folders makes it easier to organize files in a logical way.
- When a computer allows the use of folders, each file and folder has not only a name of its own, but also a path, which identifies the folder or folders in which a file or folder resides. In the path, some sort of special character such as a slash "—" is used to separate the file and folder names.
- For example, in the illustration shown in this article, the path **/Payroll/Salaries/Managers** uniquely identifies a file called **Managers** in a folder called **Salaries**, which in turn is contained in a file called **Payroll**. The folder and file names are separated by slashes in this example; the topmost or root folder has no name, and so the path begins with a slash (if the root folder had a name, it would precede this first slash).
- Many (but not all) computer systems use extensions in file names to help identify what they contain, also known as the file type. On Windows computers, extensions consist of a dot (period) at the end of a file name, followed by a few letters to identify the type of file. An extension of **.txt** identifies a text file; a **.doc** extension identifies any type of document or documentation, commonly in the Microsoft Word file format; and so on. Even when extensions are used in a

computer system, the degree to which the computer system recognizes and heeds them can vary; in some systems, they are required, while in other systems, they are completely ignored if they are present.

Database: Is the collection of organized and related files. For instance, the collection of staff-file, student-file & equipment-file of an institution is referred to as the **Institution's Database**. This collection of data organized for storage in a computer memory and designed for easy access by authorized users. The data may be in the form of text, numbers, or encoded graphics. Since their first, experimental appearance in the 1950s, databases have become so important in industrial societies that they can be found in almost every field of information. Government, military, and industrial databases are often highly restricted, and professional databases are usually of limited interest. A wide range of commercial, governmental, and nonprofit databases are available to the general public, however, and may be used by anyone who owns or has access to the equipment that they require.

Small databases were first developed or funded by the U.S. government for agency or professional use. In the 1960s, some databases became commercially available, but their use was funneled through a few so-called research centers that collected information inquiries and handled them in batches. Online databases are databases available to anyone who could link up to them by computer. For the home user, the equipment required includes a computer terminal, a telephone, and a modem, which enables the terminal and the database (usually some type of search-service system) to intercommunicate. Modified television sets can also be equipped to receive some specifically designed database services. The user simply dials the number of the service, provides a password code for identification and billing, and types in questions to a chosen database on the terminal's keyboard. The data received may either be displayed on a terminal screen or printed out.

Objectives of Database Implementation

The implementation of database was prompted by the problems of file system

i. Data redundancy
ii. Lack of standard
iii. Lack of central control
iv. Lack of data independence

Basic structuring concept

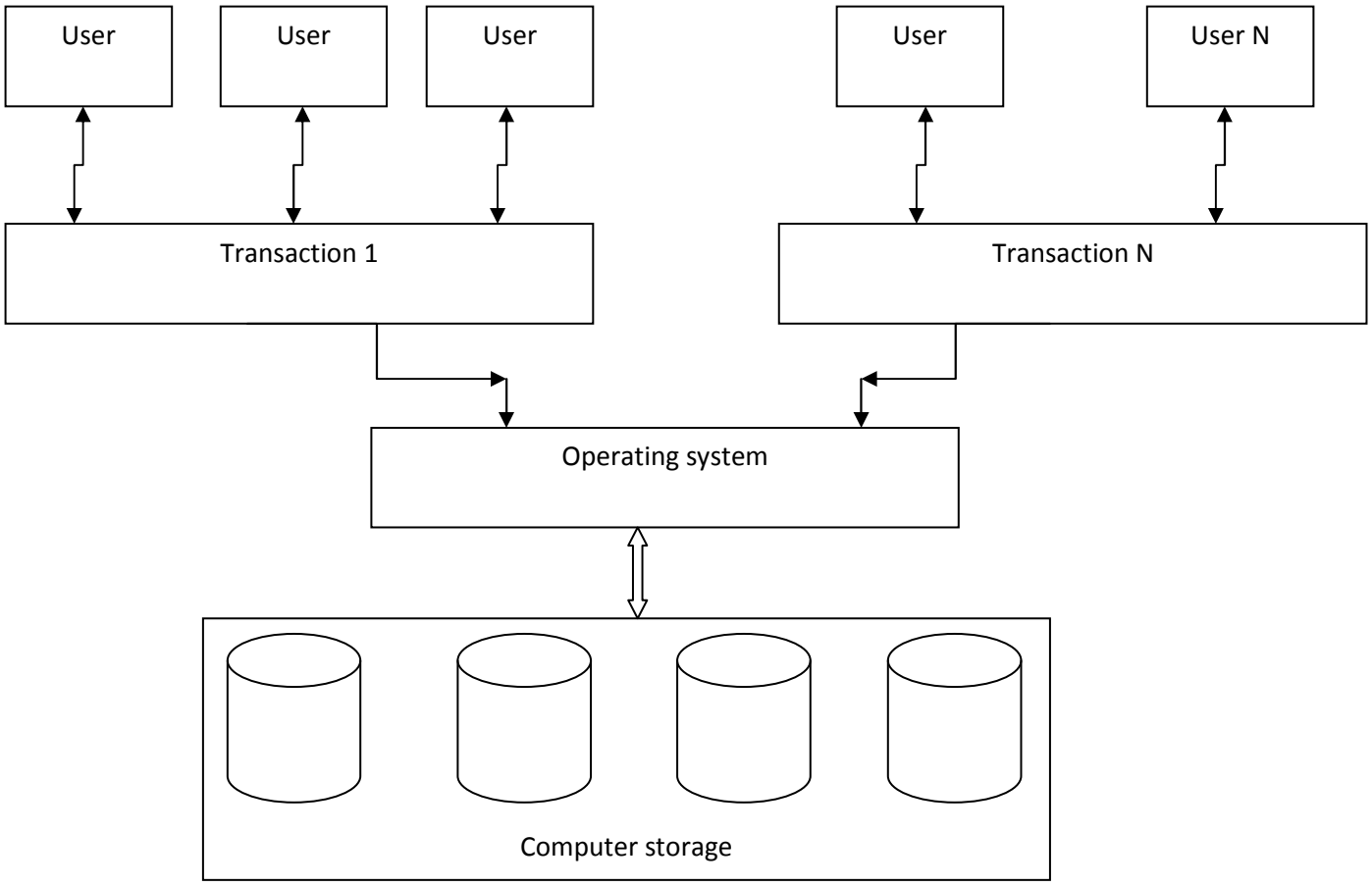
- Entity type
- Attribute type
- Relationship type

* Group of related entities is called entity type

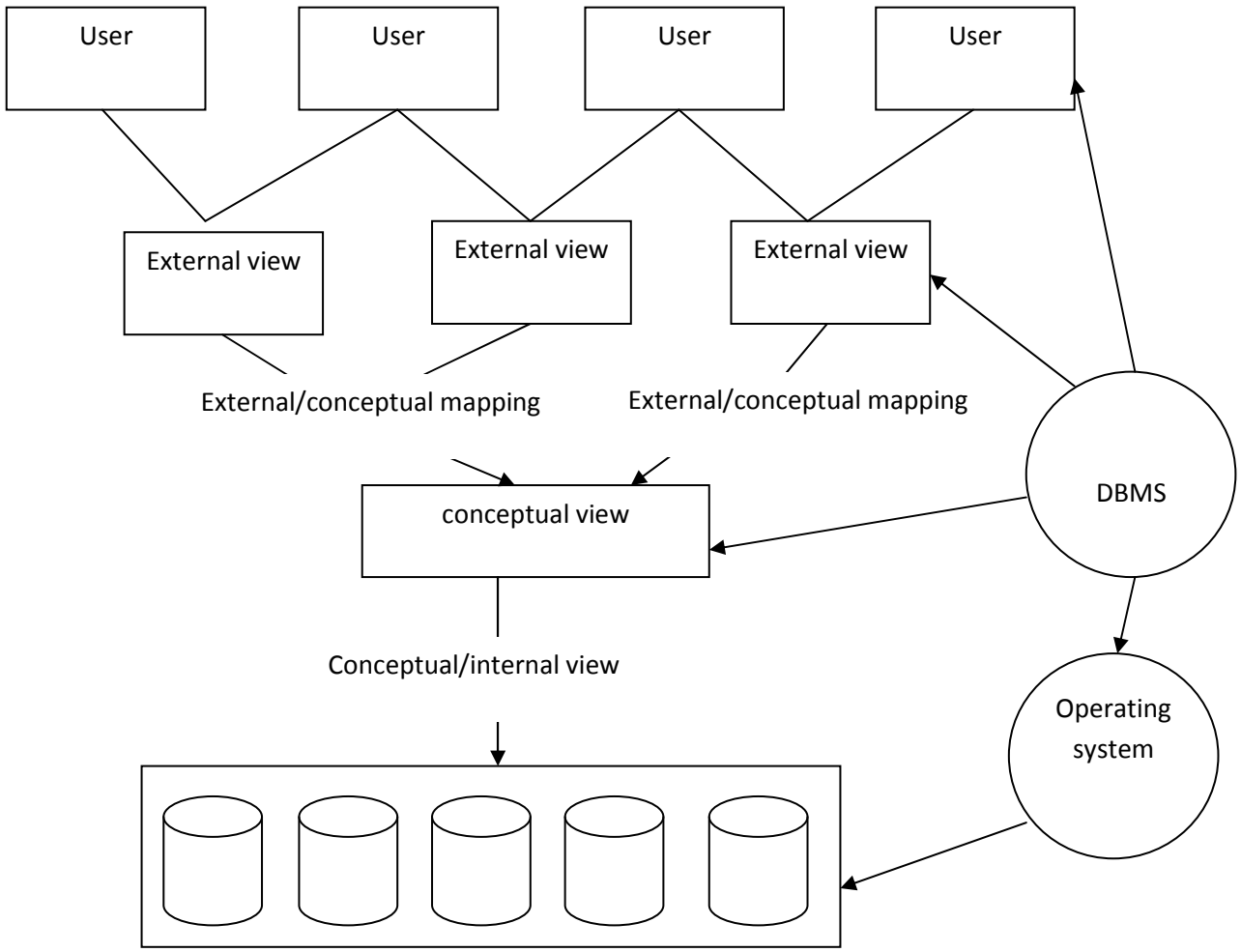
Problems of file systems environment

- Lack of data integration
- Lack of standard
- Lack of central control
- Inconsistency
- Redundancy
- Lack of multiple view

Diagram of conventional file processing system environment



Conceptual Diagram of a Centralized Database



The two major operations in database are:

- Update (insertion, deletion & modification)
- Query (to extract required records that satisfied a given condition)

Data abstraction methods

- Classification
- Aggregation
- Association
- Generalization

Database users include

- Database administrator
- End users
- Application programs

Database organization

- Physical organization (storage view)
- Logical organization (conceptual view)
- Manipulation structure (external view)

WEEK 2

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ The concept of record, field, character, byte and bits in relation to a file
- ✓ The seek, read, write, fetch, insert, delete and update operations

FILE PROCESSING OPERATIONS

A **computer file** is a block of arbitrary information, or resource for storing information, which is available to a computer program and is usually based on some kind of durable storage. A file is durable in the sense that it remains available for programs to use after the current program has finished. Computer files can be considered as the modern counterpart of paper documents which traditionally were kept in offices' and libraries' files, which are the source of the term.

The file processing operations deal with the various activities which are performed on the file. These operations are briefly described as shown below;

- File creation: The process of bringing file into existence is called file creation.
- Searching: Searching is locating data in a file by reference to a special field of each record/data called the **key**. The key is a unique filed used to identify certain record in a file. If a record is to be inserted into a file, it must be given a unique key value.
- Retrieving/reading: This involves reading an existing data from a form of storage or input medium.
- Writing: Writing is the act of recording data onto some form of storage.
- Deleting: This means removing a record or item of data from a storage medium such as disk/tape.
- File updating: This is an act of changing values in one or more records of a file without changing the organization of the file. That is making the file modern by adding most recent data to the file.
- Sorting: Sorting means rearranging data in either ascending or descending order. It involves the arrangement of grouped data elements into a predetermined sequence to facilitate file processing.

- Calculating: The arithmetic or logical manipulation of data in a file is referred to as calculation.
- File querying/interrogating: This is retrieving specific data from a file according to the set of retrieval criteria.
- File merging: Combining multiple sets of data files or records to produce only one set, usually in an ordered sequence is referred to as file merging.
- Reporting: Reporting is a file processing operation that deals with the production (printing) of report from the file in a specified format.
- File display: The contents of a data file can be displayed either on the computer screen as soft copy or printed on the paper as hard copy.
- File storage: When a file is created, it is stored in the appropriate storage medium such as disk, flash disk, tape, drum, etc.

Data Processing

- Data Processing is the analysis and organization of data by the repeated use of one or more computer programs. Data processing is used extensively in business, engineering, and science and to an increasing extent in nearly all areas in which computers are used. Businesses use data processing for such tasks as payroll preparation, accounting, record keeping, inventory control, sales analysis, and the processing of bank and credit card account statements. Engineers and scientists use data processing for a wide variety of applications, including the processing of seismic data for oil and mineral exploration, the analysis of new product designs, the processing of satellite imagery, and the analysis of data from scientific experiments.
- Data processing is divided into two kinds of processing: database processing and transaction processing. A database is a collection of common records that can be searched, accessed, and modified, such as bank account records, school transcripts, and income tax data. In database processing, a computerized

database is used as the central source of reference data for the computations. Transaction processing refers to interaction between two computers in which one computer initiates a transaction and another computer provides the first with the data or computation required for that function.

- Most modern data processing uses one or more databases at one or more central sites. Transaction processing is used to access and update the databases when users need to immediately view or add information; other data processing programs are used at regular intervals to provide summary reports of activity and database status. Examples of systems that involve all of these functions are automated teller machines, credit sales terminals, and airline reservation systems.

THE DATA-PROCESSING CYCLE

- The data-processing cycle represents the chain of processing events in most data-processing applications. It consists of data recording, transmission, reporting, storage, and retrieval. The original data is first recorded in a form readable by a computer. This can be accomplished in several ways: by manually entering information into some form of computer memory using a keyboard, by using a sensor to transfer data onto a magnetic tape or floppy disk, by filling in ovals on a computer-readable paper form, or by swiping a credit card through a reader. The data are then transmitted to a computer that performs the data-processing functions. This step may involve physically moving the recorded data to the computer or transmitting it electronically over telephone lines or the Internet. *See* Information Storage and Retrieval.
- Once the data reach the computer's memory, the computer processes it. The operations the computer performs can include accessing and updating a database and creating or modifying statistical information. After processing the data, the computer reports summary results to the program's operator.
- As the computer processes the data, it stores both the modifications and the original data. This storage can be both in the original data-entry form and in

carefully controlled computer data forms such as magnetic tape. Data are often stored in more than one place for both legal and practical reasons. Computer systems can malfunction and lose all stored data, and the original data may be needed to recreate the database as it existed before the crash.

- The final step in the data-processing cycle is the retrieval of stored information at a later time. This is usually done to access records contained in a database, to apply new data-processing functions to the data, or—in the event that some part of the data has been lost—to recreate portions of a database. Examples of data retrieval in the data-processing cycle include the analysis of store sales receipts to reveal new customer spending patterns and the application of new processing techniques to seismic data to locate oil or mineral fields that were previously overlooked.

The History of Data Processing

- To a large extent, data processing has been the driving force behind the creation and growth of the computer industry. In fact, it predates electronic computers by almost 60 years. The need to collect and analyze census data became such an overwhelming task for the United States government that in 1890 the U.S. Census Bureau contracted American engineer and inventor Herman Hollerith to build a special purpose data-processing system. With this system, census takers recorded data by punching holes in a paper card the size of a dollar bill. These cards were then forwarded to a census office, where mechanical card readers were used to read the holes in each card and mechanical adding machines were used to tabulate the results. In 1896 Hollerith founded the Tabulating Machine Company, which later merged with several other companies and eventually became International Business Machines Corporation (IBM).
- During World War II (1939-1945) scientists developed a variety of computers designed for specific data-processing functions. The Harvard Mark I computer was built from a combination of mechanical and electrical devices and was used to perform calculations for the U.S. Navy. Another computer, the British-built

Colossus, was an all-electronic computing machine designed to break German coded messages. It enabled the British to crack German codes quickly and efficiently.

- The role of the electronic computer in data processing began in 1946 with the introduction of the ENIAC, the first all-electronic computer. The U.S. armed services used the ENIAC to tabulate the paths of artillery shells and missiles. In 1950 Remington Rand Corporation introduced the first nonmilitary electronic programmable computer for data processing. This computer, called the UNIVAC, was initially sold to the U.S. Census Bureau in 1951; several others were eventually sold to other government agencies.
- With the purchase of a UNIVAC computer in 1954, General Electric Company became the first private firm to own a computer, soon followed by Du Pont Company, Metropolitan Life, and United States Steel Corporation. All of these companies used the UNIVAC for commercial data-processing applications. The primary advantages of this machine were its programmability, its high-speed arithmetic capabilities, and its ability to store and process large business files on multiple magnetic tapes. The UNIVAC gained national attention in 1952, when the American Broadcast Company (ABC) used a UNIVAC during a live television broadcast to predict the outcome of the presidential election. Based upon less than 10 percent of the election returns, the computer correctly predicted a landslide victory for Dwight D. Eisenhower over his challenger, Adlai E. Stevenson.
- In 1953, IBM produced the first of its computers, the IBM 701—-a machine designed to be mass-produced and easily installed in a customer's building. The success of the 701 led IBM to manufacture many other machines for commercial data processing. The sales of IBM's 650 computer were a particularly good indicator of how rapidly the business world accepted electronic data processing. Initial sales forecasts were extremely low because the machine was thought to be too expensive, but over 1800 were eventually made and sold.
- In the 1950s and early 1960s data processing was essentially split into two distinct areas, business data processing and scientific data processing, with

different computers designed for each. In an attempt to keep data processing as similar to standard accounting as possible, business computers had arithmetic circuits that did computations on strings of decimal digits (numbers with digits that range from 0 to 9). Computers used for scientific data processing sacrificed the easy-to-use decimal number system for the more efficient binary number system in their arithmetic circuitry.

- The need for separate business and scientific computing systems changed with the introduction of the IBM System/360 family of machines in 1964. These machines could all run the same data-processing programs, but at different speeds. They could also perform either the digit-by-digit math favored by business or the binary notation favored for scientific applications. Several models had special modes in which they could execute programs from earlier IBM computers, especially the popular IBM 1401. From that time on, almost all commercial computers were general-purpose.
- One notable exception to the trend of general-purpose computers and programming languages is the supercomputer. Supercomputers are computers designed for high-speed precision scientific computations. However, supercomputers are sometimes used for data processing that is not scientific. In these cases, they must be built so that they are flexible enough to allow other types of computations.

The division between business and scientific data processing also influenced the development of programming languages in which application programs were written. Two such languages that are still popular today are COBOL (COmmon Business Oriented Language) and Fortran (FORmula TRANslation). Both of these programming languages were developed in the late 1950s and early 1960s, with COBOL becoming the programming language of choice for business data processing and FORTRAN for scientific processing. In the 1970s other languages such as C were developed. These languages reflected the general-purpose nature of modern computers and allowed extremely efficient programs to be developed for almost any data-processing application. One of the most popular languages

currently used in data-processing applications is an extension of C called C++. C++ was developed in the 1980s and is an object-oriented language, a type of language that gives programmers more flexibility in developing sophisticated applications than other types of programming languages.

WEEK 3

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ Qualitatively file system performance in terms of fetch, insert, update and reorganization.

File systems and file managers

A file system provides a mapping between the logical and physical views of a file, through a set of services and an interface. Simply put, the file system hides all the device-specific aspects of file manipulation from users. The basic services of a file system include;

- Keeping tract of file (knowing location)
- I/O support, especially the transmission mechanism to and from main memory
- Managing of secondary storage
- Sharing of I/O devices
- Providing protection mechanisms for information held in the system

- The way a computer organizes, names, stores and manipulates files is globally referred to as its *file system.* Most computers have at least one file system. Some computers allow the use of several different file systems. For instance, on newer MS Windows computers, the older FAT-type file systems of MS-DOS and old versions of Windows are supported, in addition to the NTFS file system that is the normal file system for recent versions of Windows. Each system has its own advantages and disadvantages. Standard FAT allow only eight-character file names (plus a three-character extension) with no spaces, for example, whereas NTFS allows much longer names that can contain spaces. You can call a file **Payroll records** in NTFS, but in FAT you would be restricted to something like **payroll.dat** (unless you were using VFAT, a FAT extension allowing long file names).
- File manager programs are utility programs that allow you to manipulate files directly. They allow you to move, create, delete and rename files and folders, although they do not actually allow you to read the contents of a

file or store information in it. Every computer system provides at least one file-manager program for its native file system. Under Windows, the most commonly used file manager program is Windows Explore.r

File system

- In computing, a **file system** (often also written as **filesystem**) is a method for storing and organizing computer files and the data they contain to make it easy to find and access them. File systems may use a data storage device such as a hard disk or CD-ROM and involve maintaining the physical location of the files, they might provide access to data on a file server by acting as clients for a network protocol (e.g., NFS, SMB, or 9P clients), or they may be virtual and exist only as an access method for virtual data (e.g., procfs).
- More formally, a file system is a special-purpose database for the storage, hierarchical organization, manipulation, navigation, access, and retrieval of data.

Aspects of file systems

The most familiar file systems make use of an underlying data storage device that offers access to an array of fixed-size blocks, sometimes called *sectors*, generally a power of 2 in size (512 bytes or 1, 2, or 4 KiB are most common). The file system software is responsible for organizing these sectors into files and directories, and keeping track of which sectors belong to which file and which are not being used. Most file systems address data in fixed-sized units called "clusters" or "blocks" which contain a certain number of disk sectors (usually 1-64). This is the smallest logical amount of disk space that can be allocated to hold a file.

However, file systems need not make use of a storage device at all. A file system can be used to organize and represent access to any data, whether it be stored or dynamically generated (e.g., procfs).

File names

Whether the file system has an underlying storage device or not, file systems typically have directories which associate **file names** with files, usually by connecting the file name to an index in a file allocation table of some sort, such as the FAT in a DOS file system, or an inode in a Unix-like file system. Directory structures may be flat, or allow hierarchies where directories may contain subdirectories. In some file systems, file names are structured, with special syntax for filename extensions and version numbers. In others, file names are simple strings, and per-file metadata is stored elsewhere.

