Operating System Concepts

Introduction

INFO 2603
Platform Technologies 1

Week 3: 17-Sept-2018

Outline

- Operating System Software
  - Main Memory Management
  - Processor Management
  - Device Management
  - File Management
  - Network Management
  - User Interface
  - Cooperation Issues
  - Cloud Computing
- Types of Operating Systems

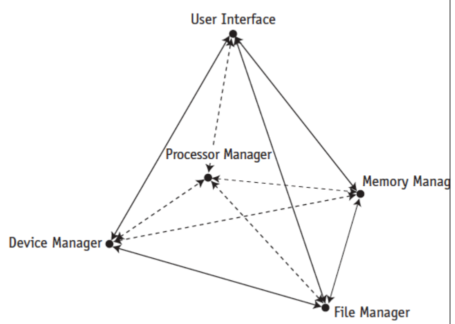
Operating System Software

Four essential managers of every major operating system:

- Memory Manager
- Processor Manager
- Device Manager
- File Manager

(figure 1.1)

This model of a non-networked operating system shows four subsystem managers supporting the User Interface.



Page 5: Understanding Operating Systems, McHoes & Flynn, 2017

Operating System Software

The four managers work together to:

- Monitor the system's resources
- Enforce policies that determine what component get which resources, when and how much
- Allocate the resources when appropriate
- Deallocate the resources when appropriate

Main Memory Management

The Memory Manager is in charge of main memory or RAM: (Random Access Memory). RAM is sometimes called primary storage. It is volatile and is wiped clean when power is turned off or fails.

It checks the validity of requests for memory space and allocates a portion of memory that isn't already in use, if the request is legal.

It uses policies to reallocate memory to make more useable space for jobs if the memory space become fragmented.

It deallocates allotted memory space when a job or process is finished.

A critical responsibility therefore is to protect all of the space in main memory.

Main Memory Management

Another kind of memory that is used when the computer is powered on is ROM (Read-only Memory). ROM data is nonvolatile, and persists when power is turned off.

The ROM chip holds firmware - the programming code that is used to start the computer and perform other necessary tasks.

Firmware describes, in steps, when and how to load each piece of the operating system after power is turned on.

Device Management

The Device Manager is responsible for:

- connecting with every device that's available on the system
- choosing the most efficient way to allocate devices based on scheduling policies
- uniquely identifying each device
- starting/stopping device operations when appropriate
- monitoring progress
- deallocating devices to make the OS available for the next waiting process

Device Management

This is a complicated task because:

- A wide range of devices are available (printers, ports, disk drives etc) and there are hundreds/thousands of manufacturers.
- Some devices can be shared, while some can be used by only one user or only job at a time.

This complexity is managed by the OS through the use of device drivers - software that contains detailed instructions required by an OS to start that device, allocate it to a job, use the device correctly and deallocate when it's appropriate.

File Management

The **File Manager** keeps track of every file in the system (data files, program files, utilities, compilers, applications etc).

Access policies are used by the File Manager to enforce restrictions on who has access to which files, and the range of actions that a user may perform on the files after accessing them.

The File Manager allocates space on secondary storage devices (hard drive, flash drive, archival device etc) based on the technical requirements of that device. e.g. contiguous storage

Network Management

There is a 5th essential manager, the **Network Manager,** that allows authorised users to share resources.

It is responsible for:

- managing network connectivity
- knowing the requirements of available devices, files, memory space, CPU capacity, transmission connections, types of encryption (if necessary).

Networks can range in size and complexity. The OS must be able to manage the available memory, CPU, devices and files available on a network.

Cooperation Issues

Consider the following example of the steps as someone chooses a menu option to open a program:

1. The Device Manager receives the electrical impulse caused by the mouse click, decodes the command by calculating the location of the cursor and sends that information through the UI, which identifies the requested command. It immediately sends the command to the Processor Manager.
2. The Processor Manager sends an acknowledgement message (waiting or loading) to be displayed on the monitor so that the user knows the command has been successfully sent.
3. The Processor Manager determines whether the user request requires a (program) file to be retrieved from secondary storage, or whether it is already in memory.

Cooperation Issues

4. If the program is in secondary storage (disk), the File Manager calculates it exact location on the disk and passes this information to the Device Manager, which retrieves the program and send it to the Memory Manager
5. The Memory Manager finds space (if necessary) for the program in main memory and records its exact location.
6. When the CPU is ready to run it, the program begins execution via the Processor Manager. When the program is finished executing, the Processor Manager relays this information to the other managers.

Cooperation Issues

7. The Processor Manager reassigns the CPU to the next program waiting in line. If the file was modified, the File Manager and Device Manager cooperate to store the results in secondary storage. IF the file was not modified, there's no need to change the stored version of it.
8. The Memory Manager releases the program's space in main memory and gets ready to make it available to the next program that requires memory.
9. The User Interface displays the results and gets ready to take the next command.

