Makara Ramoabi

20240045

Operating system

1. Scenario Analysis

When running multiple applications on Windows, such as Google Chrome, Microsoft Word, and Spotify, each program operates as a separate process. When you open Chrome, the operating system creates a process and places it in the **New** state. It then moves to the **Ready** state, waiting for CPU allocation. Once the CPU assigns time to it, the process enters the **Running** state. If you switch to Word, Chrome may move back to the **Ready** state while Word becomes **Running**. If a process waits for input, such as loading a webpage or saving a file, it enters the **Waiting (Blocked)** state. After the input/output operation completes, it returns to the **Ready** state. When you close an application, it moves to the **Terminated** state, and its resources are released. These state transitions allow the operating system to efficiently manage multitasking and ensure smooth performance.

2. Concept Research

A Process Control Block (PCB) is a data structure used by the operating system to manage and track processes. It contains important information such as the process ID (PID), process state, CPU registers, memory allocation details, scheduling information, and input/output status. The PCB acts as a record that allows the operating system to pause and resume processes efficiently during context switching. When the CPU switches from one process to another, the current process's state is saved in its PCB, and the next process's information is loaded. Without the PCB, multitasking and proper resource management would not be possible in modern operating systems.

3. Tool Practice – VirtualBox & Ubuntu Reflection

Using VirtualBox to run Ubuntu provided practical insight into process management. After launching Ubuntu, I opened the terminal and used the top command to monitor active processes. I observed various processes in different states, such as running and sleeping. Each process displayed a PID, CPU usage, and memory consumption. When I terminated a process using the kill command, I noticed it disappeared from the list, showing its transition to the terminated state. This activity demonstrated the process lifecycle clearly: creation at startup, scheduling for CPU time, waiting during idle moments, and termination when killed. The exercise helped me understand how operating systems dynamically manage processes.

4. Process Diagram Explanation

The process state diagram includes five main states: **New, Ready, Running, Waiting, and Terminated**. Arrows between these states represent transitions. A process moves from New to Ready when admitted by the OS. From Ready, it transitions to Running when scheduled by the CPU. If it requires input/output operations, it moves from Running to Waiting. Once the operation completes, it returns to Ready. If the process finishes execution or is manually stopped, it moves to the Terminated state. The diagram visually represents how processes continuously change states based on system events, helping to clarify how multitasking works in an operating system.