Metadata

Other bookkeeping information is typically associated with each file within a file system. The length of the data contained in a file may be stored as the number of blocks allocated for the file or as an exact byte count. The time that the file was last modified may be stored as the file's timestamp. Some file systems also store the file creation time, the time it was last accessed, and the time that the file's meta-data was changed. (Note that many early PC operating systems did not keep track of file times.) Other information can include the file's device type (e.g., block, character, socket, subdirectory, etc.), its owner user-ID and group-ID, and its access permission settings (e.g., whether the file is read-only, executable, etc.).

Arbitrary attributes can be associated on advanced file systems, such as XFS, ext2/ext3, some versions of UFS, and HFS+, using extended file attributes. This

feature is implemented in the kernels of Linux, FreeBSD and Mac OS X operating systems, and allows metadata to be associated with the file at the *file system* level. This, for example, could be the author of a document, the character encoding of a plain-text document, or a checksum.

Hierarchical file systems

The hierarchical file system was an early research interest of Dennis Ritchie of Unix fame; previous implementations were restricted to only a few levels, notably the IBM implementations, even of their early databases like IMS. After the success of Unix, Ritchie extended the file system concept to every object in his later operating system developments, such as Plan 9 and Inferno.

Facilities

Traditional file systems offer facilities to create, move and delete both files and directories. They lack facilities to create additional links to a directory (hard links in Unix), rename parent links (".." in Unix-like OS), and create bidirectional links to files.

Traditional file systems also offer facilities to truncate, append to, create, move, delete and in-place modify files. They do not offer facilities to prepend to or truncate from the beginning of a file, let alone arbitrary insertion into or deletion from a file. The operations provided are highly asymmetric and lack the generality to be useful in unexpected contexts. For example, interprocess pipes in Unix have to be implemented outside of the file system because the pipes concept does not offer truncation from the beginning of files.

Secure access

Secure access to basic file system operations can be based on a scheme of access control lists or capabilities. Research has shown access control lists to be difficult to secure properly, which is why research operating systems tend to use capabilities. Commercial file systems still use access control lists. *see: secure computing*

Types of file systems

File system types can be classified into disk file systems, network file systems and special purpose file systems.

Disk file systems

A *disk file system* is a file system designed for the storage of files on a data storage device, most commonly a disk drive, which might be directly or indirectly connected to the computer. Examples of disk file systems include FAT (FAT12, FAT16, FAT32), NTFS, HFS and HFS+, ext2, ext3, ISO 9660, ODS-5, and UDF. Some disk file systems are journaling file systems or versioning file systems.

Flash file systems

A *flash file system* is a file system designed for storing files on flash memory devices. These are becoming more prevalent as the number of mobile devices is increasing, and the capacity of flash memories catches up with hard drives.

While a block device layer can emulate a disk drive so that a disk file system can be used on a flash device, this is suboptimal for several reasons:

- Erasing blocks: Flash memory blocks have to be explicitly erased before they can be written to. The time taken to erase blocks can be significant, thus it is beneficial to erase unused blocks while the device is idle.

- Random access: Disk file systems are optimized to avoid disk seeks whenever possible, due to the high cost of seeking. Flash memory devices impose no seek latency.
- Wear levelling: Flash memory devices tend to wear out when a single block is repeatedly overwritten; flash file systems are designed to spread out writes evenly.

Log-structured file systems have all the desirable properties for a flash file system. Such file systems include JFFS2 and YAFFS.

Database file systems

A new concept for file management is the concept of a database-based file system. Instead of, or in addition to, hierarchical structured management, files are identified by their characteristics, like type of file, topic, author, or similar metadata.

Transactional file systems

Each disk operation may involve changes to a number of different files and disk structures. In many cases, these changes are related, meaning that it is important that they all be executed at the same time. Take for example a bank sending another bank some money electronically. The bank's computer will "send" the transfer instruction to the other bank and also update its own records to indicate the transfer has occurred. If for some reason the computer crashes before it has had a chance to update its own records, then on reset, there will be no record of the transfer but the bank will be missing some money.

Transaction processing introduces the guarantee that at any point while it is running, a transaction can either be finished completely or reverted completely (though not necessarily both at any given point). This means that if there is a

crash or power failure, after recovery, the stored state will be consistent. (Either the money will be transferred or it will not be transferred, but it won't ever go missing "in transit".)

This type of file system is designed to be fault tolerant, but may incur additional overhead to do so.

Journaling file systems are one technique used to introduce transaction-level consistency to filesystem structures.

Network file systems

A network file system is a file system that acts as a client for a remote file access protocol, providing access to files on a server. Examples of network file systems include clients for the NFS, AFS, SMB protocols, and file-system-like clients for FTP and WebDAV.

Special purpose file systems

A special purpose file system is basically any file system that is not a disk file system or network file system. This includes systems where the files are arranged dynamically by software, intended for such purposes as communication between computer processes or temporary file space.

Special purpose file systems are most commonly used by file-centric operating systems such as Unix. Examples include the procfs (/proc) file system used by some Unix variants, which grants access to information about processes and other operating system features.

Deep space science exploration craft, like Voyager I & II used digital tape-based special file systems. Most modern space exploration craft like Cassini-Huygens used Real-time operating system file systems or RTOS influenced file systems.

The Mars Rovers are one such example of an RTOS file system, important in this case because they are implemented in flash memory.

Crash counting is a feature of a file system designed as an alternative to journaling. It is claimed that it maintains consistency across crashes without the code complexity of implementing journaling.[*citation needed*]

File systems and operating systems

Most operating systems provide a file system, as a file system is an integral part of any modern operating system. Early microcomputer operating systems' only real task was file management — a fact reflected in their names (see DOS). Some early operating systems had a separate component for handling file systems which was called a disk operating system. On some microcomputers, the disk operating system was loaded separately from the rest of the operating system. On early operating systems, there was usually support for only one, native, unnamed file system; for example, CP/M supports only its own file system, which might be called "CP/M file system" if needed, but which didn't bear any official name at all.

Because of this, there needs to be an interface provided by the operating system software between the user and the file system. This interface can be textual (such as provided by a command line interface, such as the Unix shell, or OpenVMS DCL) or graphical (such as provided by a graphical user interface, such as file browsers). If graphical, the metaphor of the *folder*, containing documents, other files, and nested folders is often used (see also: directory and folder).

Flat file systems

In a flat file system, there are no subdirectories—everything is stored at the same (root) level on the media, be it a hard disk, floppy disk, etc. While simple, this system rapidly becomes inefficient as the number of files grows, and makes it difficult for users to organize data into related groups.

Like many small systems before it, the original Apple Macintosh featured a flat file system, called Macintosh File System. Its version of Mac OS was unusual in that the file management software (Macintosh Finder) created the illusion of a partially hierarchical filing system on top of MFS. This structure meant that every file on a disk had to have a unique name, even if it appeared to be in a separate folder. MFS was quickly replaced with Hierarchical File System, which supported real directories.

A recent addition to the flat file system family is Amazon's S3, a remote storage service, which is intentionally simplistic to allow users the ability to customize how their data is stored. The only constructs are buckets (imagine a disk drive of unlimited size) and objects (similar, but not identical to the standard concept of a file). Advance file management is allowed by being able to use nearly any character (including '/') in the objects name, and the ability to select subsets of the bucket's content based off identical prefixes.

File systems under Unix-like operating systems

Unix-like operating systems create a virtual file system, which makes all the files on all the devices appear to exist in a single hierarchy. This means, in those systems, there is one root directory, and every file existing on the system is located under it somewhere. Furthermore, the root directory does not have to be in any physical place. It might not be on your first hard drive - it might not even be on your computer. Unix-like systems can use a network shared resource as its root directory.

Unix-like systems assign a device name to each device, but this is not how the files on that device are accessed. Instead, to gain access to files on another device, you must first inform the operating system where in the directory tree you would like those files to appear. This process is called mounting a file system. For example, to access the files on a CD-ROM, one must tell the operating system "Take the file system from this CD-ROM and make it appear under such-and-such directory". The directory given to the operating system is called the *mount point* - it might, for example, be /media. The /media directory exists on many Unix systems (as specified in the Filesystem Hierarchy Standard) and is intended specifically for use as a mount point for removable media such as CDs, DVDs and like floppy disks. It may be empty, or it may contain subdirectories for mounting individual devices. Generally, only the administrator (i.e. root user) may authorize the mounting of file systems.

Unix-like operating systems often include software and tools that assist in the mounting process and provide it new functionality. Some of these strategies have been coined "auto-mounting" as a reflection of their purpose.

1. In many situations, file systems other than the root need to be available as soon as the operating system has booted. All Unix-like systems therefore provide a facility for mounting file systems at boot time. System administrators define these file systems in the configuration file fstab or vfstab in Solaris Operating Environment, which also indicates options and mount points.
2. In some situations, there is no need to mount certain file systems at boot time, although their use may be desired thereafter. There are some utilities for Unix-like systems that allow the mounting of predefined file systems upon demand.

3. Removable media have become very common with microcomputer platforms. They allow programs and data to be transferred between machines without a physical connection. Common examples include USB flash drives, CD-ROMs, and DVDs. Utilities have therefore been developed to detect the presence and availability of a medium and then mount that medium without any user intervention.
4. Progressive Unix-like systems have also introduced a concept called **supermounting**; see, for example, the Linux supermount-ng project. For example, a floppy disk that has been supermounted can be physically removed from the system. Under normal circumstances, the disk should have been synchronized and then unmounted before its removal. Provided synchronization has occurred, a different disk can be inserted into the drive. The system automatically notices that the disk has changed and updates the mount point contents to reflect the new medium. Similar functionality is found on standard Windows machines.
5. A similar innovation preferred by some users is the use of autofs, a system that, like supermounting, eliminates the need for manual mounting commands. The difference from supermount, other than compatibility in an apparent greater range of applications such as access to file systems on network servers, is that devices are mounted transparently when requests to their file systems are made, as would be appropriate for file systems on network servers, rather than relying on events such as the insertion of media, as would be appropriate for removable media.

File systems under Linux

Linux supports many different file systems, but common choices for the system disk include the ext* family (such as ext2 and ext3), XFS, JFS and ReiserFS.

File systems under Mac OS X

Mac OS X uses a file system that it inherited from classic Mac OS called HFS Plus. HFS Plus is a metadata-rich and case preserving file system. Due to the Unix roots of Mac OS X, Unix permissions were added to HFS Plus. Later versions of HFS Plus added journaling to prevent corruption of the file system structure and introduced a number of optimizations to the allocation algorithms in an attempt to defragment files automatically without requiring an external defragmenter.

Filenames can be up to 255 characters. HFS Plus uses Unicode to store filenames. On Mac OS X, the filetype can come from the type code, stored in file's metadata, or the filename.

HFS Plus has three kinds of links: Unix-style hard links, Unix-style symbolic links and aliases. Aliases are designed to maintain a link to their original file even if they are moved or renamed; they are not interpreted by the file system itself, but by the File Manager code in userland.

Mac OS X also supports the UFS file system, derived from the BSD Unix Fast File System via NeXTSTEP. However, as of Mac OS X 10.5 (Leopard), Mac OS X can no longer be installed on a UFS volume, nor can a pre-Leopard system installed on a UFS volume be upgraded to Leopard. [1]

File systems under Plan 9 from Bell Labs

Plan 9 from Bell Labs was originally designed to extend some of Unix's good points, and to introduce some new ideas of its own while fixing the shortcomings of Unix.

With respect to file systems, the Unix system of treating things as files was continued, but in Plan 9, *everything* is treated as a file, and accessed as a file would be (i.e., no ioctl or mmap). Perhaps surprisingly, while the file interface is

made universal it is also simplified considerably, for example symlinks, hard links and suid are made obsolete, and an atomic create/open operation is introduced. More importantly the set of file operations becomes well defined and subversions of this like ioctl are eliminated.

Secondly, the underlying 9P protocol was used to remove the difference between local and remote files (except for a possible difference in latency). This has the advantage that a device or devices, represented by files, on a remote computer could be used as though it were the local computer's own device(s). This means that under Plan 9, multiple file servers provide access to devices, classing them as file systems. Servers for "synthetic" file systems can also run in user space bringing many of the advantages of micro kernel systems while maintaining the simplicity of the system.

Everything on a Plan 9 system has an abstraction as a file; networking, graphics, debugging, authentication, capabilities, encryption, and other services are accessed via I-O operations on file descriptors. For example, this allows the use of the IP stack of a gateway machine without need of NAT, or provides a network-transparent window system without the need of any extra code.

Another example: a Plan-9 application receives FTP service by opening an FTP site. The ftpfs server handles the open by essentially mounting the remote FTP site as part of the local file system. With ftpfs as an intermediary, the application can now use the usual file-system operations to access the FTP site as if it were part of the local file system. A further example is the mail system which uses file servers that synthesize virtual files and directories to represent a user mailbox as /mail/fs/mbox. The wikifs provides a file system interface to a wiki.

These file systems are organized with the help of private, per-process namespaces, allowing each process to have a different view of the many file systems that provide resources in a distributed system.

The Inferno operating system shares these concepts with Plan 9.

File systems under Microsoft Windows

Windows makes use of the FAT and NTFS (New Technology File System) file systems.

The File Allocation Table (FAT) filing system, supported by all versions of Microsoft Windows, was an evolution of that used in Microsoft's earlier operating system (MS-DOS which in turn was based on 86-DOS). FAT ultimately traces its roots back to the short-lived M-DOS project and Standalone disk BASIC before it. Over the years various features have been added to it, inspired by similar features found on file systems used by operating systems such as Unix.

Older versions of the FAT file system (FAT12 and FAT16) had file name length limits, a limit on the number of entries in the root directory of the file system and had restrictions on the maximum size of FAT-formatted disks or partitions. Specifically, FAT12 and FAT16 had a limit of 8 characters for the file name, and 3 characters for the extension. This is commonly referred to as the 8.3 filename limit. VFAT, which was an extension to FAT12 and FAT16 introduced in Windows NT 3.5 and subsequently included in Windows 95, allowed long file names (LFN). FAT32 also addressed many of the limits in FAT12 and FAT16, but remains limited compared to NTFS.

NTFS, introduced with the Windows NT operating system, allowed ACL-based permission control. Hard links, multiple file streams, attribute indexing, quota

tracking, compression and mount-points for other file systems (called "junctions") are also supported, though not all these features are well-documented.

Unlike many other operating systems, Windows uses a *drive letter* abstraction at the user level to distinguish one disk or partition from another. For example, the path C:\WINDOWS represents a directory WINDOWS on the partition represented by the letter C. The C drive is most commonly used for the primary hard disk partition, on which Windows is usually installed and from which it boots. This "tradition" has become so firmly ingrained that bugs came about in older versions of Windows which made assumptions that the drive that the operating system was installed on was C. The tradition of using "C" for the drive letter can be traced to MS-DOS, where the letters A and B were reserved for up to two floppy disk drives. Network drives may also be mapped to drive letters.

Data retrieval process

The operating system calls on the IFS (Installable File System) manager. The IFS calls on the correct FSD (File System Driver) in order to open the selected file from a choice of four FSDs that work with different storage systems—NTFS, VFAT, CDFS (for optical drives), and Network. The FSD gets the location on the disk for the first cluster of the file from the FAT (File Allocation Table), FAT98, VFAT (Virtual File Allocation Table), or, in the case of Windows XP, the MFT (Master File Table). In short, the whole point of the FAT, FAT98, VFAT, or MFT is to map out all the files on the disk and record were they are located (which track and sector of the disk).

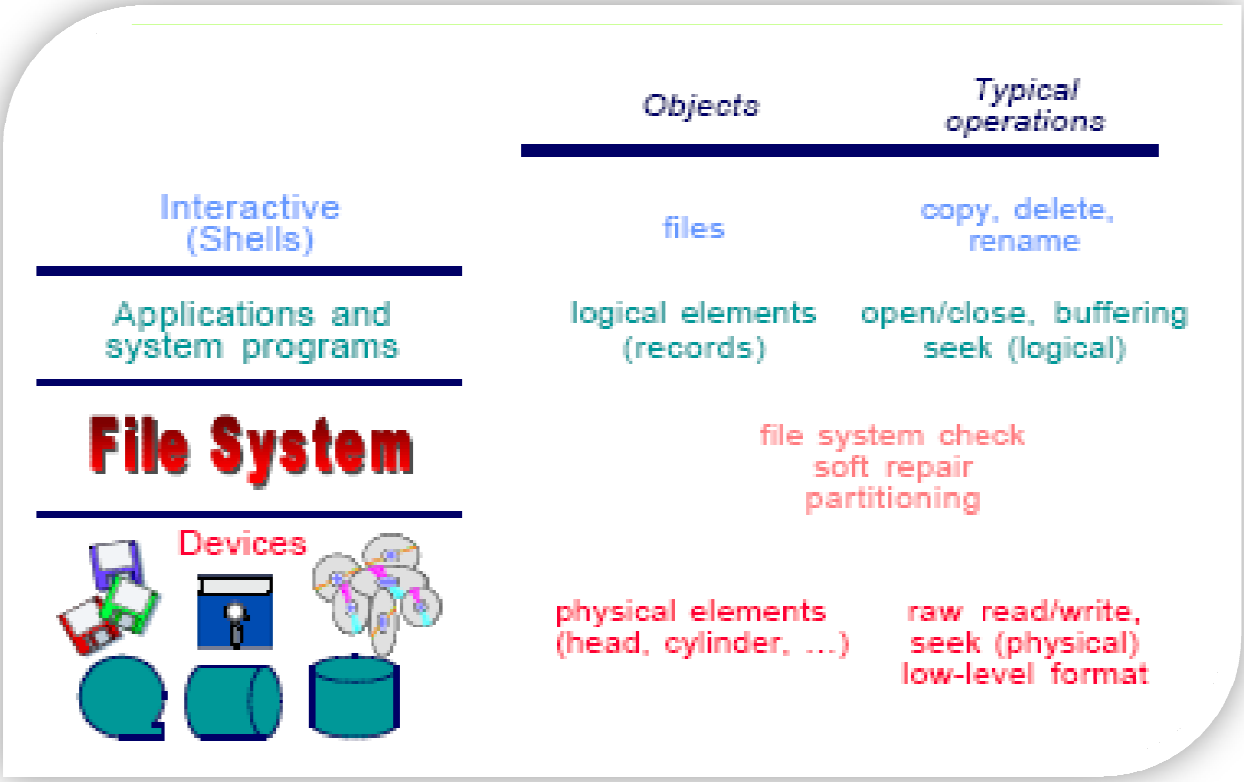
File sharing

Allowing users to share files raises a major issue of ***protection.*** A general approach is to provide controlled access to files through a set of operations

such as read, write, delete, list and append. Then permit users to perform one or more operations.

One popular protection mechanism is a condensed version of access list, where the system recognizes three classifications of users with each file and directory:

- User
- Group
- others



File system abstraction

The qualities of a good filing system

- Maintainable
- Complete

- Secure
- Accessible
- Reliable
- Readable
- Expandable
- Storage saving
- Portable, etc.

Storing files

- ✖ The discussion above describes a file as a concept presented to a user or a high-level operating system. However, any file that has any useful purpose, outside of a thought experiment, must have some physical manifestation. That is, a file (an abstract concept) in a real computer system must have a real physical analogue if it is to exist at all.
- ✖ In physical terms, most computer files are stored on some type of data storage device. For example, there is a *hard disk*, from which most operating systems run and on which most store their files. Hard discs are the most ubiquitous form of non-volatile storage at the start of the 21st century. Where files contain only temporary information, they may be stored in RAM.
- ✖ Computer files may be stored on magnetic tape. Files can also be stored on other media in some cases, such as writeable *compact discs*, *Digital Versatile Discs*, *Zip drives*, *USB flash drives*, etc

Backing up files

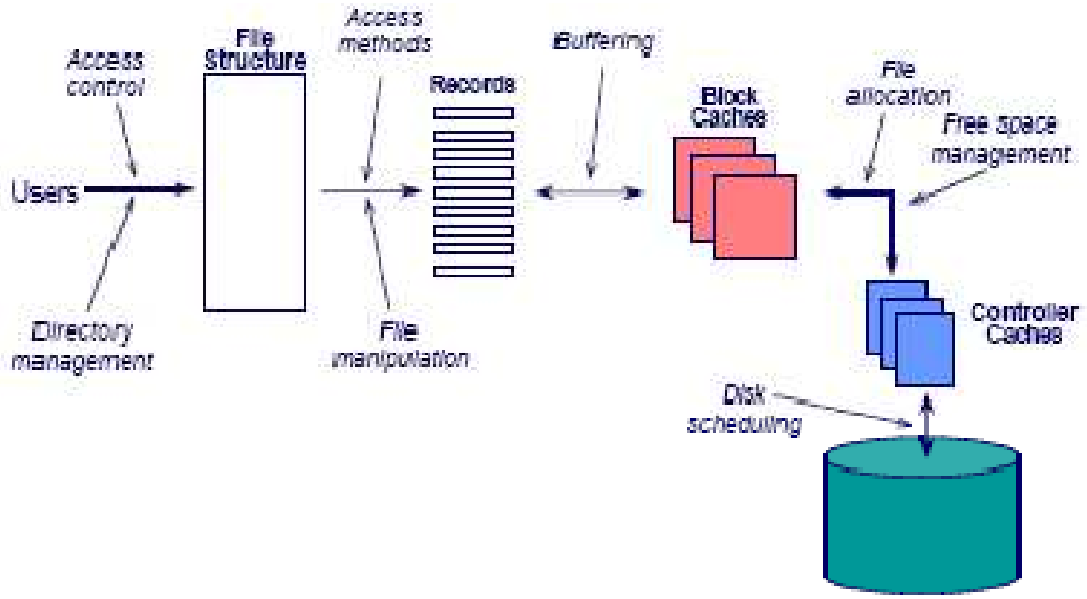
- ✖ When computer files contain information that is extremely important, a *back-up* process is used to protect against disasters that might destroy the files. Backing up files simply means making copies of the files in a separate location so that they can be restored if something happens to the computer, or if they are deleted accidentally.

- There are many ways to back up files. Most computer systems provide utility programs to assist in the back-up process, which can become very time-consuming if there are many files to safeguard. Files are often copied to removable media such as writable CDs or cartridge tapes. Copying files to another hard disk in the same computer protects against failure of one disk, but if it is necessary to protect against failure or destruction of the entire computer, then copies of the files must be made on other media that can be taken away from the computer and stored in a safe, distant location.
- The Grandfather-Father-Son method automatically makes three back ups, the grandfather file is the oldest copy of the file and the son is the current copy.