Cloud Computing

Cloud computing is the practice of using Internet-connected resources to perform processing, storage or other operations.

It allows the OS to accommodate remote access to the system resources and provides increased security for these transactions.

The OS still maintains responsibility for managing all local resources and coordinating data transfer to and from the cloud.

The OS that manages far-away resources is responsible for allocation and deallocation of all its resources, but on a massive scale now.

Types of Operating Systems

Operating systems fall into several general categories distinguished by their response, and the method used to enter data into the system.

Five main categories:

- Batch systems
- Interactive systems
- Hybrid systems
- Real-time systems
- Networks

Batch Systems

These feature jobs that are entered as a whole, and in sequence:

- only one job can be entered at a time
- once a job begins processing, no other job can start until the resident job finishes

These date from early computers, when each job consisted of stacks of cards or reels of magnetic tape for input. These were entered into the system as a unit, called a **batch**. The user did not interact directly with the computer system.

The efficiency of a batch system is measured in **throughput**: number of jobs completed in a given time (minutes, hours, days).

IBM's OS360, z/OS: popular batch processing systems

Interactive Systems

Interactive Systems allow multiple jobs to begin processing and return the results to users with better response times than batch systems.

Early versions of these OSs allowed each user to interact directly with the computer system via commands entered from a terminal.

The OS used complex algorithms to share processing power (single processor) among the waiting jobs. Huge improvements in responsiveness with turn around time in seconds/minutes.

These are slower than real-time systems.

Hybrid Systems

Hybrid systems are a combination of batch and interactive systems.

They appear to be interactive because individual users can enter multiple jobs or processes into the system and get fast responses.

They also accept and run batch programs in the background when the interactive load is light.

These systems take advantage of the free time between high-demand usage of the system and low demand times.

Real-time Systems

Real-time systems are used in time-critical environments where reliability and deadlines are critical. The time limit need not be ultra-fast but system response time must meet the deadline: significant consequences for not doing so.

Contingencies for graceful failure are needed. System capabilities and data must be preserved in these cases.

Examples: spacecraft, airport traffic control, fly-by-wire aircraft, critical industrial processes, medical systems etc.

Two types of real-time systems:

- Hard real-time systems: risk total system failure if predicted deadline is missed
- Soft real-time systems: suffer performance degradation, but not total system failure, as a consequence of a missed deadline.

Windows CE, OS-9, Symbian and LynxOS are some of the commonly known real-time operating systems.

Network Operating Systems

Network operating systems allow users to perform their tasks using few, if any, local resources e.g. cloud computing.

These manipulate resources that may be located over a wide geographical area. Wireless networking is a standard feature in many computing devices and these systems make use of networked connections.

Originally similar to single-processor OSs.

Amoeba, Plan9 and LOCUS (developed during the 1980s) are some examples of distributed operating systems.

Embedded Systems

Embedded Systems are physically placed inside products that they operate to add specific features and capabilities. Example: automobiles, digital music players, elevators, pacemakers etc.

These are different from general OSs as each one is designed to execute specific programs that are not interchangeable among systems. Embedded OSs are lean and efficient to take advantage of the limited computer resources (slower CPU, smaller memory)

Unneeded features or functions are dropped (small kernel). Only necessary elements are kept.

Windows CE, FreeBSD and Minix 3 are some examples of embedded operating systems. The use of Linux in embedded computer systems is referred to as Embedded Linux.

Case Study: Autonomous Vehicles

U.S. trucks carry more than 10 billion tons of freight each year, according to the American Trucking Association, and 43% of the expenses they incur track back to drivers. It's also an industry currently understaffed by about 50,000.

TuSimple, an autonomous-tech startup, has been touting a breakthrough in the vision of its semi trucks this year — they're now able to "see" up to 1,000 meters ahead on the highway. No other autonomous system is known to have such long-sighted vision, and TuSimple is getting rewarded greatly for the algorithms that enable it. The company has raised more than $80 million from investors including Nvidia. Competitors like Waymo are testing autonomous semis, but so far don't seem to have the depth of vision TuSimple's team says it's unlocked.

Embark is a 2-year-old self-driving truck company whose founders think they can get to market first.

Kodiak Robotics has its sights set on long-haul trucking. Led, in part, by ex-Uber, ex-Google Self-Driving Car Don Burnette, the company has raised $40 million to test AI-enabled trucks and hire enginee

Research these companies and examine the types of OSs and hardware used.

Reading Resources

Understanding Operating Systems: Ann McIver McHoes, Ida M. Flynn. 8th Edition. 2017: Chapter 1 - OS Concepts

Understanding Operating Systems: Ann McIver McHoes, Ida M. Flynn. 8th Edition. 2017: Chapter 7 - Device Management