Storage Management

Each medium is controlled by a device, such as a disk drive or tape drive, that also has its own unique characteristics. These properties include:

- Access speed
- Capacity
- Data-transfer rate
- Access method (sequential or random)

File(s)	• Collection of related information defined by its creator
	• Represent programs (source and object) and data
	• May be numeric, alphabetic, alphanumeric or binary
	• May be free-form (text files) or formatted (fixed fields)

Storage Management

The operating system implements the abstract concept of a file by managing mass-storage media, such as tapes and disks, and the devices that control them. Also files are normally organized into directories (folders) to make them easier to use. Finally, when multiple users have access to files, it may be desirable to control by whom and in what ways (i.e. read, write, append) file may be accessed.

The operating system is responsible for the following activities in connection with the file management:

- Creating and deleting files
- Creating and deleting directories to organize files
- Supporting primitives for manipulating files and directories
- Mapping files onto secondary storage
- Backing up files on stable (nonvolatile) storage media

Storage Management

Mass-Storage Management

Because of the limited size of main memory, computer systems provides secondary storage to back up main memory. Most modern computer systems use disks as the principal on-line storage medium for both programs and data. Most program are stored on disk until load into memory and then use the disk as both the source and destination of their processing. Hence, the proper management of disk storage is of central importance to a computer system.

The operating system is responsible for the following activities in connection with disk management:

- Free-space management
- Storage allocation
- Disk scheduling

Caching

Caching is an important principle of computer systems. Information is normally kept in some storage system (such as main memory). As it is used, it is copied into a faster storage system – the cache – on a temporary basis. When a particular piece of information is needed, cache is checked first. If found in cache, the information is used directly from cache; if not, the information from the source is used, by first putting a copy in the cache under the assumption that it will be needed again.

Most systems have an instruction cache to hold the instructions expected to be executed next. Without this cache, the CPU would have to wait several cycles while an instruction was fetched from main memory. Also, most systems have one or more high-speed data caches in the memory hierarchy.

Caching

Because of it’s limitation in size, the management of cache becomes an important design problem.

Large Workstations and Small Servers

Storage Performance

Level	1	2	3	4
Name	**Registers**	**Cache**	**Main Memory**	**Disk Storage**
Typical Size	< 1 KB	< 16 MB	< 64 GB	> 100 GB
Implementation Technology	Custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	Magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000,000
Bandwidth (MB/sec)	20,000 – 100,000	5,000 – 10,000	1,000 – 5,000	20 – 150
Managed by	Compiler	Hardware	Operating system	Operating system
Backed by	Cache	Main memory	Disk	CD or tape

- CMOS - semiconductor chip that holds data without requiring external power.
- SRAM - a type of semiconductor memory where the word *static* indicates that, unlike *dynamic* RAM (DRAM), it does not need to be periodically refreshed
- DRAM - a type of random-access memory that stores each bit of data in a separate capacitor within an integrated circuit.

Caching

I/O Systems

One of the purposes of an operating system is to hide the peculiarities of specific hardware devices from the user.

I/O subsystems consists of several components:

- A memory management component that includes buffering, caching, and spooling
- A general device-drive interface
- Driver for specific hardware devices

Only the device driver knows the peculiarities of the specific device to which it is assigned.

Protection and Security

If a computer system has multiple users and allows the concurrent execution of multiple processes, then access to data must be regulated. Mechanisms ensure that files, memory segments, CPU, and other resources can be operated on by only those processes that have gained proper authorization from the operating system.

The timer ensures that no process can gain control of the CPU without eventually relinquishing control.

Protection is any mechanism used in controlling the access of processes or users to the resources defined by a computer system.

- mechanism must specify controls
- provide a means to enforce the controls

Protection and Security

A system can have adequate protection and still be prone to failure and allow inappropriate access. (i.e. stolen user account information) A person’s data can be accessed copied or deleted even though file and memory protection are working.

Computer Security involves defending against internal and external attacks such as:

- Viruses and worms
- Denial of service attacks
- Identity theft
- Theft of services

Some of these function are handled by the operating system while others are handled by policies or additional software.