File storage media

File storage media is a device that can receive data and retain it for subsequent retrieval. Such devices cover a wide range of capacity and speed of access. The examples are;

Elements of storage management



WEEK 4

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ Different methods of file organisation in computer system
- ✓ File design alternatives

Concept of file operations

File organization method

Magnetic disk storage is available in many forms, including floppies, hard-disks, cartridge, exchangeable multi-platter, and fixed disks. The following deals with the concepts which are
applied, in many different ways, to all of the above methods.

A typical disk pack comprises of 6 disks held on a central spindle. As the top and bottom are disregarded with recording information, only 10 surfaces are used, each with 200 concentric recording tracks. A diagrammatic representation is shown in Figure 1. Each of the 200 tracks will be capable of holding equal amounts of information as this is a feature that is provided by the special software (housekeeping) that is used in conjunction with the handling of disk files. When the unit is in position in the drive, the tracks are accessed by a comb of 10 read/write heads held rigidly on an arm assembly which moves in and out between the disk as illustrated.

In this way 10 tracks may be referenced without further movement of the heads once they are positioned over the first track. For the sake of economy it is therefore usual to record over a 'cylinder' of data (see Figure 1) and avoid unnecessary movement of the heads. The disk pack is loaded onto the drive and revolves at normal speed of approximately 3600 rpm when being accessed. Each track can hold upwards of 4000 bytes. Sometimes the sectors on a disk are used in a similar manner to inter-block gaps on a tape file but in other cases they are ignored as far as data handling is concerned.

Because this device has the ability to locate an area of data directly it is known as a Direct Access Device and is popular for the versatility that this affords.

Direct access devices provide a storage medium which is a satisfactory compromise between main store and magnetic tape.

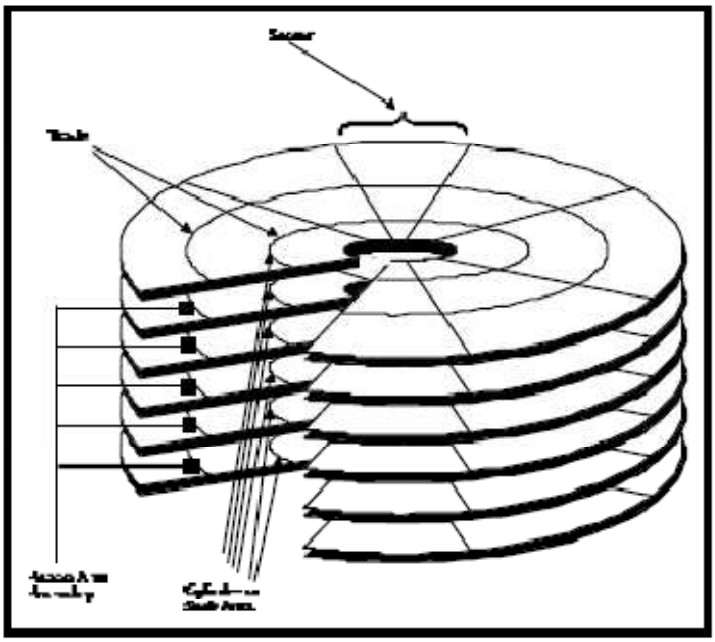


Figure 1 - Disk Pack

There are a large number of ways records can be organized on disk or tape. The main methods of file organization used for files are:

- Serial
- Sequential
- Indexed Sequential
- Random (or Direct)

a. ***Serial Organization***

Serial files are stored in chronological order, that is, as each record is received it is stored in the next available storage position. In general it is only used on a serial medium such as magnetic tape. This type of file organization means that the records are in no particular order and therefore to retrieve a single record, the

whole file needs to be read from the beginning to end. Serial organization is usually the method used for creating Transaction files (unsorted), Work and Dump files.

b. ***Sequential Organization***

Sequential files are serial files whose records are sorted and stored in an ascending or descending on a particular key field. The physical order of the records on the disk is not necessarily sequential, as most manufacturers support an organization where certain records (inserted after the file has been set up) are held in a logical sequence but are physically placed into an overflow area. They are no longer physically contiguous with the preceding and following logical records, but they can be retrieved in sequence.

c. ***Indexed Sequential Organization***

Indexed Sequential file organization is logically the same as sequential organization, but an index is built indicating the block containing the record with a given value for the Key field. This method combines the advantages of a sequential file with the possibility of direct access using the Primary Key (the primary Key is the field that is used to control the sequence of the records). These days manufacturers providing Indexed Sequential Software allow for the building of indexes using fields other than the primary Key. These additional fields on which indexes are built are called Secondary Keys.

There are three major types of indexes used:

- ***Basic Index***: This provides a location for each record (key) that exists in the system.
- ***Implicit Index***: This type of index gives a location of all possible records (keys) whether they exist or not.

- ***Limit Index***: This index groups the records (keys) and only provides the location of the highest key in the group. Generally they form a hierarchical index. Data records are blocked before being written to disk. An index may consist of the highest key in each block, (or on each track).

Index	Data	
1 A0025	A0012	
2 A0053	A0017	Block 1
3 A0075	A0025	
	A0037	
	A0038	Block 2
	A0053	
	A0064	
	A0073	Block 3
	A0075	

In the above example, data records are shown as being 3 to a block. The index, then, holds the key of the highest record in each block. (An essential element of the index, which has been omitted from the diagram for simplicity, is the physical address of the block of data records). Should we wish to access record 5, whose key is A0038, we can quickly determine from the index that the record is held m block 2, since this key is greater than the highest key in block 1, A0025, and less than the highest key in block 2, A0053. By way of the index we can go directly to the record we wish to retrieve, hence the term "direct access".

d. ***Random*** *(or Direct)*

A randomly organized file contains records arranged physically without regard to the sequence of the primary key. Records are loaded to disk by establishing a direct relationship between the Key of the record and its address on the file, normally by use of a formula (or algorithm) that converts the primary Key to a physical disk address. This relationship is also used for retrieval.

The use of a formula (or algorithm) is known as 'Key Transformation' and there are several techniques that can be used:

- Division Taking Quotient
- Division Taking Remainder
- Truncation
- Folding
- Squaring
- Radix Conversion

These methods are often mixed to produce a unique address (or location) for each record (key). The production of the same address for two different records is known as a synonym.

Random files show a distinct advantage where:

- Hit Rate is low
- Data cannot be batched or sorted
- Fast response is required.

Catering for expansion

A normal tendency of master files is to expand. Records may be increased in size or may be added. Even if the total size or number of records does not increase, there will almost inevitably be changes. Although it is usual to update

files on disk by overlay there must be provision for additions and preferably some means of re-utilizing storage arising from deletion.

Overflow arises from:

i. A record being assigned to a block that is already full.
ii. A record being expanded so that it can no longer be accommodated in the block.

There are a number of methods for catering for expansion:

i. Specifying less than 100% block packing density on initial load.
ii. Specifying less than 100% cylinder packing density.
iii. Specifying extension blocks (usually at the end of the file).

The first method is only effective and efficient if the expansion is regular. Where localized expansion occurs to any extent, even in one block, this system will fail. The other two methods have the result of allowing space for first and second level overflow, respectively. Extension blocks are used when all other overflow facilities have been exhausted. Ideally, first level overflow is situated on the same cylinder as the overflowed block hence there is no penalty incurred in terms of head movement. Second level overflow normally consists of one or more blocks at the end of the file. If a significant number of accesses to second level overflow are made, run-time will increase considerably. There is a need to reorganize the file to bring record back from overflow.

Each file organization can be accessed or processed in different ways, often combing the advantages of one organization with the advantages of another.
Summary of file organization and access methods:

ACCESS METHODS			
FILE ORGANISATION	Serial	Sequential	Random
Serial	X		
Sequential		X	
Indexed Sequential		X	X
Random	X		X

The transfer time of data from a direct storage device such as a disk drive can be calculated, however, the formulae needed for the different types of file organizations differ. Examples of these formulas are shown as following.

SEQUENTIAL FILE TRANSFER TIMINGS:

This time taken to transfer ALL records will equal

The transfer time for the file
+
Total SEEK time for the file
+
Total Latency for the file

In order to calculate the above the following are required:

1. $\frac{\text{Size of file in characters}}{\text{Transfer Rate}}$ = **Transfer time for file**

2. $\frac{\text{Bytes per Track}}{\text{Characters per record}}$ = **Number of records per track**

3. $\frac{\text{Number of Records}}{\text{Number of records per track}}$ = **Number tracks required**

4. $\frac{\text{Number of tracks required}}{\text{Number of surfaces}}$ = **Number of cylinders required**

5. Number of cylinders **X** Minimum SEEK Time = **Total seek time for file**

6. Number of cylinders **X** Average Latency = **Total Latency for file**

For Ransom or Direct file organizations both the SEEK time and Latency between each record transferred need to be included in the calculation:

1. $\frac{\text{Size of file in Characters}}{\text{Transfer Rate}}$ = **Transfer time for file**

2. Number of records in file X Average SEEK time = **Total SEEK time for file**

3. Number of Records in file X Average Latency = **Total Latency for file**

The sum of these three calculations will give the transfer time for All records.

In order to calculate the Latency (Rotational Delay) the time for one rotation of the disk needs to be expressed in milliseconds.

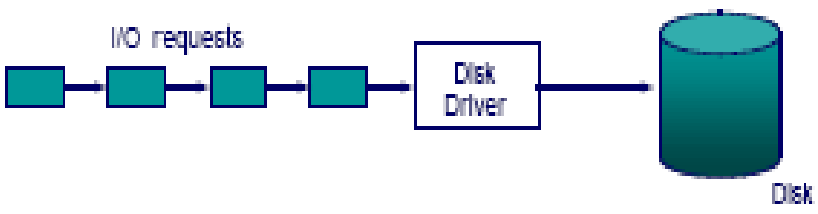
$\frac{\text{RPM}}{60}$ = **R.P.Seconds**

$\frac{1}{\text{R.P.Seconds}}$ = **Latency express as M. seconds**

Access time = **SEEK + Average Latency.**

Disk scheduling

In multiprogramming systems, there may be several disk I/O requests at the same time. As a result, a disk driver is typically faced with a pool of I/O requests:



The most costly component of a disk I/O operation is the ***seek time***. By scheduling multiple disk requests, the total seek time can be reduced. For example, shortest seek time first.

Disk scheduling strategies

Commonly used strategies include (in addition to some common CPU scheduling policies):

- *First Come First Served (FCFS)* or FIFO
- *Shortest Service Time First (SSTF)*
- *SCAN*—back and forth over disk
- *C-SCAN*—circular SCAN or one way SCAN and fast return
- *LOOK*—look for a request before moving in that direction
- *C-LOOK*—circular LOOK

File allocation

The file system allocates disk space, when a file is created. With many files residing on the same disk, the main problem is how to allocate space for them.
File allocation scheme has impact on the efficient use of disk space and file access time.

Common file allocation techniques are:

- Contiguous
- Chained (linked)
- Indexed

All these techniques allocate disk space on a per block (smallest addressable disk unit) basis.

Contiguous allocation

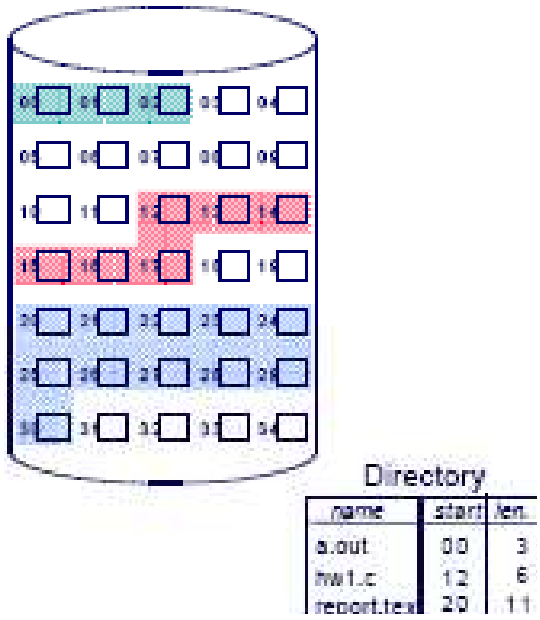
Allocate disk space like paged, segmented memory. Keep a free list of unused disk space.

***Advantages*:**

• easy access, both sequential and random

• simple

• few seeks



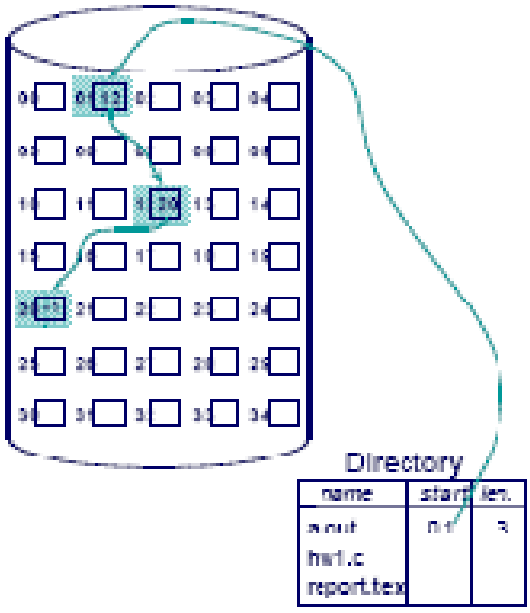
Disadvantages:

• external fragmentation

• may not know the file size in advance

Chained (linked) allocation

Space allocation is similar to page frame allocation. Mark allocated blocks as in-use.



***Advantages*:**

• no external fragmentation

• files can grow easily

***Disadvantages*:**

• lots of seeking

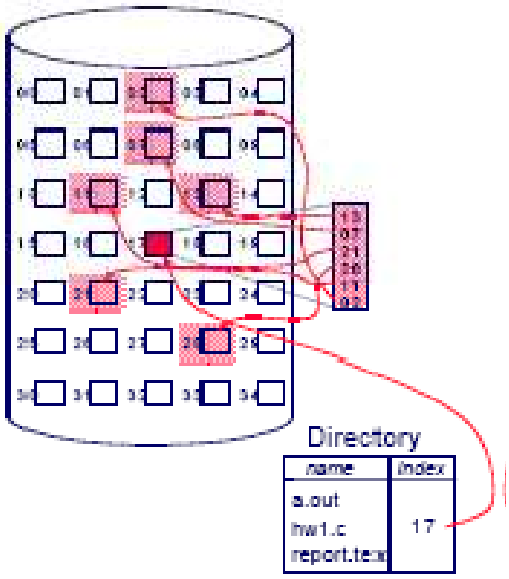
• random access difficult

Example:

MSDOS (FAT) file system.

Indexed allocation

Allocate an array of pointers during file creation. Fill the array as new disk blocks are assigned.



***Advantages*:**

• small internal fragmentation

• easy sequential and direct access

***Disadvantages*:**

• lots of seeks if the file is big

• maximum file size is limited to the size of a block

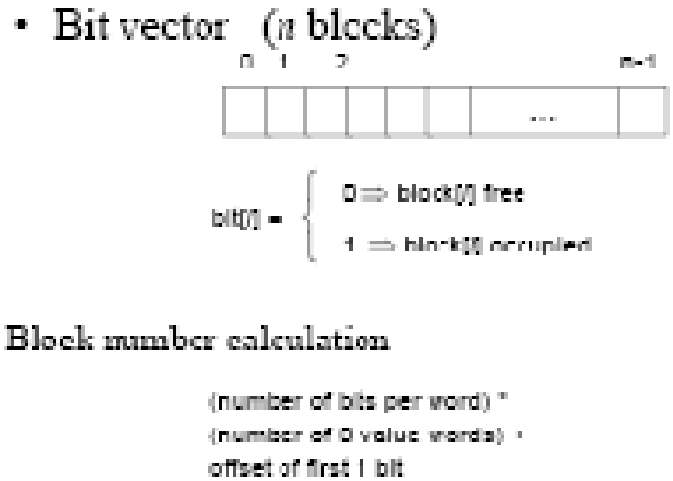
Example:

UNIX file system.

Free space management

Since the amount of disk space is limited (posing a management problem similar to that of physical memory), it is necessary to reuse the space released by deleted files. In general, file systems keep a list of *free* disk blocks (initially, all the blocks are free) and manage this list by one of the following techniques:

• Bit vectors



- Bit map requires extra space. Example:
  block size = 2^{12} bytes
  disk size = 2^{30} bytes (1 gigabyte)
  $n = 2^{30}/2^{12} = 2^{18}$ bits (or 32K bytes)
- Easy to get contiguous files
- Linked list (free list)
  - Cannot get contiguous space easily
  - No waste of space
- Grouping
- Counting

• Need to protect:

– Pointer to free list

– Bit map

• Must be kept on disk

• Copy in memory and disk may differ.

• Cannot allow for block[*i*] to have a situation where bit[*i*] = 1 in memory and bit[*i*] = 0 on disk.

– Solution:

• Set bit[*i*] = 1 in disk.

• Allocate block[*i*]

• Set bit[*i*] = 1 in memory

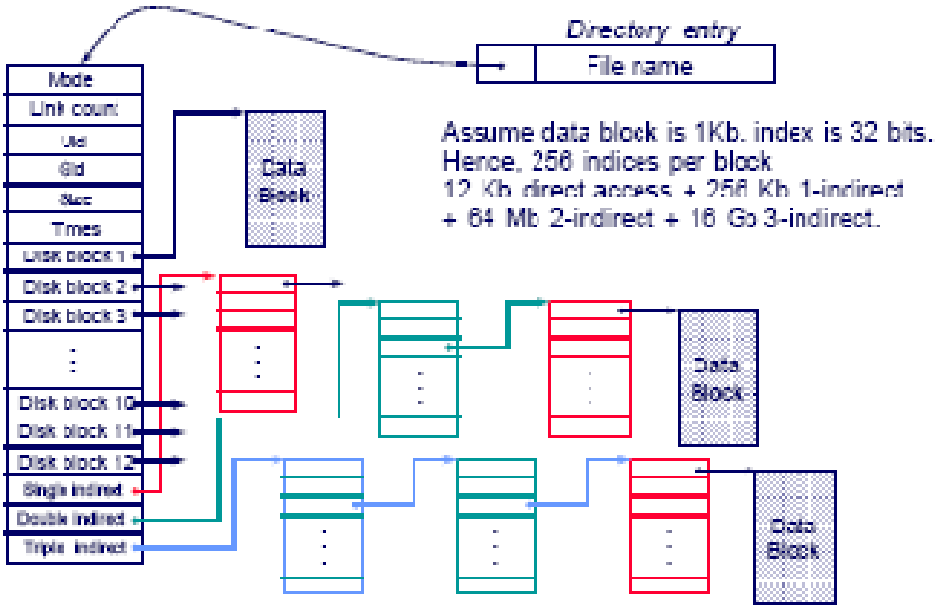
• Linked lists or chains

– single list of a set of free block lists

• Indexing

– single level, multiple levels

An example: UNIX I-nodes



Other file system issues

• *Disk blocking*

– multiple sectors per block for efficiency

• *Disk quotas*

• *Reliability*

– Backup/restore (disaster scenarios)

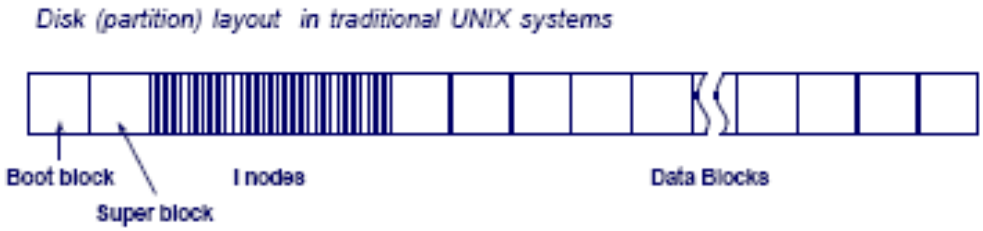
– File system (consistency) check (e.g., UNIX **fsck**)

• *Performance*

– Block or buffer caches (a collection of blocks kept in memory)

Case study—UNIX file system

The boot block usually contains (bootstrap) code to boot the system. The super block contains critical information about the layout of the file system, such as number of I-nodes and the number of disk blocks. Each I-node entry contains the file attributes, except the name. The first I-node points to the block containing the root directory of the file system.



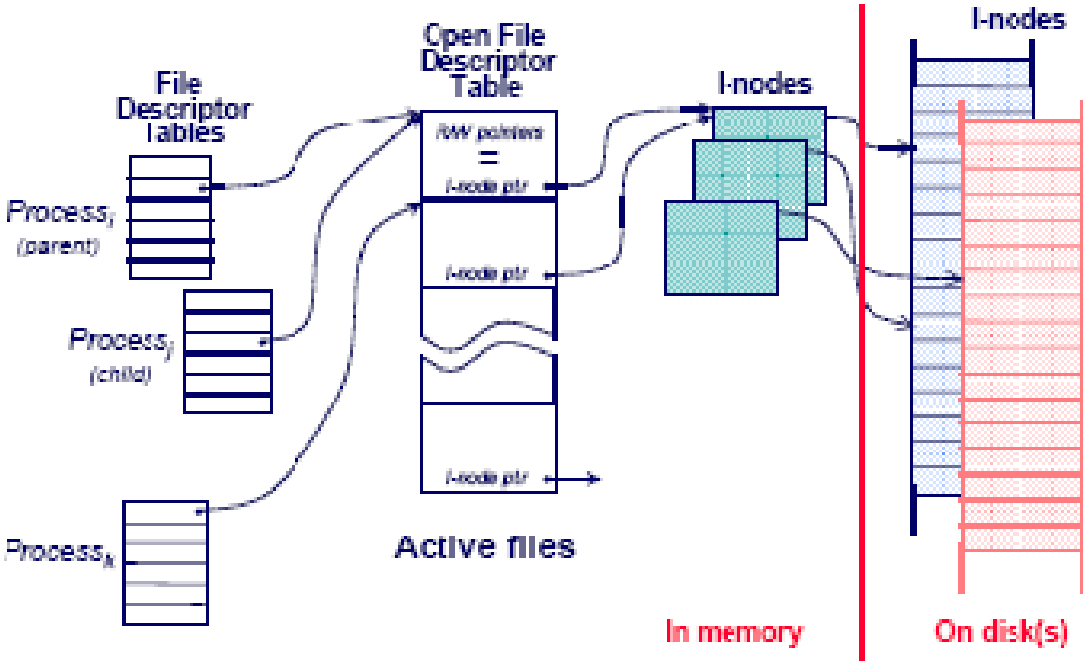
There are three different in direction to access a file:

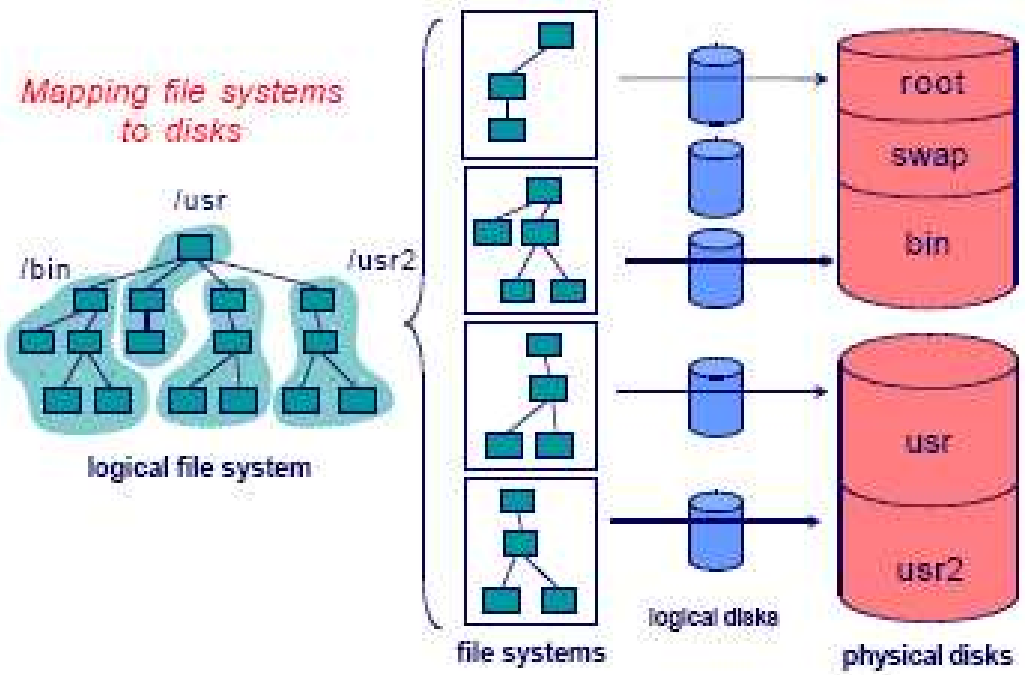
• *File Descriptor Table:* one per process, keeping track of open files.

• *Open File Table*: one per system, keeping track of all the files currently open.

• *I-node Table:* one per system (disk volume or partition) keeping track all files.

Directories are stored just like ordinary files. User programs can read directories, but special care is needed to write a directory. Each directory contains *<file name, I-node number>* pairs. *Root (i.e., /)* is a special directory with no name.





Efficiency and Performance

• Efficiency dependent on:

– disk allocation and directory algorithms

– types of data kept in file's directory entry

• Performance

– disk cache – separate section of main memory for frequently sued blocks

– free-behind and read-ahead – techniques to optimize sequential access

– improve PC performance by dedicating section of memory as virtual disk, or RAM disk.

Recovery technique

• Consistency checker – compares data in directory structure with data blocks on disk, and tries to fix inconsistencies.

• Use system programs to *back up* data from disk to another storage device (floppy disk,

magnetic tape)

• Recover lost file or disk by *restoring* data from backup.

WEEK 5

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ Different methods of file organization in computer system
- ✓ Activity ratio and hit rate.
- ✓ Different types of files: Master file, Transaction file, Reference file, etc.
- ✓ The concept of master file, transaction file and activity file.

File access methods

Algorithm used for the storage and/or retrieval of records to/from a data file is referred to as file access method. This term is used interchangeably (and loosely) for both the access method and file organization. Access methods are of two kinds; those that determine the structural characteristics of the file on which it is used (file organization) and those that do not, as in secondary indexing. In the case of the first one the same algorithm is used for the retrieval of records as was used for its original physical placement/organization. File access method include the following;

i. **Serial access method**: This is method of accessing records in an unsorted file. The term refers to the ability of accessing each record in a file by a sequential searching process, i.e from the first record to the last. One must finish before the next begins, i.e one after another. This access method is associated with time consuming.

ii. **Sequential access method**: A method of access to a data file where each record is sequenced in some order. A file is said to be sequentially accessed if the sequence of transaction presented to it matches a sequence in which records are organized.

iii. **Indexed sequential access method**: This is an access method for a sequentially organized file whose records are indexed with their corresponding address. This access method supports both sequential access and indexed access and the two types of access that can be comfortably done in this method are sequential access and random access. Note that this method is only possible with disk files.

    a. **Sequential access**: The access of the index sequential file makes minimal use of the index. The records can be accessed serially as they are ordered one after another and from the first to the last record in the file.

b. **Random access:** The random access of the index sequential makes full use of the index to access the records using the record address associated with the index. The records are accessed without any defined sequence of sorting. The method is used when the transaction is to be processed immediately without waiting for sorting and its widely used in commercial data processing because it can be allow both sequential and random methods of access.

iv. **Random access method:** Random access is a type of memory access in which storage location can be accessed in any order. This means that a file is said to be randomly accessed if the sequence of transactions submit to it does not match any sequence in which records may be organized.

Summary of file access methods

The information stored in a file can be accessed in a variety of methods:

- *Sequential*: in order, one record after another.
- *Direct (random)*: in any order, skipping the previous records.
- Indexed or *Keyed*: in any order, but with particular value(s).

Other access methods, such as indexed, can be built on top of the above basic techniques. Indexed sequential access method (ISAM) is built on random and sequential access.

Types of files

Master file: A master file consists of data fields which are of a permanent nature. The values of these fields must continually be brought up-to-date so that the file will always contain the most recent transaction or affairs in the

organization. For instance, an employee file is made of records whose fields may include; Employee Number, Name, Date of birth, Qualification, Salary grade, etc. These fields are permanent, although, Qualification and Salary grade might need to be updated at a future time/date.

Transaction file: Transaction file is related to the activities within an organization. This file contains operational data extracted from the transaction of the organization. For example, if a new staff is employed or an one is promoted, the data is transaction data and when such records are collected, a transaction file is obtained. A transaction file is often used to update the master file. Once it has played this role, it is no longer needed. It may however be retained or archived for security control purposes. Examples of transaction file include; invoices, hour worked, new qualifications, promotion, electricity meter reading, etc.

Work file: The work file is derived from the either the master or transaction files. They are often temporal in nature as they may not be needed/required if serve the purpose. For example, a personnel file can be interrogated to display the records of employees who meet certain criteria, and when the records are extracted from the master file, they are kept in a temporary work file before they are printed. In most cases, work files are created to hold the sorted records.

Reference file: A reference file is a file which contains records that may be used in the future or for reference purposes. Reference file is also called standing file whose records are responsibly permanent.

Archive file: This file is also referred to as historical file as it contains old files/records which are currently not useful or no longer useful. For instance, the files containing particulars of former clients, records of graduated students in an institution, etc.

Data file: A file containing data/value, such as a file created within an application programs. Data files are normally organized as sets of records with one or more associated access methods. For example, it may be a word processing document, a spreadsheet, a database file or a chart file.

Below are various types of data files with their extensions and functions.

File type	Extension	Function
Executable	exe, com, bin	ready-to-run code
Text	txt, doc	textual data, documents
Source	c, f77, asm	source in various languages
Object	obj, o	object code
Library	lib, a	library routines
Archive	tar, zip, arc	grouped files
Compressed	Z, gz	compressed
Print/view	ps, eps, pdf	printing or viewing
Word processor	ppt, wp, tex	various word processors

File characteristics

Data file(s) should have at least one of the following behaviors or qualities:

i. **Hit rate**: This is the frequency with which active records in the file are accessed. File records that are accessed more often are said to have high hit rate, while those records which are rarely accessed are said to have low hit rate.

ii. **Volatility**: The ease at which records are either added or deleted from file is described as file volatility. This calls for careful handling of data file to avoid sudden loss of its contents (records) due to its volatility nature.

iii. **Growth**: This is referred to the increase in size of the file. When a file is created, it grows as new records are added to the file. The file grows as its contents (records) increase.

iv. **Size**: The file size is expressed in terms of characters in the field of the record and the number of records. When a file is created, the size is determined by the amount of data stored in the file.

WEEK 6

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ Different types of files: Master file, Transaction file, Reference file, etc.
- ✓ The concept of master file, transaction file and activity file.

File generations

This is file security technique applied to the master files held on magnetic tapes. This technique involves retaining files relating to two previous periods/generations in addition to the current updated file and the transaction file. The two periods files and the current updated file constitute three generations referred to as *Grandfather* – *Father* – *Son*.

Description of Generation Technique

Old Master File: The old master file used during the first updating period is termed *Father* while the new master file produced is termed the *son*.

New Master File (that's the son): The new master file (son) used during the second updating period becomes the *Father* while the master file produced during the second updating forms the *Son*.

Grand-Father-Father & Son Analog: Now considering the two updating periods, the old master file used in the first period now becomes the *Grandfather*, the new master file used in the second updating becomes the *Father* while the new master file produced in the second period forms the *Son*.

Therefore, we have, *Grandfather – Father – Son* forming a generation. On the next updating run, the *Grandfather* file is over written and can then be used again, as the *great-grand-father* is not allowed.

Transaction file: Transaction file is related to the activities within an organization. This file contains operational data extracted from the transaction of the organization. For example, if a new staff is employed or an one is promoted, the data is transaction data and when such records are collected, a transaction file is obtained. A transaction file is often used to update the master file. Once it has played this role, it is no longer needed. It may however be retained or archived for security control purposes. Examples of transaction file include; invoices, hour worked, new qualifications, promotion, electricity meter reading, etc.

Activity file: The work file is derived from the either the master or transaction files. They are often temporal in nature as they may not be needed/required if

serve the purpose. For example, a personnel file can be interrogated to display the records of employees who meet certain criteria, and when the records are extracted from the master file, they are kept in a temporary work file before they are printed. In most cases, work files are created to hold the sorted records.

Reference file: A reference file is a file which contains records that may be used in the future or for reference purposes. Reference file is also called standing file whose records are responsibly permanent.

Archive file: This file is also referred to as historical file as it contains old files/records which are currently not useful or no longer useful. For instance, the files containing particulars of former clients, records of graduated students in an institution, etc.

WEEK 7

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ Types of storage devices and media
- ✓ The characteristics of magnetic storage media, tape, disk, cartridge, bubble, hard disk, CD-ROM, floppy disks, zip disk, tape streamer, flash memory, optical disk.

The nature of storage devices

Purpose of storage

Various forms of storage, based on various natural phenomena, have been invented. So far, no practical universal storage medium exists, and all forms of storage have some drawbacks. Therefore a computer system usually contains several kinds of storage, each with an individual purpose.

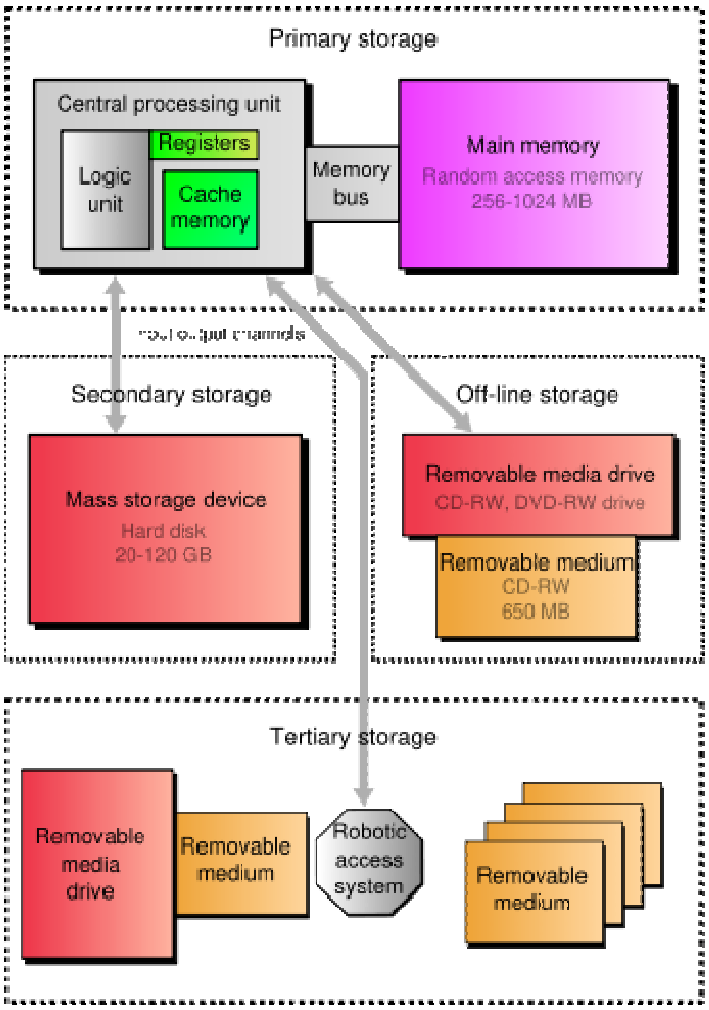
A digital computer represents each datum using the binary numeral system. Text, numbers, pictures, audio, and nearly any other form of information can be converted into a string of bits, or binary digits, each of which has a value of 1 or 0. The most common unit of storage is the byte, equal to 8 bits. A piece of information can be handled by any computer whose storage space is large enough to accommodate *the binary representation of the piece of information*, or simply data. For example, using eight million bits, or about one megabyte, a typical computer could store a small novel.

Traditionally the most important part of every computer is the central processing unit (CPU, or simply a processor), because it actually operates on data, performs any calculations, and controls all the other components.

Without a significant amount of memory, a computer would merely be able to perform fixed operations and immediately output the result. It would have to be reconfigured to change its behavior. This is acceptable for devices such as desk calculators or simple digital signal processors. Von Neumann machines differ in that they have a memory in which they store their operating instructions and data. Such computers are more versatile in that they do not need to have their hardware reconfigured for each new program, but can simply be reprogrammed with new in-memory instructions; they also tend to be simpler to design, in that a

relatively simple processor may keep state between successive computations to build up complex procedural results. Most modern computers are von Neumann machines.

In practice, almost all computers use a variety of memory types, organized in a storage hierarchy around the CPU, as a tradeoff between performance and cost. Generally, the lower a storage is in the hierarchy, the lesser its bandwidth and the greater its access latency is from the CPU. This traditional division of storage to primary, secondary, tertiary and off-line storage is also guided by cost per bit.



Various forms of storage, divided according to their distance from the central processing unit. The fundamental components of a general-purpose computer are arithmetic and logic unit, control circuitry, storage space, and input/output devices. Technology and capacity as in common home computers around 2005.

Primary storage

Primary storage, presently known as **memory**, is the only one directly accessible to the CPU. The CPU continuously reads instructions stored there

and executes them. Any data actively operated on is also stored there in uniform manner.

Historically, early computers used delay lines, Williams tubes, or rotating magnetic drums as primary storage. By 1954, those unreliable methods were mostly replaced by magnetic core memory, which was still rather cumbersome. Undoubtedly, a revolution was started with the invention of a transistor, that soon enabled then-unbelievable miniaturization of electronic memory via solid-state silicon chip technology.

This led to a modern random access memory (RAM). It is small-sized, light, but quite expensive at the same time. (The particular types of RAM used for primary storage are also volatile, i.e. they lose the information when not powered).

As shown in the diagram, traditionally there are two more sub-layers of the primary storage, besides main large-capacity RAM:

- Processor registers are located inside the processor. Each register typically holds a word of data (often 32 or 64 bits). CPU instructions instruct the arithmetic and logic unit to perform various calculations or other operations on this data (or with the help of it). Registers are technically among the fastest of all forms of computer data storage, being switching transistors integrated on the CPU's chip, and functioning as electronic "flip-flops".
- Processor cache is an intermediate stage between ultra-fast registers and much slower main memory. It's introduced solely to increase performance of the computer. Most actively used information in the main memory is just duplicated in the cache memory, which is faster, but of much lesser capacity. On the other hand it is much slower, but much larger than processor registers. Multi-level hierarchical cache setup is also commonly

used—*primary cache* being smallest, fastest and located inside the processor; *secondary cache* being somewhat larger and slower.

Main memory is directly or indirectly connected to the CPU via a *memory bus*, today sometimes referred to as a front side bus. It is actually comprised of two buses (not on the diagram): an address bus and a data bus. The CPU firstly sends a number through an address bus, a number called memory address, that indicates the desired location of data. Then it reads or writes the data itself using the data bus. Additionally, a memory management unit (MMU) is a small device between CPU and RAM recalculating the actual memory address, for example to provide an abstraction of virtual memory or other tasks.

As the RAM types used for primary storage are volatile (cleared at start up), a computer containing only such storage would not have a source to read instructions from, in order to start the computer. Hence, non-volatile primary storage containing a small startup program (BIOS) is used to bootstrap the computer, that is, to read a larger program from non-volatile *secondary* storage to RAM and start to execute it. A non-volatile technology used for this purpose is called ROM, for read-only memory (the terminology may be somewhat confusing as most ROM types are also capable of *random access*).

Many types of "ROM" are not literally *read only*, as updates are possible; however it is slow and memory must be erased in large portions before it can be re-written. Some embedded systems run programs directly from ROM (or similar), because such programs are rarely changed. Standard computers do not store non-rudimentary programs in ROM, rather use large capacities of secondary storage, which is non-volatile as well, and not as costly.

Recently, *primary storage* and *secondary storage* in some uses refer to what was historically called, respectively, *secondary storage* and *tertiary storage*.

Secondary storage

A hard disk drive with protective cover removed.

Secondary storage, or *storage* in popular usage, differs from primary storage in that it is not directly accessible by the CPU. The computer usually uses its input/output channels to access secondary storage and transfers desired data using intermediate area in primary storage. Secondary storage does not lose the data when the device is powered down—it is non-volatile. Per unit, it is typically also an order of magnitude less expensive than primary storage. Consequently, modern computer systems typically have an order of magnitude more secondary storage than primary storage and data is kept for a longer time there.

In modern computers, hard disks are usually used as secondary storage. The time taken to access a given byte of information stored on a hard disk is typically a few thousandths of a second, or milliseconds. By contrast, the time taken to access a given byte of information stored in random access memory is measured in billionths of a second, or nanoseconds. This illustrates the very significant access-time difference which distinguishes solid-state memory from rotating magnetic storage devices: hard disks are typically about a million times slower than memory. Rotating optical storage devices, such as CD and DVD drives, have even longer access times.

Some other examples of secondary storage technologies are: flash memory (e.g. USB sticks or keys), floppy disks, magnetic tape, paper tape, punch cards, standalone RAM disks, and Zip drives.

The secondary storage is often formatted according to a filesystem format, which provides the abstraction necessary to organize data into files and directories, providing also additional information (called metadata) describing the owner of a certain file, the access time, the access permissions, and other information.

Most computer operating systems use the concept of virtual memory, allowing utilization of more primary storage capacity than is physically available in the system. As the primary memory fills up, the system moves the least-used chunks (*pages*) to secondary storage devices (to a swap file or page file), retrieving them later when they are needed. As more of these retrievals from slower secondary storage are necessary, the more the overall system performance is degraded.

Tertiary storage

Largetape library. Tape cartridges placed on shelves in the front, robotic arm moving in the back. Visible height of the library is about 180 cm.

Tertiary storage or **tertiary memory**, provides a third level of storage. Typically it involves a robotic mechanism which will *mount* (insert) and *dismount* removable mass storage media into a storage device according to the system's demands; this data is often copied to secondary storage before use. It is primarily used for archival of rarely accessed information since it is much slower than secondary storage (e.g. 5-60 seconds vs. 1-10 milliseconds). This is primarily useful for extraordinarily large data stores, accessed without human operators. Typical examples include tape libraries and optical jukeboxes.

When a computer needs to read information from the tertiary storage, it will first consult a catalog database to determine which tape or disc contains the information. Next, the computer will instruct a robotic arm to fetch the medium and place it in a drive. When the computer has finished reading the information, the robotic arm will return the medium to its place in the library.

Off-line storage

Off-line storage, also known as **disconnected storage**, is a computer data storage on a medium or a device that is not under the control of a processing unit. The medium is recorded, usually in a secondary or tertiary storage device, and then physically removed or disconnected. It must be inserted or connected by a human operator before a computer can access it again. Unlike tertiary storage, it cannot be accessed without human interaction.

Off-line storage is used to transfer information, since the detached medium can be easily physically transported. Additionally in case a disaster, for example a fire, destroys the original data, a medium in a remote location will be probably unaffected, enabling disaster recovery. Off-line storage increases a general information security, since it is physically inaccessible from a computer, and data confidentiality or integrity cannot be affected by computer-based attack

techniques. Also, if the information stored for archival purposes is accessed seldom or never, off-line storage is less expensive than tertiary storage.

In modern personal computers, most secondary and tertiary storage media are also used for off-line storage. Optical discs and flash memory devices are most popular, and to much lesser extent removable hard disk drives. In enterprise uses, magnetic tape is predominant. Older examples are floppy disks, Zip disks, or punched cards.

Characteristics of storage

A 1GB DDR RAM memory module

Storage technologies at all levels of the storage hierarchy can be differentiated by evaluating certain core characteristics as well as measuring characteristics specific to a particular implementation. These core characteristics are volatility, mutability, accessibility, and addressibility. For any particular implementation of any storage technology, the characteristics worth measuring are capacity and performance.

Volatility

Non-volatile memory

Will retain the stored information even if it is not constantly supplied with electric power. It is suitable for long-term storage of information. Nowadays used for most of secondary, tertiary, and off-line storage. In 1950s and 1960s, it was also used for primary storage, in the form of magnetic core memory.

Volatile memory

Requires constant power to maintain the stored information. The fastest memory technologies of today are volatile ones (not a universal rule). Since primary storage is required to be very fast, it predominantly uses volatile memory.

Differentiation

Dynamic memory

A form of volatile memory which also requires the stored information to be periodically re-read and re-written, or refreshed, otherwise it would vanish.

Static memory

A form of volatile memory similar to DRAM with the exception that it does not refresh on occasion.

Mutability

Read/write storage or mutable storage

Allows information to be overwritten at any time. A computer without some amount of read/write storage for primary storage purposes would be useless for many tasks. Modern computers typically use read/write storage also for secondary storage.

Read only storage

Retains the information stored at the time of manufacture, and **write once storage** (WORM) allows the information to be written only once at some point after manufacture. These are called **immutable storage**. Immutable storage is used for tertiary and off-line storage. Examples include CD-ROM and CD-R.