Protection and Security

Individual User Setting

Most systems uses user names and associated user identifiers. In Windows Vista, which is a log in system, authenticates the user’s login and associates all of the user’s processes and threads.

Multi-user Setting

A system may need to authenticate access among a set of users rather than a single user (i.e. file access rwx). The owner of a file on a UNIX system may be allowed to issue all operations on that file, whereas a selected set of user may only be allowed to read the file. This is done by defining user groups and assigning users to that group.

Protection and Security

UNIX file permissions

```
-rwxr-xr-x 1 nick users 382 Jan 19 11:49 bscoped.pl
drwxr-xr-x 3 nick users 1024 Jan 19 11:19 lib/
-rwxr-xr-x 1 nick users 1874 Jan 19 10:23 socktest.pl
```

Distributed Systems

A distributed system is a collection of physically separate, possibly of different types, computer systems that are networked to provide the user with access to the various resources that the system maintains. Access to shared resources increases computation speed, functionality, data availability, and reliability.

A network is a communication path between to or more systems. Distributed systems depend on networking for their functionality.

Networks vary by the protocols used. TCP/IP is the most common network protocol, although ATM and other protocols are in widespread use.

Distribution Systems

Networks are characterized based on the distance between their nodes.

- Local-Area Network (LAN)   (room, floor, building)
- Wide-Area Network (WAN) (buildings, cities, countries)
- Metropolitan-Area Network (MAN) (building within cities)
- Small-Area Network (SAN) (several feet – in home)

Special-Purpose Systems

Real-Time Embedded Systems

Embedded computers are the most form of computers in existence. These devices are found everywhere, from car engines and manufacturing robots to DVDs and microwave ovens. They tend to have very specific tasks. The systems they run on are usually primitive, and so the operating systems provide limited features. Usually, they have little or no user interface, preferring to spend their time monitoring and managing hardware devices.

Embedded system almost always run real-time operating systems.

Computing Environments

Traditional

- PC connected to network
- Server providing file and printer services
- Portability achieved using laptop
- Home/office internet connection
  - slow

Current Trend

- Web access
- Handheld synchronizing with PC
- PDAs connect to wireless network
- Internet connection relatively inexpensive

Computing Environments

Client-Server Computing

As PCs have become faster, more powerful, and cheaper, designers have shifted away from centralized system architecture. Terminals connected to centralized systems are now being supplanted by PCs. Correspondingly, user-interface functionality once handled directly by centralized systems is increasingly being handled by PCs. As a result, many of today's systems act as server systems to satisfy requests generated by client systems. This form of specialized distributed system is called a client-server system.

Compute-server system provides and interface to which a client can send a request to perform an action, in response

Computing Environments

Client-Server Computing

Compute-server system provides and interface to which a client can send a request to perform an action, in response, the server executes the action and sends back results to the client.

File-server system provides a file-system interface where clients can create, update, read, and delete files. (i.e. web servers that delivers files to clients running a browser)

Peer-to-Peer Computing

Another structure fro a distributed system is the peer-to-peer (P2P) system model, where clients and servers are not distinguished from one another; instead, all nodes within the system are considered peers, and each may act as either a client or a server, depending on whether it is requesting or providing a service. (i.e. Napster, Gnutella)

Computing Environments

Web-Based Computing

The Web has become ubiquitous, leading to more access by a wider variety of devices than was dreamt of a few years ago. PCs are still the most prevalent access device, with workstations, handheld PDAs, and even cell phones also providing access.

Web computing has increased the emphasis on networking. Devices that were not previously networked now include wired or wireless access. Devices that were networked now have faster network connectivity, provided by either improved networking technology, optimized network implementation code or both.

The implementation of Web-based computing has given rise to new categories of devices, such as load balancers, which distribute network connections among a pool of similar servers.

Open-Source Operating Systems

Open-source operating systems are those made available in source-code format rather than as compiled binary code. Linux is the most famous open-source operating system, while Microsoft Windows is a well-known example of the opposite closed-source approach.

Benefits of open-source operating systems

- Community of interested programmers
- Help in debugging
- Analyze code
- Provide support
- Suggest changes