Slow write, fast read storage

Read/write storage which allows information to be overwritten multiple times, but with the write operation being much slower than the read operation. Examples include CD-RW.

Accessibility

Random access

Any location in storage can be accessed at any moment in approximately the same amount of time. Such characteristic is well suited for primary and secondary storage.

Sequential access

The accessing of pieces of information will be in a serial order, one after the other; therefore the time to access a particular piece of information depends upon which piece of information was last accessed. Such characteristic is typical of off-line storage.

Addressability

Location-addressable

Each individually accessible unit of information in storage is selected with its numerical memory address. In modern computers, location-addressable storage usually limits to primary storage, accessed internally by computer

programs, since location-addressability is very efficient, but burdensome for humans.

File addressable

Information is divided into *files* of variable length, and a particular file is selected with human-readable directory and file names. The underlying device is still location-addressable, but the operating system of a computer provides the file system abstraction to make the operation more understandable. In modern computers, secondary, tertiary and off-line storage use file systems.

Content-addressable

Each individually accessible unit of information is selected with a hash value, or a short identifier with a number pertaining to the memory address the information is stored on. Content-addressable storage can be implemented using software (computer program) or hardware (computer device), with hardware being faster but more expensive option.

Capacity

Raw capacity

The total amount of stored information that a storage device or medium can hold. It is expressed as a quantity of bits or bytes (e.g. 10.4 megabytes).

Density

The compactness of stored information. It is the storage capacity of a medium divided with a unit of length, area or volume (e.g. 1.2 megabytes per square inch).

Performance

Latency

The time it takes to access a particular location in storage. The relevant unit of measurement is typically nanosecond for primary storage, millisecond for secondary storage, and second for tertiary storage. It may make sense to separate read latency and write latency, and in case of sequential access storage, minimum, maximum and average latency.

Throughput

The rate at which information can be read from or written to the storage. In computer data storage, throughput is usually expressed in terms of megabytes per second or MB/s, though bit rate may also be used. As with latency, read rate and write rate may need to be differentiated. Also accessing media sequentially, as opposed to randomly, typically yields maximum throughput.

Fundamental storage technologies

As of 2008, the most commonly used data storage technologies are semiconductor, magnetic, and optical, while paper still sees some limited usage. Some other fundamental storage technologies have also been used in the past or are proposed for development.

Semiconductor

Semiconductor memory uses semiconductor-based integrated circuits to store information. A semiconductor memory chip may contain millions of tiny transistors or capacitors. Both *volatile* and *non-volatile* forms of semiconductor memory exist. In modern computers, primary storage almost exclusively consists of dynamic volatile semiconductor memory or dynamic random access memory.

Since the turn of the century, a type of non-volatile semiconductor memory known as flash memory has steadily gained share as off-line storage for home computers. Non-volatile semiconductor memory is also used for secondary storage in various advanced electronic devices and specialized computers.

Magnetic

Magnetic storage uses different patterns of magnetization on a magnetically coated surface to store information. Magnetic storage is *non-volatile*. The information is accessed using one or more read/write heads which may contain one or more recording transducers. A read/write head only covers a part of the surface so that the head or medium or both must be moved relative to another in order to access data. In modern computers, magnetic storage will take these forms:

- Magnetic disk
  - Floppy disk, used for off-line storage
  - Hard disk, used for secondary storage
- Magnetic tape data storage, used for tertiary and off-line storage

In early computers, magnetic storage was also used for primary storage in a form of magnetic drum, or core memory, core rope memory, thin film memory, twistor memory or bubble memory. Also unlike today, magnetic tape was often used for secondary storage.

Optical

Optical storage, the typical Optical disc, stores information in deformities on the surface of a circular disc and reads this information by illuminating the surface with a laser diode and observing the reflection. Optical disc storage is *non-volatile*. The deformities may be permanent (read only media), formed once

(write once media) or reversible (recordable or read/write media). The following forms are currently in common use:[4]

- CD, CD-ROM, DVD, BD-ROM: Read only storage, used for mass distribution of digital information (music, video, computer programs)
- CD-R, DVD-R, DVD+R BD-R: Write once storage, used for tertiary and off-line storage
- CD-RW, DVD-RW, DVD+RW, DVD-RAM, BD-RE: Slow write, fast read storage, used for tertiary and off-line storage

- Ultra Density Optical or UDO is similar in capacity to BD-R or BD-RE and is slow write, fast read storage used for tertiary and off-line storage.

Magneto-optical disc storage is optical disc storage where the magnetic state on a ferromagnetic surface stores information. The information is read optically and written by combining magnetic and optical methods. Magneto-optical disc storage is *non-volatile*, *sequential access*, slow write, fast read storage used for tertiary and off-line storage.

3D optical data storage has also been proposed.

Paper

Paper data storage, typically in the form of paper tape or punch cards, has long been used to store information for automatic processing, particularly before general-purpose computers existed. Information was recorded by punching holes into the paper or cardboard medium and was read mechanically (or later optically) to determine whether a particular location on the medium was solid or contained a hole.

Uncommon

Vacuum tube memory

A Williams tube used a cathode ray tube, and a Selectron tube used a large vacuum tube to store information. These primary storage devices were short-lived in the market, since Williams tube was unreliable and Selectron tube was expensive.

Electro-acoustic memory

Delay line memory used sound waves in a substance such as mercury to store information. Delay line memory was dynamic volatile, cycle sequential read/write storage, and was used for primary storage.

Phase-change memory

uses different mechanical phases of phase change material to store information in an X-Y addressable matrix, and reads the information by observing the varying electric resistance of the material. Phase-change memory would be non-volatile, random access read/write storage, and might be used for primary, secondary and off-line storage. Most rewritable and many write once optical disks already use phase change material to store information.

Holographic storage

stores information optically inside crystals or photopolymers. Holographic storage can utilize the whole volume of the storage medium, unlike optical disc storage which is limited to a small number of surface layers. Holographic storage would be non-volatile, sequential access, and either write once or read/write storage. It might be used for secondary and off-line storage. See Holographic Versatile Disc (HVD).

Molecular memory

stores information in polymers that can store electric charge. Molecular memory might be especially suited for primary storage.

Related technologies

Network connectivity

A secondary or tertiary storage may connect to a computer utilizing computer networks. This concept does not pertain to the primary storage, which is shared between multiple processors in a much lesser degree.

- **Direct-attached storage** (DAS) is a traditional mass storage, that does not use any network. This is still a most popular approach. This term was coined lately, together with NAS and SAN.
- **Network-attached storage** (NAS) is mass storage attached to a computer which another computer can access at file level over a local-area network, a private wide-area network, or in the case of online file storage, over the Internet. NAS is commonly associated with the NFS and CIFS/SMB protocols.
- **Storage area network** (SAN) is a specialized network, that provides other computers with storage capacity. The crucial difference between NAS and SAN is the former presents and manages file systems to client computers, whilst a latter provides access at block-addressing (raw) level, leaving it to attaching systems to manage data or file systems within the provided capacity. SAN is commonly associated with Fibre Channel networks.

Robotic storage

Large quantities of individual magnetic tapes, and optical or magneto-optical discs may be stored in robotic tertiary storage devices. In tape storage field they are known as tape libraries, and in optical storage field optical jukeboxes, or optical disk libraries per analogy. Smallest forms of either technology containing just one drive device are referred to as autoloaders or autochangers.

Robotic-access storage devices may have a number of slots, each holding individual media, and usually one or more picking robots that traverse the slots and load media to built-in drives. The arrangement of the slots and picking devices affects performance. Important characteristics of such storage are possible expansion options: adding slots, modules, drives, robots. Tape libraries may have from 10 to more than 100,000 slots, and provide terabytes or petabytes of near-line information. Optical jukeboxes are somewhat smaller solutions, up to 1,000 slots.

Robotic storage is used for backups, and for high-capacity archives in imaging, medical, and video industries. Hierarchical storage management is a most known archiving strategy of automatically *migrating* long-unused files from fast hard disk storage to libraries or jukeboxes. If the files are needed, they are *retrieved* back to disk.

WEEK 8

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ Types of storage devices and media
- ✓ The characteristics of magnetic storage media, tape, disk, cartridge, bubble, hard disk, CD-ROM, floppy disks, zip disk, tape streamer, flash memory, optical disk.

Storage media/devices

A device that can receive data and retain for subsequent retrieval is called storage medium. The file storage and retrieval describe the organization, storage, location and retrieval of coded information in computer system. Important factors in storing and retrieving information are the type of media or storage device used to store information, the media's storage capacity, the speed of access and information transfer to and from the storage media, the number of times new information can be written to the media and how the media interacts with the computer.

Types of file storage

File storage can be classified as permanent storage and temporal storage. File can also be classified and having been stored to or retrieved from primary or secondary memory. The primary memory is also known as main memory is the computer's main random access memory (RAM). All information that is processed by the computer must first pass through main memory. Secondary is any form of memory/storage other than the main computer memory. Such memories (devices) cover a wide range of capacities and speed of access. The examples are; magnetic disk storage which include floppies, hard-disks, cartridge, exchangeable multi-platter, CD-ROMs, fixed disks, flash disk and magnetic tape.

a. Permanent storage

Information is stored permanently on storage that is written to only once, such ROM (read only memory) chips and CD-ROM (compact disk read only memory). Permanent storage media is used for archiving files or in case of ROM chips, for storing basic information that the computer needs to function that cannot be overwritten.

b. Temporary storage

Temporary information storage is used as intermediate storage between permanent or semi-permanent storage and computer's central processing unit (CPU). The temporary storage is the form of memory chip called random access memory (RAM). Information is stored in RAM while the CPU is using it. It is then returned to a more permanent form of memory. RAM chips are known as volatile memory because they must have power supplied to them continuously otherwise they lose the content of their memory.

How information storage and retrieval work

All information possessed can be expressed as numbers in binary notation. These binary numbers are strings of bits, 1s and os, that are grouped together in sequences of eight called byte. The two value bits are physically stored in storage media in variety of ways. For instance, they can be represented by the presence and absence of electrical charge on a pair of closely spaced conductors called a capacitor.

In computer system, the CPU processes information and maintains it temporarily in RAM in the form of strings of bits called files. When this temporary information needs to be stored permanently, the CPU finds unused space in the storage media, generally a hard disk, floppy disk or magnetic tape. It then instructs a device capable of making change to the media (called a read/write head) to start transferring bits. The read/write head and its associated electronics convert each bit to the equivalent physical value on the media. The recording/writing mechanism then continues to the next location on the media capable of storing bits.

When the CPU needs to access some piece of stored information, the process is reversed. The CPU determines where on the physical media the appropriate file

is stored, then directs the read/write head to position itself at that location on the media, and then directs it to read the information stored there.

Storage media:

Information is stored on many different types of media, the most common being floppy disks, hard drives, CD-ROMs, magnetic tapes and flash disks.

Floppy disks

Floppy disks are most often used to store information, such as application programs that are frequently accessed by the users. A floppy disk is a thin piece of magnetic material inside a protective envelop. The size of the disk is usually given as a diameter of the magnetic media. The two most common sizes are; 2.5 inch and 5.25 inch. Both sizes of floppies are removable disk – that is, they must be inserted into a compatible disk drive to read from or written to. This drive is usually internal to, or part of a computer. Most floppy drives today are double sided, with one head on each side of the disk. This doubles the storage capacity of the disk, allowing it to be written to on both sides. Information is organized on the disk by dividing the disk into tracks and sectors. Tracks are concentric circular regions on the surface of the disk. Before a floppy can be used, the computer has to format it by placing special information on the disk that enables the computer to find each track and sector.

Hard drive

Hard drives consist of grid circular platters of magnetizable material sealed in a metal box with associated read/writ heads. They are usually internal to a computer. Most

hard drives have multiple platters stacked on top of one another, each with it's own read/write head. The media in a hard drive is generally not removable from the drive assembly, although external hard drive do exist with removable hard disks. The read/write heads in a hard drive are precisely aligned with the surfaces of the hard disks, allowing thousands of tracks and dozens of sectors per track. The combination of more heads and more tracks allows hard drives to store more data and to transfer data at a higher rate than floppy disks.

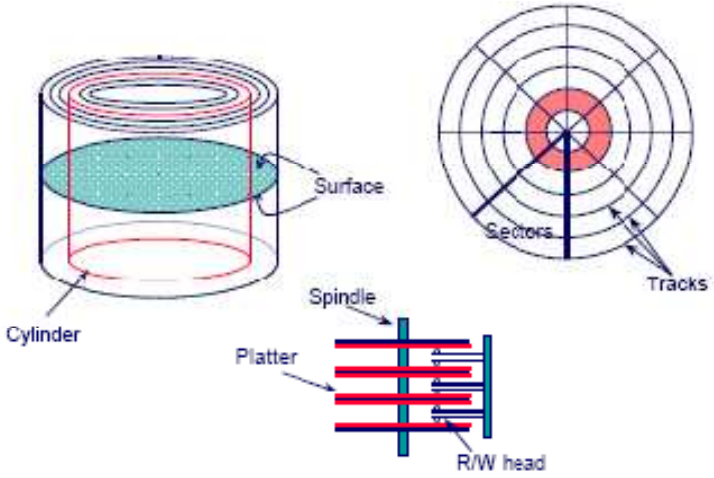
Accessing information on a hard disk involves moving the heads to the right track and then waits for the correct sector to revolve underneath the head. Seek time is the average time needed/required to move the heads from one track to some other desired track on the disk. The time needed to move from one track to a neighbouring track is often in the 1 millisecond (i.e one thousand per second) range, and the average seek time to reach arbitrary track anywhere on the disk is in the 6 to 15 millisecond range.

Rational latency is the average time required for the correct sector to come under the head once they are positioned on the correct track. This time depends on how fast the disk is revolving. Today, many drives run 120 to 180 (or more) revolutions per second or faster, yielding average rotational latencies of a few milliseconds.

If a file required more than one sector for storage, the positions of the sectors on the individual tracks can greatly affect the average access time. Typically, it takes the disk controller a small amount of time to finish reading a sector. If the next sector to be read is the neighbouring sector on the track, the electronic may not have enough time to get ready to read it before it rotates under the read/write head. If this is the case, the drive must wait until the sector comes all the way round again. This access time can be reduced by interleaving, or alternatively placing the sectors on the tracks so that sequential sectors for the same file are

separated from each other by one or two sectors. When information is distributed optimally, the device controller is ready to start reading just as the appropriate sector comes under the read/write head.

After many files have been written to and erased from a disk, fragmentation can occur. Fragmentation happens when pieces of single files are inefficiently distributed in many locations on a disk. The result is an increase in average file access time. This problem can be fixed by running a defragmentation program, which goes through the drive track by track and rearranges the sectors for each file so that they can be accessed more quickly.



Disk structure

Unlike floppy drives, in which the read/write heads actually touch the surface of the material, the heads in most hard disks float slightly of the surface. When the heads accidentally touch the media, either because the drive is dropped or bumped hard or because of an electrical malfunction, the surface becomes scratched. Any data stored where the head has touched the disk is lost. This is called head crash. To help reduce the possibility of head crash, most disk controllers park the heads over an unused track on the disk when the drive is not being used by the CPU.

It is important to have a large hard disk drive to undertake some tasks like video and sound editing, however the latest desktop computers come with a Minimum of 60Gbyte of capacity which is enough for most standard tasks. Currently, hard disk drives have capacities up to 120Gbytes and data transfer rates of 160Mbits per second.

CD-ROMs

While magnetic material is thee dominant media for read/write information storage (that's files that are read from and written to frequently), other media have become popular for more permanent storage applications. One of the most common alternatives information storage mediums is the CD-ROM. CD-ROMs are plastic disks on which individual bits are stored as pits burned onto the surface of the disk by high-powered lasers. The surface of the disk is then covered with a layer of reflecting material such as aluminum.

The computer uses a CD-ROM drive to access information on the CD-ROM. The drive be external to, or part of, the computer. A light-sensitive instrument in the drive reads the disk by watching the amount of light reflected back from a smaller laser positioned over the spinning disk. Such disks can hold large amounts of information, but can only be written to once. The drives capable of writing to CD-ROMs are called *write once, read many (WORM) drives.* Due to their inexpensive production costs, CD-ROMs are widely used today for storing music, video, and application programs.

Magnetic tape

Magnetic tape has served as efficient and reliable information storage media since 1950s. Most magnetic tapes are made of mylar, a type of strong plastic, in which metallic particles have been embedded. A read/write head is identical to those used for audio tape reads and writes binary information to the tape. Reel – to-reel magnetic tape is commonly used to store information for large mainframe or supercomputers. High-density cassette tapes, resembling audio cassette tapes are used to store information for personal computers and mainframes.

Magnetic tape storage has advantage of being able to hold enormous amounts of data. For this reason, it is used to store information on the largest computer system. Magnetic tape has two major shortcomings; it has a very slow data access time when compared to other form of storage media and access to information on magnetic tape is sequential. In sequential data storage, data are stored with the first bit at the beginning of the tape and the last bit at the end of the tape, in a linear fashion. To access a random bit of information, the tape drive has to forward or reverse through the tape until it finds the location of the bit. The bit closest to the location of the read/write head can be accessed relatively faster, but bits far away take a considerable time to access. RAM on the other hand, is random access, meaning that it can locate any one bit as easily as any other.

Flash memory

Another type of storage device, called Flash memory, traps small amounts of electric charge in ‘wells’ on the surface of a chip. Side effects of this trapped charge, such as

the electric field it creates are late used to read the stored value. To rewrite to flash memory, the charges in the wells must be first drained. Such drives are useful for storing information that changes infrequently. Most flash memories are of large capacity and hence are used to store large volume of data.

Future technology

Although magnetic and CD-ROM technologies continue to increase in storage density, a varieties of new technologies are emerging. Redundant Arrays of Independent Disks (RAIDs) are storage systems that look like one device but are actually composed of multiple hard disks. These systems provide more storage and also read data simultaneously from many drives. The result is a faster rate of data transfer to the CPU, which is important for many very high speed computer applications, especially those involving large databases of information.

Disk management issues

• ***Formatting***

– *Physical*: divide the blank slate into sectors identified by headers containing such information as sector number; sector interleaving.

– *Logical*: marking bad blocks; partitioning (optional) and writing a blank directory on disk; installing file allocation tables, and other relevant information (file system initialization)

• ***Reliability***

– disk interleaving or striping

– RAIDs (Redundant Array of Inexpensive Disks): various levels, e.g., level 0 is disk mirroring or shadowing, consists of keeping a duplicate of each disk)

• ***Controller caches***

– newer disks have on-disk caches (128KB—512KB)

WEEK 9

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ Different file access types:- random access and direct access storage methods.
- ✓ Seek time and rotational delay
- ✓ The concept of a buffer and its functions
- ✓ The calculation of buffer requirement of a file.

Random access method: Random access is a type of memory access in which storage location can be accessed in any order. This means that a file is said to be randomly accessed if the sequence of transactions submit to it does not match any sequence in which records may be organized.

Direct access: The direct access of the index sequential makes full use of the index to access the records using the record address associated with the index. The records are accessed without any defined sequence of sorting. The method is used when the transaction is to be processed immediately without waiting for sorting and its widely used in commercial data processing because it can be allow both sequential and random methods of access.

Seek Time

Seek time is one of the three delays associated with reading or writing data on a computer's drive and somewhat similar for CD or DVD drives. The others are rotational delay and transfer time and their sum is the access time. In order to read or write data in a particular place on the disk, the read/write head of the disk needs to be physically moved to the correct place. This process is known as seeking and the time it takes for the head to move to the right place is the seek time. Seek time for a given disk varies depending on how far the head destination is from its origin at the time of each read/write instruction.

Rotational Delay

The rotational delay is the time required for the address area of the disk to rotate into a position where it is accessible by the read/write head.

Transfer Time

Transfer Time is the number of bits that are transferred per unit of time. The unit therefore is in bit per second (bps).

Concept of buffer

Buffer is a temporary memory for data, normally used to accommodate the difference in the rate at which two devices can handle data during transfer. The buffer may be built into a peripherals device such as printer or disk drive or may be part of the system's main memory.

Buffering technique is used to compensate for the slow and possible erratic rate at which a peripheral device produces or consume data. If the device communicates directly with the program, the program is constraint to run in synchronism with the device to operate independently.

Consider a program sending output to a slow device, a memory area (the buffer) is set aside for communication. The program places data in the buffer at its own rate. Although the device may be slow, the program does not have to stop unless the buffer fills up, at the same time the device runs at full speed unless the buffer is empty. This buffering technique is applicable to input device.

By providing a data buffer in the device or control unit, devices that would ordinarily require the channel for a long period can be made to perform independently of the channel. For instance, a card reader may physically require about 60 ms to read a card and transfer the data to memory through the channel. However, a buffered card reader always reads one card before it is needed and save the 80 bytes in a buffer in the card reader or control unit. When the channel requests that a card be read, the contents of the 80-bytes buffered are transferred to memory at high speed (e.g 100 µs) and the channel is released. The device then proceeds to read the next card and buffer it independently. Card readers, card punches and printers are frequently buffered.

Functions of Buffer

Buffer synchronizes the speed of data transfer among the systems/devices that are under the control of the CPU. It makes individual devices (that is, input and output devices) to perform their functions independently. This is because, the rate at which the CPU performs/sends or receive it data is very high compare to the rate at which the I/O devices receive or send data. Buffer is equally used to accommodate the differences in the rate at which two devices can handle data during data communication/transfer.

WEEK 10

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ What is file organizational structure
- ✓ File processing
- ✓ Acoustical data structure

File Processing

File is a body of stored data or information in an electronic format. Almost all information stored on computers is in the form of files. Files reside on mass storage devices such as hard drives, CD-ROMs, magnetic tape, and floppy disks. When the central processing unit (CPU) of a computer needs data from a file, or needs to write data to a file, it temporarily stores the file in its main memory, or Random Access Memory (RAM), while it works on the data.

Information in computers is usually classified into two different types of files: data files (those containing data) and program files (those that manipulate data). Within each of these categories, many different types of files exist that store various kinds of information.

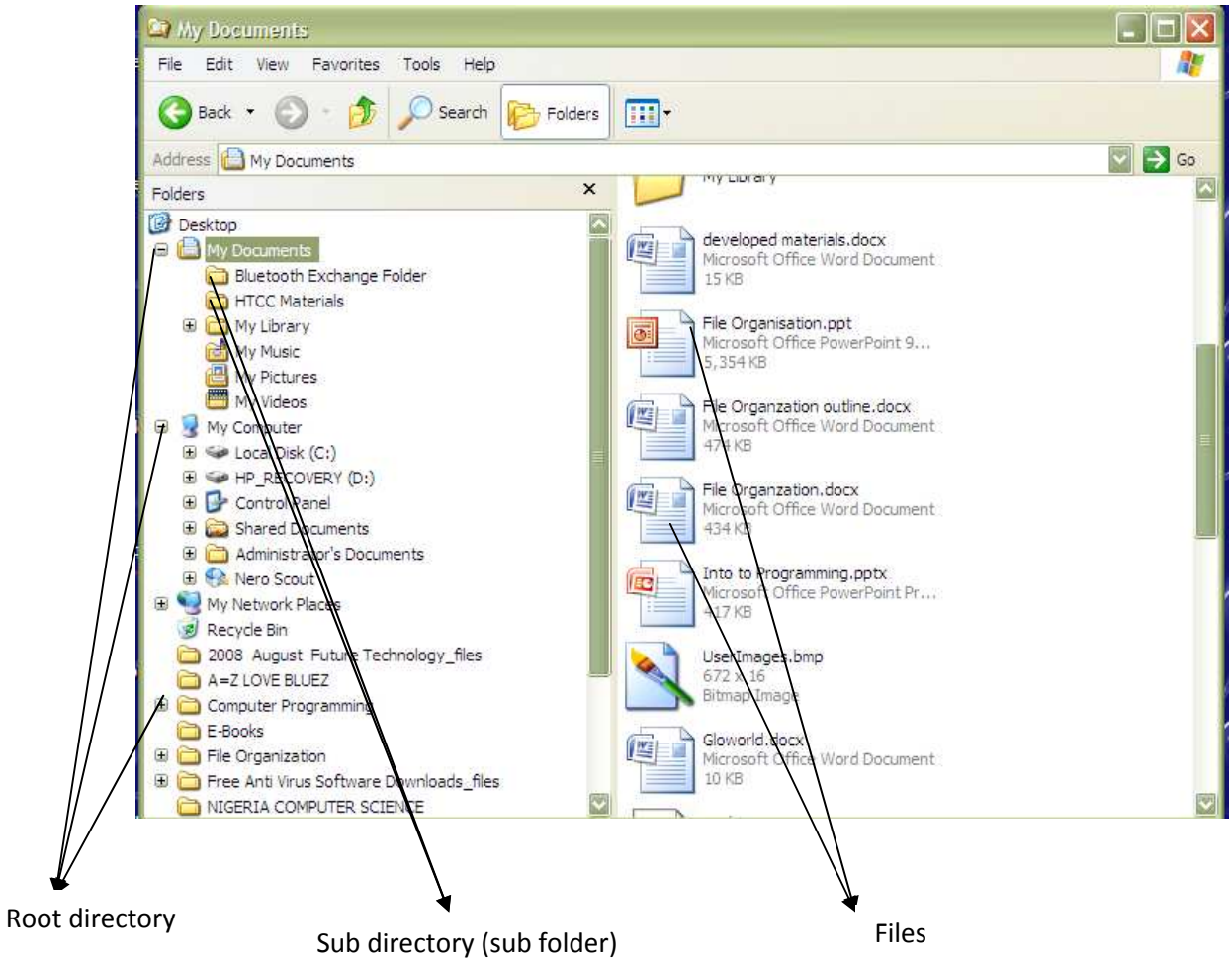
Different computer operating systems have unique rules for the naming of files. Windows 95 (Win95) and disk operating systems (DOS), for instance, make use of an extension attached to the end of each filename in order to indicate the type of file. Extensions begin with a period (.), and then have one or more letters. An example of a file extension used in Win95 and DOS is .bak, which indicates that the file is a backup file.

When saving a file, a user can give it any name within the rules of the operating system. In addition, the name must be unique. Two files in the same directory may not have the same name, but some operating systems allow the same name for one file to be placed in more than one location. These additional names are called aliases.

Directory files contain information used to organize other files into a hierarchical structure. In the Macintosh operating system, directory files are called folders. The topmost directory in any file system is the root directory. A directory contained within another directory is called a subdirectory. Directories containing one or more subdirectories are called parent directories. Directory files contain programs or commands that are executable by the computer.

Executable files have a .exe suffix at the end of their names and are often called EXE (pronounced EX-ee) files. Text files contain characters represented by their ASCII (American Standard Code for Information Interchange) codes. These files are often called ASCII (pronounced ASK-ee) files. Files that contain words, sentences, and bodies of paragraphs are

frequently referred to as text files. The diagram below shows the root directory, sub director and file



File Processing Activities

The file processing operations deal with the various activities which are performed on the file. These operations are briefly described as shown below;

- File creation: The process of bringing file into existence is called file creation.
- Searching: Searching is locating data in a file by reference to a special field of each record/data called the **key**. The key is a unique filed used to identify certain

record in a file. If a record is to be inserted into a file, it must be given a unique key value.

- Retrieving/reading: This involves reading an existing data from a form of storage or input medium.
- Writing: Writing is the act of recording data onto some form of storage.
- Deleting: This means removing a record or item of data from a storage medium such as disk/tape.
- File updating: This is an act of changing values in one or more records of a file without changing the organization of the file. That is making the file modern by adding most recent data to the file.
- Sorting: Sorting means rearranging data in either ascending or descending order. It involves the arrangement of grouped data elements into a predetermined sequence to facilitate file processing.
- Calculating: The arithmetic or logical manipulation of data in a file is referred to as calculation.
- File querying/interrogating: This is retrieving specific data from a file according to the set of retrieval criteria.
- File merging: Combining multiple sets of data files or records to produce only one set, usually in an ordered sequence is referred to as file merging.
- Reporting: Reporting is a file processing operation that deals with the production (printing) of report from the file in a specified format.
- File display: The contents of a data file can be displayed either on the computer screen as soft copy or printed on the paper as hard copy.
- File storage: When a file is created, it is stored in the appropriate storage medium such as disk, flash disk, tape, drum, etc.

WEEK 11

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ What is a table
- ✓ What is an array
- ✓ What is a list

Table

Table is a collection of records. Each record stores information associated with a key by which specific records are found or the records may be arranged in an array so that the index is the key. In commercial applications the word table is often used as a synonym for matrix or array.

Array

Array is an ordered collection of a number of elements of the same type, the number being fixed unless the array is flexible. The elements of one array may be of type integer, those of another array may be of type real, while the elements of a third array ma be of type character string. Each element has a unique list of index values that determine its position in the ordered collection. Each index is of a discrete type and the number of dimension in the ordering is fixed.

One dimensional array called *vector* consists of a list of elements distinguished y a single index. For instance, if *v* is a one-dimensional array and *I* is an index value, the v_i refer to the *i*th element of *v.* If the index ranges from L through U then the L is called the *lower bound* of *v* and U id the *upper bound.* Usually, in mathematics and often in mathematical computing the index type is taken as integer and the lower bound is taken as one.

In a two-dimensional array, or matrix, the elements are ordered in the form of a table comprising a fixed number of rows and a fixed number of columns. Each element in such an array is distinguished by a pair of indexes. The first index gives the row and the second index gives the column of the array in which the elements is located. The element in the *i*th row and *j*th column is called *i,j*th element of the array. If *i* ranges from L1 through U1 and *j* ranges from L2 through

U2 then L1 is the first *lower bound* of the array, U1 is the first *upper bound,* L2 is the second *lower bound* and U2 is the second *upper bound.*

Again it is common practice to take the indexes as integers and to set both L1 and L2 equal to one. An example of two-dimensional array with U1 = m, U2 = n is give in the diagram below;

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ : & : & \dots & : \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

Two-dimensional array here

List

List is a finite ordered sequence of items (x_1, x_2, …, x_n), where $n \geq 0$. If $n = 0$, the list has no elements and is called *null list* (or empty list). If $n > 0$, the list has at least one element, x_1, which is called the head of the list. The list consisting of the remaining items is called the *tail* of the original list. The tail of the null list is the null list, as the tail of a list containing only one element.

The items in a list can be arbitrary in nature, unless stated otherwise. In particular it is possible for an item to be in another list, in this case it is known as a sublist. For example, let L be the lists (A, B, (C, D), E), then the third list of L is the list (C, D), which is a sublist of L. If a list has one or more sublist it is called a list structure. If it has no sublist it is a *linear list.* The two basic representation forms for lists are sequentially allocated lists and linked lists, the latter being more flexible.

WEEK 12

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

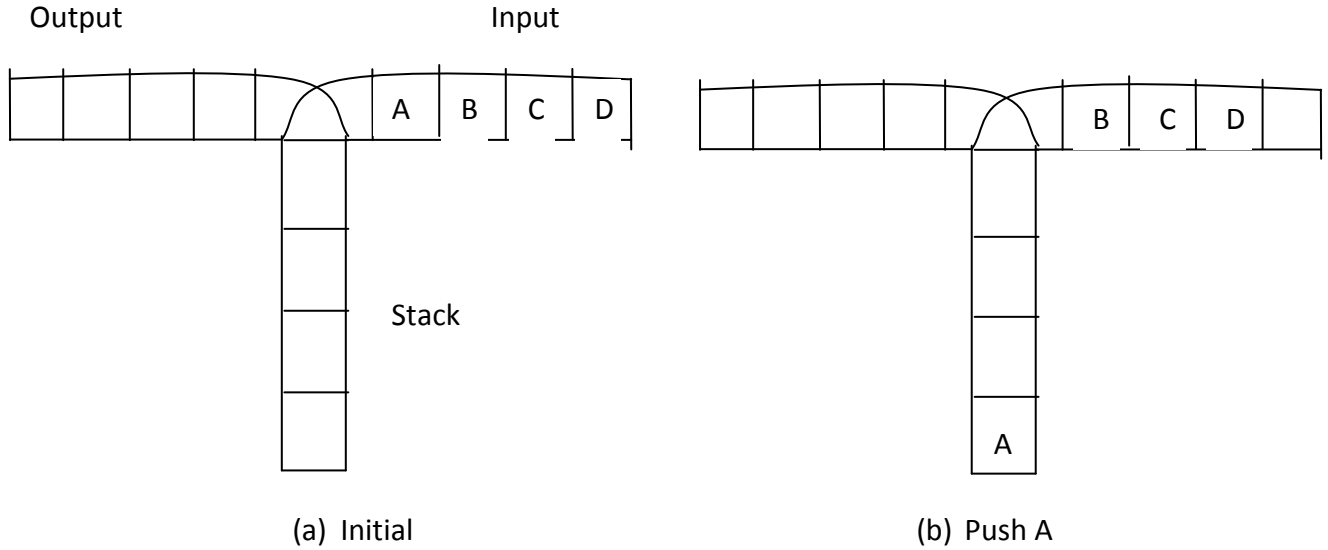
- ✓ What is a stack
- ✓ What is queue
- ✓ Compare stack and queue
- ✓ Examples of file processing techniques
  - o Batch
  - o Real-time
  - o On-line
  - o Serial
  - o Sequential
  - o Indexed sequential
  - o random

Stack

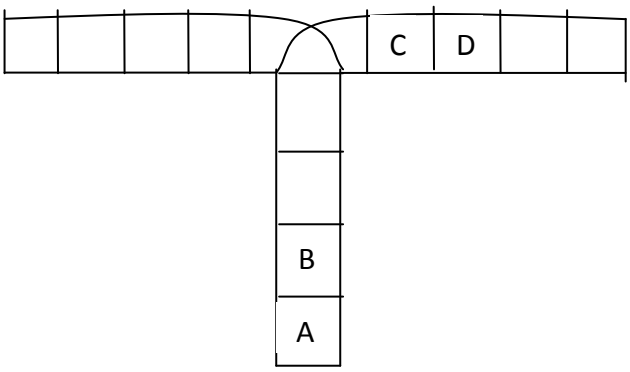
A stack is a linear list that can be accessed for either input or output at just one of its two ends. In stack operations, all accesses involving insertions and removals are made at one end of the list, called ***top***. This implies access on a last in first out (LIFO) basis where the most recently inserted item on the list is the first to be removed.

The operations ***push*** and ***pop*** refer respectively to the insertion and removal of items at the top of the stack. Stacks occur frequently in computing and in particular are closely associated with recursion.

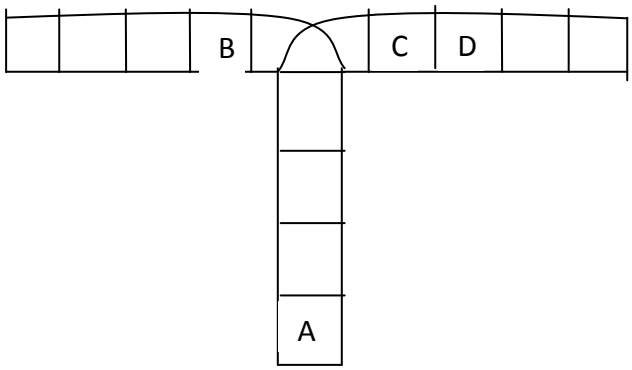
One example of this model of access can be found in a stack of plates on a kitchen shelf or in a spring-loaded dispenser in a cafeteria. In both cases, the only two logical possibilities are to add a plate to the top of the stack. A stack is also exemplified by a railroad spur. In this model, we can insert a boxcar from the input to the open end of the spur, and we can remove a boxcar from the open end of the spur to the output, missing insertion and deletion as wish. The essence of these examples is that the next object to be removed will always be the last one that was added, hence the acronym LIFO or Last-in First-out. Stacks illustrated by railroad spur as in diagrams below;



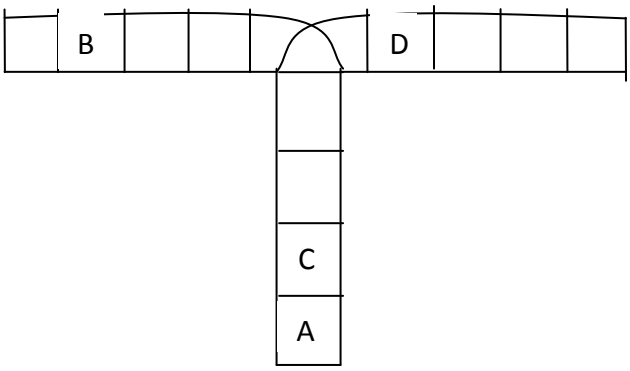
(a) Initial (b) Push A



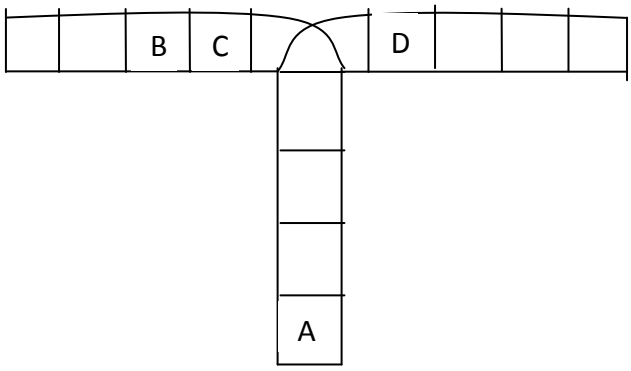
(c) Push B



(d) Pop B



(e) Push C



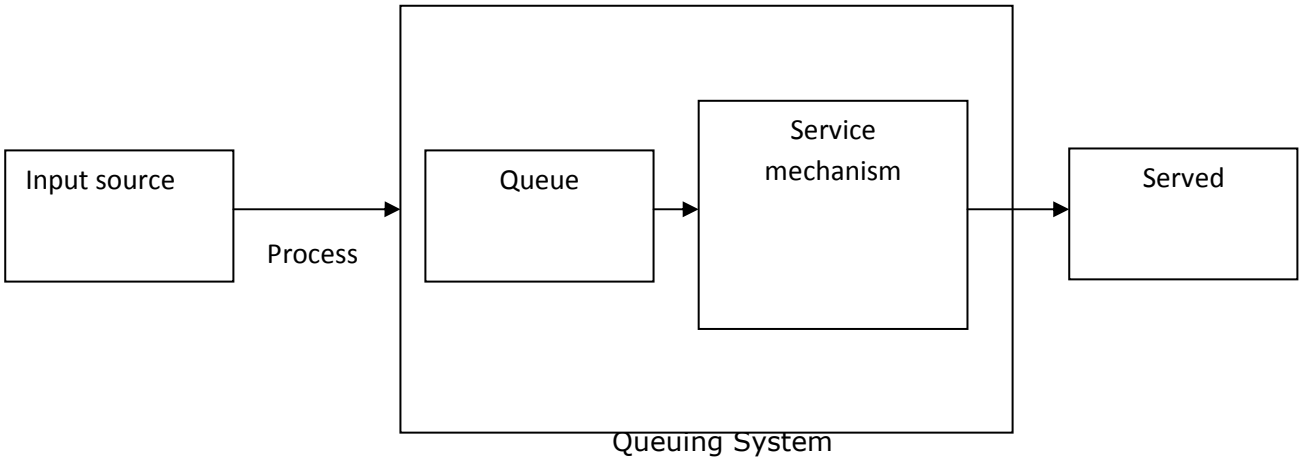
(f) Pop C

Figures showing: A Railroad Model of a

QUEUE

Queue is concerned with the most useful handling of various waiting-line situations, from airplanes waiting to land to computer programs filed for processing. The field arose from telephone networking studies in the early 20th century, and deals with such factors as the pattern of arrivals in a queue, the varying needs of each arrival, and the resulting probabilities and statistical patterns of processing times.

The basic process assumed by most queuing models is that processes requiring service are generated over a period of time by a queue. At certain times, a number of queues are selected for service by some rules known as "service discipline". The required service is then performed for the process by the service mechanism, after which the process leaves the system. This queue process is illustrated in the diagram below;



COMPARE STACK AND QUEUE

File Processing Techniques

Batch: Batch Processing is referred to as running of a batch file - a stored "batch" of operating-system commands carried out one after the other without user intervention; on larger computers, the process of acquiring programs and data sets from users, running them one or a few at a time, and then providing the results to the users. Batch processing can also refer to the process of storing transactions for a period of time before they are posted to a master file, typically in a separate operation undertaken at night.

Real-Time: This is a file processing technique whereby transactions' data are immediately processed as the events to which they relate occur. The need for real-time processing arises when the effective operation of some business can only be attained by the provision of up-to-date information on request without delay. In a real-time system, there is immediate response to the users' requests.

Real-time is suitable when it is necessary to have the latest possible information on;

- Airline seat availability and reservation
- Status of production orders in manufacturing industries
- Wholesale suppliers and manufacturers availability of stock
- Industrial process control

It's worth to mention that every real-time system is communication oriented and provide for random enquiries from remote locations with instantaneous response, because of the characteristics, real-time operation is also on-line or conversational processing.

Online: In computer science when file is activated and ready for operation; capable of communicating with or being controlled by a computer. For example, a printer is online when it can be used for printing; a database is online when it can be used by a person who connects with the computer on which it is stored. *See also* Offline.

Serial: In computer science, is reference to data transfer, serial transmission is the sending of information one bit at a time over a single wire, as through a serial port. In other aspects of computing, serial (or sequential) access refers to finding information based on the location of the last item found; a serial computer is one with a single arithmetic logic unit; serial addition is digit-by-digit addition of the sort people usually perform (as opposed to parallel addition, in which all digits are added at the same time).

Sequential access:

The accessing of pieces of information will be in a serial order, one after the other; therefore the time to access a particular piece of information depends upon which piece of information was last accessed. Such characteristic is typical of off-line storage.

Indexed sequential:

Indexed sequential access: This is an access method for a sequentially organized file whose records are indexed with their corresponding address. This access method supports both sequential access and indexed access and the two types of access that can be comfortably done in this method are sequential access and random access. Note that this method is only possible with disk files.

i. **Sequential access**: The access of the index sequential file makes minimal use of the index. The records can be accessed serially as they are ordered one after another and from the first to the last record in the file.
ii. **Random access:** The random access of the index sequential makes full use of the index to access the records using the record address associated with the index. The records are accessed without any defined sequence of sorting. The method is used when the transaction is to be processed immediately without waiting for sorting and its widely used in commercial data processing because it can be allow both sequential and random methods of access.

Random access: Random access is a type of memory access in which storage location can be accessed in any order. This means that a file is said to be

randomly accessed if the sequence of transactions submit to it does not match any sequence in which records may be organized.

WEEK 13

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ File security techniques
  - o Backup
  - o Encryption
  - o Approved users
  - o Passwords
  - o Firewalls
  - o Intrusion detection systems
  - o Application safeguards
  - o Disaster recovery plans

CONCEPT OF FILE SECURITY

Computer Security, techniques developed to safeguard information and information systems stored on computers. Potential threats include the destruction of computer hardware and software and the loss, modification, theft, unauthorized use, observation, or disclosure of computer data.

Computers and the information they contain are often considered confidential systems because their use is typically restricted to a limited number of users. This confidentiality can be compromised in a variety of ways. For example, computers and computer data can be harmed by people who spread computer viruses and worms. A computer virus is a set of computer program instructions that attaches itself to programs in other computers. The viruses are often parts of documents that are transmitted as attachments to e-mail messages. A worm is similar to a virus but is a self-contained program that transports itself from one computer to another through networks. Thousands of viruses and worms exist and can quickly contaminate millions of computers.

People who intentionally create viruses are computer experts often known as hackers. Hackers also violate confidentiality by observing computer monitor screens and by impersonating authorized users of computers in order to gain access to the users' computers. They invade computer databases to steal the identities of other people by obtaining private, identifying information about them. Hackers also engage in software piracy and deface Web sites on the Internet. For example, they may insert malicious or unwanted messages on a Web site, or alter graphics on the site. They gain access to Web sites by impersonating Web site managers.

Malicious hackers are increasingly developing powerful software crime tools such as automatic computer virus generators, Internet eavesdropping sniffers,

password guessers, vulnerability testers, and computer service saturators. For example, an Internet eavesdropping sniffer intercepts Internet messages sent to other computers. A password guesser tries millions of combinations of characters in an effort to guess a computer's password. Vulnerability testers look for software weaknesses. These crime tools are also valuable security tools used for testing the security of computers and networks.

An increasingly common hacker tool that has gained widespread public attention is the computer service saturator, used in denial-of-service attacks, which can shut down a selected or targeted computer on the Internet by bombarding the computer with more requests than it can handle. This tool first searches for vulnerable computers on the Internet where it can install its own software program. Once installed, the compromised computers act like "zombies" sending usage requests to the target computer. If thousands of computers become infected with the software, then all would be sending usage requests to the target computer, overwhelming its ability to handle the requests for service.

A variety of simple techniques can help prevent computer crimes, such as protecting computer screens from observation, keeping printed information and computers in locked facilities, backing up copies of data files and software, and clearing desktops of sensitive information and materials. Increasingly, however, more sophisticated methods are needed to prevent computer crimes. These include using encryption techniques, establishing software usage permissions, mandating passwords, and installing firewalls and intrusion detection systems. In addition, controls within application systems and disaster recovery plans are also necessary.

FILE SECURITY TECHNIQUES

BACKUP

Storing backup copies of software and data or files and having backup computer and communication capabilities are important basic safeguards because the data can then be restored if it was altered or destroyed by a computer crime or accident. Computer data should be backed up frequently and should be stored nearby in secure locations in case of damage at the primary site. Transporting sensitive data to storage locations should also be done securely.

ENCRYPTION

Another technique to protect confidential information is encryption. Computer users can scramble information to prevent unauthorized users from accessing it. Authorized users can unscramble the information when needed by using a secret code called a key. Without the key the scrambled information would be impossible or very difficult to unscramble. A more complex form of encryption uses two keys, called the public key and the private key, and a system of double encryption. Each participant possesses a secret, private key and a public key that is known to potential recipients. Both keys are used to encrypt, and matching keys are used to decrypt the message. However, the advantage over the single-key method lies with the private keys, which are never shared and so cannot be intercepted. The public key verifies that the sender is the one who transmitted it. The keys are modified periodically, further hampering unauthorized unscrambling and making the encrypted information more difficult to decipher.

APPROVED USERS

Another technique to help prevent abuse and misuse of computer data is to limit the use of computers and data files to approved persons. Security software can verify the identity of computer users and limit their privileges to use, view, and alter files. The software also securely records their actions to establish accountability. Military organizations give access rights to classified, confidential, secret, or top-secret information according to the corresponding security clearance level of the user. Other types of organizations also classify information and specify different degrees of protection.

PASSWORDS

Passwords are confidential sequences of characters that allow approved persons to make use of specified computers, software, or information. To be effective, passwords must be difficult to guess and should not be found in dictionaries. Effective passwords contain a variety of characters and symbols that are not part of the alphabet. To thwart imposters, computer systems usually limit the number of attempts and restrict the time it takes to enter the correct password.

A more secure method is to require possession and use of tamper-resistant plastic cards with microprocessor chips, known as “smart cards,” which contain a stored password that automatically changes after each use. When a user logs on, the computer reads the card's password, as well as another password entered by the user, and matches these two respectively to an identical card password generated by the computer and the user's password stored in the computer in encrypted form. Use of passwords and 'smart cards' is beginning to be reinforced by biometrics, identification methods that use unique personal characteristics, such as fingerprints, retinal patterns, facial characteristics, or voice recordings.

FIREWALLS

Computers connected to communication networks, such as the Internet, are particularly vulnerable to electronic attack because so many people have access to them. These computers can be protected by using firewall computers or software placed between the networked computers and the network. The firewall examines, filters, and reports on all information passing through the network to ensure its appropriateness. These functions help prevent saturation of input capabilities that otherwise might deny usage to legitimate users, and they ensure that information received from an outside source is expected and does not contain computer viruses.

INTRUSION DETECTION SYSTEMS

Security software called intrusion detection systems may be used in computers to detect unusual and suspicious activity and, in some cases, stop a variety of harmful actions by authorized or unauthorized persons. Abuse and misuse of sensitive system and application programs and data such as password, inventory, financial, engineering, and personnel files can be detected by these systems.

APPLICATION SAFEGUARDS

The most serious threats to the integrity and authenticity of computer information come from those who have been entrusted with usage privileges and yet commit computer fraud. For example, authorized persons may secretly transfer money in financial networks, alter credit histories, sabotage information, or commit bill payment or payroll fraud. Modifying, removing, or misrepresenting existing data threatens the integrity and authenticity of computer information. For example, omitting sections of a bad credit history so that only the good credit history

remains violates the integrity of the document. Entering false data to complete a fraudulent transfer or withdrawal of money violates the authenticity of banking information. These crimes can be prevented by using a variety of techniques. One such technique is *checksumming*. Checksumming sums the numerically coded word contents of a file before and after it is used. If the sums are different, then the file has been altered. Other techniques include authenticating the sources of messages, confirming transactions with those who initiate them, segregating and limiting job assignments to make it necessary for more than one person to be involved in committing a crime, and limiting the amount of money that can be transferred through a computer.

DISASTER RECOVERY PLANS

Organizations and businesses that rely on computers need to institute disaster recovery plans that are periodically tested and upgraded. This is because computers and storage components such as diskettes or hard disks are easy to damage. A computer's memory can be erased or flooding, fire, or other forms of destruction can damage the computer’s hardware. Computers, computer data, and components should be installed in safe and locked facilities.

WEEK 14

THIS WEEK SPECIFIC LEARNING OUTCOMES

To understand:

- ✓ File dumping vs file archiving
  - o File Dumping
  - o File Archiving
- ✓ Operating System

What is file status?

File dumping vs file archiving

File Dumping

File dumping is a file processing activity whereby large numbers of users' files are stored on magnetic disk. This is one of the periodic records of the state of disks that are made on some form of offline storage device. This protects against failure either in hardware or software that can lead to corruption of stored files. In the event of a system error that causes file/information to be lost, the most recently copied version of the file can be reinstated from the dump.

On a large multiuser system, the total volume of stored file means that it may not be practicable to dump all the file on every occasion. In these cases an incremental dump can be taken, containing only those files that are marked as having been altered since the last dump. This reduces the total amount of file to be copied during the dump, allowing dumping to be made frequently.

File Archiving

File Archiving is the collection of historically valuable records, ranging from papers and documents to photographs, films, videotapes, and sound recordings. It is a repository for information that the user wishes to retain, but without requiring immediate access. There are three activities that must be distinguished in archiving; viz

a. The routine checking of back-up copies initiated by the system manager to protect users and system manager against corruption of stored information/file.
b. The autonomous transferring of file from higher-performance to a lower-performance storage system, initiated by the operating system to achieve

economies in the total cost to the system manager of the information storage.

c. The voluntary transferring of a file between normal file storage and archive storage initiated by the user to achieve economies in the total costs to the user of information storage.

OPERATING SYSTEM

An ***Operating System*** (commonly abbreviated *OS* and *O/S*) is the software component of a computer system that is responsible for the management and coordination of activities and the sharing of the resources of the computer. The operating system acts as a host for applications that are run on the machine. As a host, one of the purposes of an operating system is to handle the details of the operation of the hardware. This relieves application programs from having to manage these details and makes it easier to write applications. Almost all computers, including handheld computers, desktop computers, supercomputers, and even video game consoles, use an operating system of some type. Some of the oldest models may however use an embedded operating system, that may be contained on a compact disk or other data storage device.

Operating systems offer a number of services to application programs and users. Applications access these services through application programming interfaces (APIs) or system calls. By invoking these interfaces, the application can request a service from the operating system, pass parameters, and receive the results of the operation. Users may also interact with the operating system with some kind a software user interface (UI) like typing commands by using command line interface (CLI) or using a graphical user interface (GUI, commonly pronounced “gooey”). For hand-held and desktop computers, the user interface is generally considered part of the operating system. On large multi-user systems like Unix

and Unix-like systems, the user interface is generally implemented as an application program that runs outside the operating system. (Whether the user interface should be included as part of the operating system is a point of contention.)

Common contemporary operating systems include Microsoft Windows, Mac OS, Linux and Solaris. Microsoft Windows has a significant majority of market share in the desktop and notebook computer markets, while servers generally run on Linux or other Unix-like systems. Embedded device markets are split amongst several operating systems.

Technology

An operating system is a collection of technologies which are designed to allow the computer to perform certain functions. These technologies may or may not be present in every operating system, and there are often differences in how they are implemented. However as stated above most modern operating systems are derived from common design ancestors, and are therefore basically similar.

Program execution

Executing a program involves the creation of a process by the operating system. The kernel creates a process by setting aside or allocating some memory, loading program code from a disk or another part of memory into the newly allocated space, and starting it running.

Interrupts

Interrupts are central to operating systems as they allow the operating system to deal with the unexpected activities of running programs and the world outside the computer. Interrupt-based programming is one of the most basic forms of time-

sharing, being directly supported by most CPUs. Interrupts provide a computer with a way of automatically running specific code in response to events. Even very basic computers support hardware interrupts, and allow the programmer to specify code which may be run when that event takes place.

When an interrupt is received, the computer's hardware automatically suspends whatever program is currently running by pushing the current state on a stack, and its registers and program counter are also saved. This is analogous to placing a bookmark in a book when someone is interrupted by a phone call. This task requires no operating system as such, but only that the interrupt be configured at an earlier time.

In modern operating systems, interrupts are handled by the operating system's kernel. Interrupts may come from either the computer's hardware, or from the running program. When a hardware device triggers an interrupt, the operating system's kernel decides how to deal with this event, generally by running some processing code, or ignoring it. The processing of hardware interrupts is a task that is usually delegated to software called device drivers, which may be either part of the operating system's kernel, part of another program, or both. Device drivers may then relay information to a running program by various means.

A program may also trigger an interrupt to the operating system, which are very similar in function. If a program wishes to access hardware for example, it may interrupt the operating system's kernel, which causes control to be passed back to the kernel. The kernel may then process the request which may contain instructions to be passed onto hardware, or to a device driver. When a program wishes to allocate more memory, launch or communicate with another program, or signal that it no longer needs the CPU, it does so through interrupts.

Protected mode and supervisor mode

Modern CPUs support something called dual mode operation. CPUs with this capability use two modes: protected mode and supervisor mode, which allow certain CPU functions to be controlled and affected only by the operating system kernel. Here, protected mode does not refer specifically to the 80286 (Intel's x86 16-bit microprocessor) CPU feature, although its protected mode is very similar to it. CPUs might have other modes similar to 80286 protected mode as well, such as the virtual 8086 mode of the 80386 (Intel's x86 32-bit microprocessor or i386).

However, the term is used here more generally in operating system theory to refer to all modes which limit the capabilities of programs running in that mode, providing things like virtual memory addressing and limiting access to hardware in a manner determined by a program running in supervisor mode. Similar modes have existed in supercomputers, minicomputers, and mainframes as they are essential to fully supporting UNIX-like multi-user operating systems.

When a computer first starts up, it is automatically running in supervisor mode. The first few programs to run on the computer, being the BIOS, bootloader and the operating system have unlimited access to hardware. However when the operating system passes control to another program, it can place the CPU into protected mode.

In protected mode, programs may have access to a more limited set of the CPU's instructions. A user program may leave protected mode only by triggering an interrupt, causing control to be passed back to the kernel. In this way the operating system can maintain exclusive control over things like access to hardware and memory.

The term "protected mode resource" generally refers to one or more CPU registers, which contain information that the running program isn't allowed to

alter. Attempts to alter these resources generally causes a switch to supervisor mode.

Memory management

Among other things, a multiprogramming operating system kernel must be responsible for managing all system memory which is currently in use by programs. This ensures that a program does not interfere with memory already used by another program. Since programs time share, each program must have independent access to memory.

Cooperative memory management, used by many early operating systems assumes that all programs make voluntary use of the kernel's memory manager, and do not exceed their allocated memory. This system of memory management is almost never seen anymore, since programs often contain bugs which can cause them to exceed their allocated memory. If a program fails it may cause memory used by one or more other programs to be affected or overwritten. Malicious programs, or viruses may purposefully alter another program's memory or may affect the operation of the operating system itself. With cooperative memory management it takes only one misbehaved program to crash the system.

Memory protection enables the kernel to limit a process' access to the computer's memory. Various methods of memory protection exist, including memory segmentation and paging. All methods require some level of hardware support (such as the 80286 MMU) which doesn't exist in all computers.

In both segmentation and paging, certain protected mode registers specify to the CPU what memory address it should allow a running program to access. Attempts to access other addresses will trigger an interrupt which will cause the

CPU to re-enter supervisor mode, placing the kernel in charge. This is called a segmentation violation or Seg-V for short, and since it is usually a sign of a misbehaving program, the kernel will generally kill the offending program, and report the error.

Windows 3.1-Me had some level of memory protection, but programs could easily circumvent the need to use it. Under Windows 9x all MS-DOS applications ran in supervisor mode, giving them almost unlimited control over the computer. A general protection fault would be produced indicating a segmentation violation had occurred, however the system would often crash anyway.

Virtual memory

The use of virtual memory addressing (such as paging or segmentation) means that the kernel can choose which memory each program may use at any given time, allowing the operating system to use the same memory locations for multiple tasks.

If a program tries to access memory that isn't in its current range of accessible memory, but nonetheless has been allocated to it, the kernel will be interrupted in the same way as it would if the program were to exceed its allocated memory. (See section on memory management.) Under UNIX this kind of interrupt is referred to as a page fault.

When the kernel detects a page fault it will generally adjust the virtual memory range of the program which triggered it, granting it access to the memory requested. This gives the kernel discretionary power over where a particular application's memory is stored, or even whether or not it has actually been allocated yet.

In modern operating systems, application memory which is accessed less frequently can be temporarily stored on disk or other media to make that space available for use by other programs. This is called swapping, as an area of memory can be use by multiple programs, and what that memory area contains can be swapped or exchanged on demand.

Multitasking

Multitasking refers to the running of multiple independent computer programs on the same computer, giving the appearance that it is performing the tasks at the same time. Since most computers can do at most one or two things at one time, this is generally done via time sharing, which means that each program uses a share of the computer's time to execute.

An operating system kernel contains a piece of software called a scheduler which determines how much time each program will spend executing, and in which order execution control should be passed to programs. Control is passed to a process by the kernel, which allows the program access to the CPU and memory. At a later time control is returned to the kernel through some mechanism, so that another program may be allowed to use the CPU. This so-called passing of control between the kernel and applications is called a context switch.

An early model which governed the allocation of time to programs was called cooperative multitasking. In this model, when control is passed to a program by the kernel, it may execute for as long as it wants before explicitly returning control to the kernel. This means that a malfunctioning program may prevent any other programs from using the CPU.

The philosophy governing preemptive multitasking is that of ensuring that all programs are given regular time on the CPU. This implies that all programs must be limited in how much time they are allowed to spend on the CPU without being interrupted. To accomplish this, modern operating system kernels make use of a timed interrupt. A protected mode timer is set by the kernel which triggers a return to supervisor mode after the specified time has elapsed. (See above sections on Interrupts and Dual Mode Operation.)

On many single user operating systems cooperative multitasking is perfectly adequate, as home computers generally run a small number of well tested programs. Windows NT was the first version of Microsoft Windows which enforced preemptive multitasking, but it didn't reach the home user market until Windows XP, (since Windows NT was targeted at professionals.)

Disk access and file systems

Access to files stored on disks is a central feature of all operating systems. Computers store data on disks using files, which are structured in specific ways in order to allow for faster access, higher reliability, and to make better use out of the drive's available space. The specific way in which files are stored on a disk is called a file system, and enables files to have names and attributes. It also allows them to be stored in a hierarchy of directories or folders arranged in a directory tree.

Early operating systems generally supported a single type of disk drive and only one kind of file system. Early file systems were limited in their capacity, speed, and in the kinds of file names and directory structures they could use. These limitations often reflected limitations in the operating systems they were designed for, making it very difficult for an operating system to support more than one file system.

While many simpler operating systems support a limited range of options for accessing storage systems, more modern operating systems like UNIX and Linux support a technology known as a virtual file system or VFS. A modern operating system like UNIX supports a wide array of storage devices, regardless of their design or file systems to be accessed through a common application programming interface (API). This makes it unnecessary for programs to have any knowledge about the device they are accessing. A VFS allows the operating system to provide programs with access to an unlimited number of devices with an infinite variety of file systems installed on them through the use of specific device drivers and file system drivers.

A connected storage device such as a hard drive is accessed through a device driver. The device driver understands the specific language of the drive and is able to translate that language into a standard language used by the operating system to access all disk drives. On UNIX this is the language of block devices.

When the kernel has an appropriate device driver in place, it can then access the contents of the disk drive in raw format, which may contain one or more file systems. A file system driver is used to translate the commands used to access each specific file system into a standard set of commands that the operating system can use to talk to all file systems. Programs can then deal with these file systems on the basis of filenames, and directories/folders, contained within a hierarchical structure. They can create, delete, open, and close files, as well as gather various information about them, including access permissions, size, free space, and creation and modification dates.

Various differences between file systems make supporting all file systems difficult. Allowed characters in file names, case sensitivity, and the presence of various kinds of file attributes makes the implementation of a single interface for every file system a daunting task. Operating systems tend to recommend the use

of (and so support natively) file systems specifically designed for them; for example, NTFS in Windows and ext*n* and ReiserFS in Linux. However, in practice, third party drives are usually available to give support for the most widely used filesystems in most general-purpose operating systems (for example, NTFS is available in Linux through NTFS-3g, and ext2/3 and ReiserFS are available in Windows through FS-driver and rfstool).

Device drivers

A device driver is a specific type of computer software developed to allow interaction with hardware devices. Typically this constitutes an interface for communicating with the device, through the specific computer bus or communications subsystem that the hardware is connected to, providing commands to and/or receiving data from the device, and on the other end, the requisite interfaces to the operating system and software applications. It is a specialized hardware-dependent computer program which is also operating system specific that enables another program, typically an operating system or applications software package or computer program running under the operating system kernel, to interact transparently with a hardware device, and usually provides the requisite interrupt handling necessary for any necessary asynchronous time-dependent hardware interfacing needs.

The key design goal of device drivers is abstraction. Every model of hardware (even within the same class of device) is different. Newer models also are released by manufacturers that provide more reliable or better performance and these newer models are often controlled differently. Computers and their operating systems cannot be expected to know how to control every device, both now and in the future. To solve this problem, OSes essentially dictate how every type of device should be controlled. The function of the device driver is then to translate these OS mandated function calls into device specific calls. In theory a

new device, which is controlled in a new manner, should function correctly if a suitable driver is available. This new driver will ensure that the device appears to operate as usual from the operating systems' point of view for any person.

WEEK 15

Networking

Currently most operating systems support a variety of networking protocols, hardware, and applications for using them. This means that computers running dissimilar operating systems can participate in a common network for sharing resources such as computing, files, printers, and scanners using either wired or wireless connections. Networks can essentially allow a computer's operating system to access the resources of a remote computer to support the same functions as it could if those resources were connected directly to the local computer. This includes everything from simple communication, to using networked file systems or even sharing another computer's graphics or sound hardware. Some network services allow the resources of a computer to be accessed transparently, such as SSH which allows networked users direct access to a computer's command line interface.

Client/server networking involves a program on a computer somewhere which connects via a network to another computer, called a server. Servers, usually running UNIX or Linux, offer (or host) various services to other network computers and users. These services are usually provided through ports or numbered access points beyond the server's network address. Each port number is usually associated with a maximum of one running program, which is responsible for handling requests to that port. A daemon, being a user program, can in turn access the local hardware resources of that computer by passing requests to the operating system kernel.

Many operating systems support one or more vendor-specific or open networking protocols as well, for example, SNA on IBM systems, DECnet on systems from Digital Equipment Corporation, and Microsoft-specific protocols (SMB) on Windows. Specific protocols for specific tasks may also be supported such as NFS for file access. Protocols like ESound, or esd can be easily extended over the network to provide sound from local applications, on a remote system's sound hardware.

Security

A computer being secure depends on a number of technologies working properly. A modern operating system provides access to a number of resources, which are available to software running on the system, and to external devices like networks via the kernel.

The operating system must be capable of distinguishing between requests which should be allowed to be processed, and others which should not be processed. While some systems may simply distinguish between "privileged" and "non-privileged", systems commonly have a form of requester *identity*, such as a user name. To establish identity there may be a process of *authentication*. Often a username must be quoted, and each username may have a password. Other methods of authentication, such as magnetic cards or biometric data, might be used instead. In some cases, especially connections from the network, resources may be accessed with no authentication at all (such as reading files over a network share).

In addition to the allow/disallow model of security, a system with a high level of security will also offer auditing options. These would allow tracking of requests for access to resources (such as, "who has been reading this file?"). Internal security, or security from an already running program is only possible if all

possibly harmful requests must be carried out through interrupts to the operating system kernel. If programs can directly access hardware and resources, they cannot be secured.

External security involves a request from outside the computer, such as a login at a connected console or some kind of network connection. External requests are often passed through device drivers to the operating system's kernel, where they can be passed onto applications, or carried out directly. Security of operating systems has long been a concern because of highly sensitive data held on computers, both of a commercial and military nature. The United States Government Department of Defense (DoD) created the *Trusted Computer System Evaluation Criteria* (TCSEC) which is a standard that sets basic requirements for assessing the effectiveness of security. This became of vital importance to operating system makers, because the TCSEC was used to evaluate, classify and select computer systems being considered for the processing, storage and retrieval of sensitive or classified information.

Network services include offerings such as file sharing, print services, email, web sites, and file transfer protocols (FTP), most of which can have compromised security. At the front line of security are hardware devices known as firewalls or intrusion detection/prevention systems. At the operating system level, there are a number of software firewalls available, as well as intrusion detection/prevention systems. Most modern operating systems include a software firewall, which is enabled by default. A software firewall can be configured to allow or deny network traffic to or from a service or application running on the operating system. Therefore, one can install and be running an insecure service, such as Telnet or FTP, and not have to be threatened by a security breach because the firewall would deny all traffic trying to connect to the service on that port.

An alternative strategy, and the only sandbox strategy available in systems that do not meet the Popek and Goldberg virtualization requirements, is the operating system not running user programs as native code, but instead either emulates a processor or provides a host for a p-code based system such as Java.

Internal security is especially relevant for multi-user systems; it allows each user of the system to have private files that the other users cannot tamper with or read. Internal security is also vital if auditing is to be of any use, since a program can potentially bypass the operating system, inclusive of bypassing auditing.

Examples:

Microsoft Windows

While the Windows 9x series offered the option of having profiles for multiple users, they had no concept of access privileges, and did not allow concurrent access; and so were not true multi-user operating systems. In addition, they implemented only partial memory protection. They were accordingly widely criticised for lack of security.

The Windows NT series of operating systems, by contrast, are true multi-user, and implement absolute memory protection. However, a lot of the advantages of being a true multi-user operating system were nullified by the fact that, prior to Windows Vista, the first user account created during the setup process was an administrator account, which was also the default for new accounts. Though Windows XP did have limited accounts, the majority of home users did not change to an account type with fewer rights -- partially due to the number of programs which unnecessarily required administrator rights -- and so most home users ran as administrator all the time.

Windows Vista changes this[3] by introducing a privilege elevation system called User Account Control. When logging in as a standard user, a logon session is created and a token containing only the most basic privileges is assigned. In this way, the new logon session is incapable of making changes that would affect the entire system. When logging in as a user in the Administrators group, two separate tokens are assigned. The first token contains all privileges typically awarded to an administrator, and the second is a restricted token similar to what a standard user would receive. User applications, including the Windows Shell, are then started with the restricted token, resulting in a reduced privilege environment even under an Administrator account. When an application requests higher privileges or "Run as administrator" is clicked, UAC will prompt for confirmation and, if consent is given (including administrator credentials if the account requesting the elevation is not a member of the administrators group), start the process using the unrestricted token.[4]

Linux/Unix

Linux and UNIX both have two tier security, which limits any system-wide changes to the root user, a special user account on all UNIX-like systems. While the root user has virtually unlimited permission to affect system changes, programs running as a regular user are limited in where they can save files, what hardware they can access, etc. In many systems, a user's memory usage, their selection of available programs, their total disk usage or quota, available range of programs' priority settings, and other functions can also be locked down. This provides the user with plenty of freedom to do what needs done, without being able to put any part of the system in jeopardy (barring accidental triggering of system-level bugs) or make sweeping, system-wide changes. The user's settings are stored in an area of the computer's file system called the user's home directory, which is also provided as a location where the user may store their

work, similar to My Documents on a Windows system. Should a user have to install software or make system-wide changes, they must become the root user temporarily, usually with the su command, which is answered with the computer's root password when prompted. Some systems (such as Ubuntu and its derivatives) are configured by default to allow select users to run programs as the root user via the sudo command, using the user's own password for authentication instead of the system's root password. One is sometimes said to "go root" when elevating oneself to root access.

File system support in modern operating systems

Support for file systems is highly varied among modern operating systems although there are several common file systems which almost all operating systems include support and drivers for.

Linux and UNIX

Many Linux distributions support some or all of ext2, ext3, ReiserFS, Reiser4, JFS , XFS , GFS, GFS2, OCFS, OCFS2, and NILFS. The ext file systems, namely ext2 and ext3 are based on the original Linux file system. Others have been developed by companies to meet their specific needs, hobbyists, or adapted from UNIX, Microsoft Windows, and other operating systems. Linux has full support for XFS and JFS, along with FAT (the MS-DOS file system), and HFS which is the primary file system for the Macintosh.

In recent years support for Microsoft Windows NT's NTFS file system has appeared in Linux, and is now comparable to the support available for other native UNIX file systems. ISO 9660 and UDF are supported which are standard file systems used on CDs, DVDs, and BluRay discs. It is possible to install Linux on the majority of these file systems. Unlike other operating systems, Linux and

UNIX allow any file system to be used regardless of the media it is stored on, whether it is a hard drive, CD or DVD, or even contained within a file located on an another file system.

Microsoft Windows

Microsoft Windows presently supports NTFS and FAT file systems, along with network file systems shared from other computers, and the ISO 9660 and UDF filesystems used for CDs, DVDs, and other optical discs such as BluRay. Under Windows each file system is usually limited in application to certain media, for example CDs must use ISO 9660 or UDF, and as of Windows Vista, NTFS is the only file system which the operating system can be installed on. Details of its design are not known. Windows Embedded CE 6.0 introduced ExFAT, a file system more suitable for flash drives.

Mac OS X

Mac OS X supports HFS+ with journaling as its primary file system. It is derived from the Hierarchical File System of the earlier Mac OS. Mac OS X has facilities to read and write FAT, NTFS, UDF, and other file systems, but cannot be installed to them. Due to its UNIX heritage Mac OS X now supports virtually all the file systems supported by the UNIX VFS. Recently Apple Inc. started work on porting Sun Microsystem's ZFS filesystem to Mac OS X and preliminary support is already available in Mac OS X 10.5.

Special purpose file systems

FAT file systems are commonly found on floppy discs, flash memory cards, digital cameras, and many other portable devices because of their relative

simplicity. Performance of FAT compares poorly to most other file systems as it uses overly simplistic data structures, making file operations time-consuming, and makes poor use of disk space in situations where many small files are present. ISO 9660 and Universal Disk Format are two common formats that target Compact Discs and DVDs. Mount Rainier is a newer extension to UDF supported by Linux 2.6 kernels and Windows Vista that facilitates rewriting to DVDs in the same fashion as has been possible with floppy disks.

Journalized file systems

File systems may provide journaling, which provides safe recovery in the event of a system crash. A journaled file system writes some information twice: first to the journal, which is a log of file system operations, then to its proper place in the ordinary file system. Journaling is handled by the file system driver, and keeps track of each operation taking place that changes the contents of the disk. In the event of a crash, the system can recover to a consistent state by replaying a portion of the journal. Many UNIX file systems provide journaling including ReiserFS, JFS, and Ext3.

In contrast, non-journaled file systems typically need to be examined in their entirety by a utility such as fsck or chkdsk for any inconsistencies after an unclean shutdown. Soft updates is an alternative to journaling that avoids the redundant writes by carefully ordering the update operations. Log-structured file systems and ZFS also differ from traditional journaled file systems in that they avoid inconsistencies by always writing new copies of the data, eschewing in-place updates.

Graphical user interfaces

Most modern computer systems support graphical user interfaces (GUI), and often include them. In some computer systems, such as the original implementations of Microsoft Windows and the Mac OS, the GUI is integrated into the kernel.

While technically a graphical user interface is not an operating system service, incorporating support for one into the operating system kernel can allow the GUI to be more responsive by reducing the number of context switches required for the GUI to perform its output functions. Other operating systems are modular, separating the graphics subsystem from the kernel and the Operating System. In the 1980s UNIX, VMS and many others had operating systems that were built this way. Linux and Mac OS X are also built this way. Modern releases of Microsoft Windows such as Windows Vista implement a graphics subsystem that is mostly in user-space, however versions between Windows NT 4.0 and Windows Server 2003's graphics drawing routines exist mostly in kernel space. Windows 9x had very little distinction between the interface and the kernel.

Many computer operating systems allow the user to install or create any user interface they desire. The X Window System in conjunction with GNOME or KDE is a commonly-found setup on most Unix and Unix-like (BSD, Linux, Minix) systems. A number of Windows shell replacements have been released for Microsoft Windows, which offer alternatives to the included Windows shell, but the shell itself cannot be separated from Windows.

Numerous Unix-based GUIs have existed over time, most derived from X11. Competition among the various vendors of Unix (HP, IBM, Sun) led to much fragmentation, though an effort to standardize in the 1990s to COSE and CDE failed for the most part due to various reasons, eventually eclipsed by the widespread adoption of GNOME and KDE. Prior to open source-based toolkits

and desktop environments, Motif was the prevalent toolkit/desktop combination (and was the basis upon which CDE was developed).

Graphical user interfaces evolve over time. For example, Windows has modified its user interface almost every time a new major version of Windows is released, and the Mac OS GUI changed dramatically with the introduction of Mac OS X in 2001.

History

The first computers did not have operating systems. By the early 1960s, commercial computer vendors were supplying quite extensive tools for streamlining the development, scheduling, and execution of jobs on batch processing systems. Examples were produced by UNIVAC and Control Data Corporation, amongst others.

The operating systems originally deployed on mainframes, and, much later, the original microcomputer operating systems, only supported one program at a time, requiring only a very basic scheduler. Each program was in complete control of the machine while it was running. Multitasking (timesharing) first came to mainframes in the 1960s.

In 1969-70, UNIX first appeared on the PDP-7 and later the PDP-11. It soon became capable of providing cross-platform time sharing using preemptive multitasking, advanced memory management, memory protection, and a host of other advanced features. UNIX soon gained popularity as an operating system for mainframes and minicomputers alike.

MS-DOS provided many operating system like features, such as disk access. However many DOS programs bypassed it entirely and ran directly on hardware. IBM's version, PC-DOS, ran on IBM microcomputers, including the IBM PC and

the IBM PC XT, and MS-DOS came into widespread use on clones of these machines.

IBM PC compatibles could also run Microsoft Xenix, a UNIX-like operating system from the early 1980s. Xenix was heavily marketed by Microsoft as a multi-user alternative to its single user MS-DOS operating system. The CPUs of these personal computers could not facilitate kernel memory protection or provide dual mode operation, so Microsoft Xenix relied on cooperative multitasking and had no protected memory.

The 80286-based IBM PC AT was the first computer technically capable of using dual mode operation, and providing memory protection.

Classic Mac OS, and Microsoft Windows 1.0-3.11 supported only cooperative multitasking (Windows 95, 98, & ME supported preemptive multitasking only when running 32 bit applications, but ran legacy 16 bit applications using cooperative multitasking), and were very limited in their abilities to take advantage of protected memory. Application programs running on these operating systems must yield CPU time to the scheduler when they are not using it, either by default, or by calling a function.

Windows NT's underlying operating system kernel which was a designed by essentially the same team as Digital Equipment Corporation's VMS, a UNIX-like operating system which provided protected mode operation for all user programs, kernel memory protection, preemptive multi-tasking, virtual file system support, and a host of other features.

Classic AmigaOS and versions of Microsoft Windows from Windows 1.0 through Windows Me did not properly track resources allocated by processes at runtime.

If a process had to be terminated, the resources might not be freed up for new programs until the machine was restarted.

The AmigaOS did have preemptive multitasking.

Mainframes

Through the 1960s, many major features were pioneered in the field of operating systems. The development of the IBM System/360 produced a family of mainframe computers available in widely differing capacities and price points, for which a single operating system OS/360 was planned (rather than developing ad-hoc programs for every individual model). This concept of a single OS spanning an entire product line was crucial for the success of System/360 and, in fact, IBM`s current mainframe operating systems are distant descendants of this original system; applications written for the OS/360 can still be run on modern machines. In the mid-70's, the MVS, the descendant of OS/360 offered the first[*citation needed*] implementation of using RAM as a transparent cache for disk resident data.

OS/360 also pioneered a number of concepts that, in some cases, are still not seen outside of the mainframe arena. For instance, in OS/360, when a program is started, the operating system keeps track of all of the system resources that are used including storage, locks, data files, and so on. When the process is terminated for any reason, all of these resources are re-claimed by the operating system. An alternative CP-67 system started a whole line of operating systems focused on the concept of virtual machines.

Control Data Corporation developed the SCOPE operating system in the 1960s, for batch processing. In cooperation with the University of Minnesota, the KRONOS and later the NOS operating systems were developed during the

1970s, which supported simultaneous batch and timesharing use. Like many commercial timesharing systems, its interface was an extension of the Dartmouth BASIC operating systems, one of the pioneering efforts in timesharing and programming languages. In the late 1970s, Control Data and the University of Illinois developed the PLATO operating system, which used plasma panel displays and long-distance time sharing networks. Plato was remarkably innovative for its time, featuring real-time chat, and multi-user graphical games.

Burroughs Corporation introduced the B5000 in 1961 with the MCP, (Master Control Program) operating system. The B5000 was a stack machine designed to exclusively support high-level languages with no machine language or assembler, and indeed the MCP was the first OS to be written exclusively in a high-level language – ESPOL, a dialect of ALGOL. MCP also introduced many other ground-breaking innovations, such as being the first commercial implementation of virtual memory. MCP is still in use today in the Unisys ClearPath/MCP line of computers.

UNIVAC, the first commercial computer manufacturer, produced a series of EXEC operating systems. Like all early main-frame systems, this was a batch-oriented system that managed magnetic drums, disks, card readers and line printers. In the 1970s, UNIVAC produced the Real-Time Basic (RTB) system to support large-scale time sharing, also patterned after the Dartmouth BASIC system.

General Electric and MIT developed General Electric Comprehensive Operating Supervisor (GECOS), which introduced the concept of ringed security privilege levels. After acquisition by Honeywell it was renamed to General Comprehensive Operating System (GCOS).

Digital Equipment Corporation developed many operating systems for its various computer lines, including TOPS-10 and TOPS-20 time sharing systems for the 36-bit PDP-10 class systems. Prior to the widespread use of UNIX, TOPS-10 was a particularly popular system in universities, and in the early ARPANET community.

In the late 1960s through the late 1970s, several hardware capabilities evolved that allowed similar or ported software to run on more than one system. Early systems had utilized microprogramming to implement features on their systems in order to permit different underlying architecture to appear to be the same as others in a series. In fact most 360's after the 360/40 (except the 360/165 and 360/168) were microprogrammed implementations. But soon other means of achieving application compatibility were proven to be more significant.

The enormous investment in software for these systems made since 1960s caused most of the original computer manufacturers to continue to develop compatible operating systems along with the hardware. The notable supported mainframe operating systems include:

- Burroughs MCP -- B5000,1961 to Unisys Clearpath/MCP, present.
- IBM OS/360 -- IBM System/360, 1966 to IBM z/OS, present.
- IBM CP-67 -- IBM System/360, 1967 to IBM z/VM, present.
- UNIVAC EXEC 8 -- UNIVAC 1108, 1964, to Unisys Clearpath IX, present.

Microcomputers

The first microcomputers did not have the capacity or need for the elaborate operating systems that had been developed for mainframes and minis; minimalistic operating systems were developed, often loaded from ROM and known as *Monitors*. One notable early disk-based operating system was CP/M,

which was supported on many early microcomputers and was closely imitated in MS-DOS, which became wildly popular as the operating system chosen for the IBM PC (IBM's version of it was called IBM-DOS or PC-DOS), its successors making Microsoft one of the world's most profitable companies. In the 80's Apple Computer Inc. (now Apple Inc.) abandoned its popular Apple II series of microcomputers to introduce the Apple Macintosh computer with the an innovative Graphical User Interface (GUI) to the Mac OS.

The introduction of the Intel 80386 CPU chip with 32-bit architecture and paging capabilities, provided personal computers with the ability to run multitasking operating systems like those of earlier minicomputers and mainframes. Microsoft's responded to this progress by hiring Dave Cutler, who had developed the VMS operating system for Digital Equipment Corporation. He would lead the development of the Windows NT operating system, which continues to serve as the basis for Microsoft's operating systems line. Steve Jobs, a co-founder of Apple Inc., started NeXT Computer Inc., which developed the Unix-like NEXTSTEP operating system. NEXTSTEP would later be acquired by Apple Inc. and used, along with code from FreeBSD as the core of Mac OS X.

Minix, an academic teaching tool which could be run on early PCs, would inspire another reimplementation of Unix, called Linux. Started by computer student Linus Torvalds with cooperation from volunteers over the internet, developed a kernel which was combined with the tools from the GNU Project. The Berkeley Software Distribution, known as BSD, is the UNIX derivative distributed by the University of California, Berkeley, starting in the 1970s. Freely distributed and ported to many minicomputers, it eventually also gained a following for use on PCs, mainly as FreeBSD, NetBSD and OpenBSD.

Examples

Microsoft Windows

The Microsoft Windows family of operating systems originated as an add-on to the older MS-DOS operating system for the IBM PC. Modern versions are based on the newer Windows NT kernel that was originally intended for OS/2. Windows runs on x86, x86-64 and Itanium processors. Earlier versions also ran on the DEC Alpha, MIPS, Fairchild (later Intergraph) Clipper and PowerPC architectures (some work was done to port it to the SPARC architecture).

As of June 2008, Microsoft Windows holds a large amount of the worldwide desktop market share. Windows is also used on servers, supporting applications such as web servers and database servers. In recent years, Microsoft has spent significant marketing and research & development money to demonstrate that Windows is capable of running any enterprise application, which has resulted in consistent price/performance records (see the TPC) and significant acceptance in the enterprise market.

The most widely used version of the Microsoft Windows family is Windows XP, released on October 25, 2001.

In November 2006, after more than five years of development work, Microsoft released Windows Vista, a major new operating system version of Microsoft Windows family which contains a large number of new features and architectural changes. Chief amongst these are a new user interface and visual style called Windows Aero, a number of new security features such as User Account Control, and few new multimedia applications such as Windows DVD Maker.

Microsoft has announced a new version codenamed Windows 7 will be released in late 2009 - mid 2010

Plan 9

Ken Thompson, Dennis Ritchie and Douglas McIlroy at Bell Labs designed and developed the C programming language to build the operating system Unix. Programmers at Bell Labs went on to develop Plan 9 and Inferno, which were engineered for modern distributed environments. Plan 9 was designed from the start to be a networked operating system, and had graphics built-in, unlike Unix, which added these features to the design later. Plan 9 has yet to become as popular as Unix derivatives, but it has an expanding community of developers. It is currently released under the Lucent Public License. Inferno was sold to Vita Nuova Holdings and has been released under a GPL/MIT license.

Unix and Unix-like operating systems

Ken Thompson wrote B, mainly based on BCPL, which he used to write Unix, based on his experience in the MULTICS project. B was replaced by C, and Unix developed into a large, complex family of inter-related operating systems which have been influential in every modern operating system (see History).

The *Unix-like* family is a diverse group of operating systems, with several major sub-categories including System V, BSD, and Linux. The name "UNIX" is a trademark of The Open Group which licenses it for use with any operating system that has been shown to conform to their definitions. "Unix-like" is commonly used to refer to the large set of operating systems which resemble the original Unix.

Unix-like systems run on a wide variety of machine architectures. They are used heavily for servers in business, as well as workstations in academic and

engineering environments. Free software Unix variants, such as GNU, Linux and BSD, are popular in these areas.

Market share statistics for freely available operating systems are usually inaccurate since most free operating systems are not purchased, making usage under-represented. On the other hand, market share statistics based on total downloads of free operating systems are often inflated, as there is no economic disincentive to acquire multiple operating systems so users can download multiple systems, test them, and decide which they like best.

Some Unix variants like HP's HP-UX and IBM's AIX are designed to run only on that vendor's hardware. Others, such as Solaris, can run on multiple types of hardware, including x86 servers and PCs. Apple's Mac OS X, a hybrid kernel-based BSD variant derived from NeXTSTEP, Mach, and FreeBSD, has replaced Apple's earlier (non-Unix) Mac OS.

Unix interoperability was sought by establishing the POSIX standard. The POSIX standard can be applied to any operating system, although it was originally created for various Unix variants.

Mac OS X

Mac OS X is a line of proprietary, graphical operating systems developed, marketed, and sold by Apple Inc., the latest of which is pre-loaded on all currently shipping Macintosh computers. Mac OS X is the successor to the original Mac OS, which had been Apple's primary operating system since 1984. Unlike its predecessor, Mac OS X is a UNIX operating system built on technology that had been developed at NeXT through the second half of the 1980s and up until Apple purchased the company in early 1997.

The operating system was first released in 1999 as Mac OS X Server 1.0, with a desktop-oriented version (Mac OS X v10.0) following in March 2001. Since then, five more distinct "end-user" and "server" editions of Mac OS X have been released, the most recent being Mac OS X v10.5, which was first made available in October 2007. Releases of Mac OS X are named after big cats; Mac OS X v10.5 is usually referred to by Apple and users as "Leopard".

The server edition, Mac OS X Server, is architecturally identical to its desktop counterpart but usually runs on Apple's line of Macintosh server hardware. Mac OS X Server includes workgroup management and administration software tools that provide simplified access to key network services, including a mail transfer agent, a Samba server, an LDAP server, a domain name server, and others.

Real-time operating systems

A real-time operating system (RTOS) is a multitasking operating system intended for applications with fixed deadlines (real-time computing). Such applications include some small embedded systems, automobile engine controllers, industrial robots, spacecraft, industrial control, and some large-scale computing systems.

An early example of a large-scale real-time operating system was Transaction Processing Facility developed by American Airlines and IBM for the Sabre Airline Reservations System.

Embedded systems

Embedded systems use a variety of dedicated operating systems. In some cases, the "operating system" software is directly linked to the application to produce a monolithic special-purpose program. In the simplest embedded systems, there is no distinction between the OS and the application.

Embedded systems that have fixed deadlines use a real-time operating system such as VxWorks, eCos, QNX, and RTLinux.

Some embedded systems use operating systems such as Symbian OS, Palm OS, Windows CE, BSD, and Linux, although such operating systems do not support real-time computing.

Windows CE shares similar APIs to desktop Windows but shares none of desktop Windows' codebase.

Hobby development

Operating system development, or OSDev for short, as a hobby has a large cult-like following. As such, operating systems, such as Linux, have derived from hobby operating system projects. The design and implementation of an operating system requires skill and determination, and the term can cover anything from a basic "Hello World" boot loader to a fully featured kernel. One classical example of this is the Minix Operating System—an OS that was designed by A.S. Tanenbaum as a teaching tool but was heavily used by hobbyists before Linux eclipsed it in popularity.

Other

Older operating systems which are still used in niche markets include OS/2 from IBM; Mac OS, the non-Unix precursor to Apple's Mac OS X; BeOS; XTS-300. Some, most notably AmigaOS and RISC OS, continue to be developed as minority platforms for enthusiast communities and specialist applications. OpenVMS formerly from DEC, is still under active development by Hewlett-Packard.

Research and development of new operating systems continues. GNU Hurd is designed to be backwards compatible with Unix, but with enhanced functionality and a microkernel architecture. Singularity is a project at Microsoft Research to develop an operating system with better memory protection based on the .Net managed code model. Systems development follows the same model used by other Software development, which involves maintainers, version control "trees"[5], forks, "patches", and specifications. From the AT&T-Berkeley lawsuit the new unencumbered systems were based on 4.4BSD which forked as FreeBSD and NetBSD efforts to replace missing code after the Unix wars. Recent forks include DragonFly BSD and Darwin from BSD Unix [6].

Kernel Preemption

In recent years concerns have arisen because of long latencies often associated with some kernel run-times, sometimes on the order of 100ms or more in systems with monolithic kernels. These latencies often produce noticeable slowness in desktop systems, and can prevent operating systems from performing time-sensitive operations such as audio recording and some communications. [7]

Modern operating systems extend the concepts of application preemption to device drivers and kernel code, so that the operating system has preemptive control over internal run-times as well. Under Windows Vista, the introduction of the Windows Display Driver Model (WDDM) accomplishes this for display drivers, and in Linux, the preemptable kernel model introduced in version 2.6 allows all device drivers and some other parts of kernel code to take advantage of preemptive multi-tasking.

Under Windows prior to Windows Vista and Linux prior to version 2.6 all driver execution was co-operative, meaning that if a driver entered an infinite loop it would freeze the system.