International Journal of Scientific Research and Engineering Development-–Volume 8 Issue 5, Sep - Oct 2025
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Review on Process Management in Operating System

Nisha Pawar*, Omkar Vidhate**, Aryan Niware***
*(Lecturer at P.E.S Polytechnic, Chh. Sambhaji Nagar(Aurangabad), Maharashtra, India
Email: pawarnisha1797@gmail.com)
** (Diploma in Computer Engineering, P.E.S Polytechnic, Chh. Sambhaji Nagar(Aurangabad), Maharashtra, India
Email: labc9253@gmail.com
*** (Diploma in Computer Engineering, P.E.S Polytechnic, Chh. Sambhaji Nagar(Aurangabad), Maharashtra, India
Email: labc9253@gmail.com)

--************************----------------------------------

Abstract:

Process management is crucial for an operating system, managing process creation, execution, and termination. This paper quickly looks at how processes work, when they're done, talking between them, waiting for things, and changing their mind about what's happening. It emphasizes system calls that serve as bridges between user programs and the kernel. These mechanisms work together to make CPUs use their resources well, respond quickly, and stay reliable in today's computer programs.

Keywords — **Process management, Operating system, System calls, Scheduling, Inter-process communication, Synchronization.**

--************************----------------------------------

I. INTRODUCTION

An OS is the base software that connects computer parts to apps. The system handles the CPU, RAM, I/O devices, and files, offering standard services for efficient and secure program execution by users. Its primary duty involves managing processes. A process is just an active program that includes its instructions, what it's doing now, and which tools it has available. Organised process management enables multitasking and resource sharing, ensuring system responsiveness remains high.

Process management encompasses every stage from creation through execution, storage in memory, scheduling on the CPU for interaction with others, and termination. The OS keeps track of processes via PCBs. It decides which process runs next based on scheduling policies. Context switches simulate parallelism. Modern systems allow for the creation of threads, which are small parts of an operation within a process that use shared resources but can operate separately to enhance efficiency and make better use of multiple processor cores.

II. LITERATURE SURVEY

Standard OS texts like Silberschatz et al. And Tanenbaum explains the basics of process management, such as process stages, timing and exchanges between processes. Research papers show how managing processes well helps computers work faster and more reliably. These studies provide the foundation for the analysis discussed in this paper.

III. COMPONENTS OF PROCESS MANAGEMENT

A. Process Life Cycle and States:

A transformation occurs as an entity transitions from inception to conclusion. New, ready, running, waiting/blocked, terminated are typical states. When creating a new state, the process initiates, and the operating system establishes essential data structures and distributes resources. In the Ready state, the process has everything it needs except the CPU; it waits in the ready queue to be scheduled for execution. The CPU begins running when it executes process instructions. If the task needs to wait until something happens, like when data comes in or resources become available, it moves into the

Waiting or Blocked state. After finishing, the procedure goes into the Terminated state, releasing all resources from the operating system.

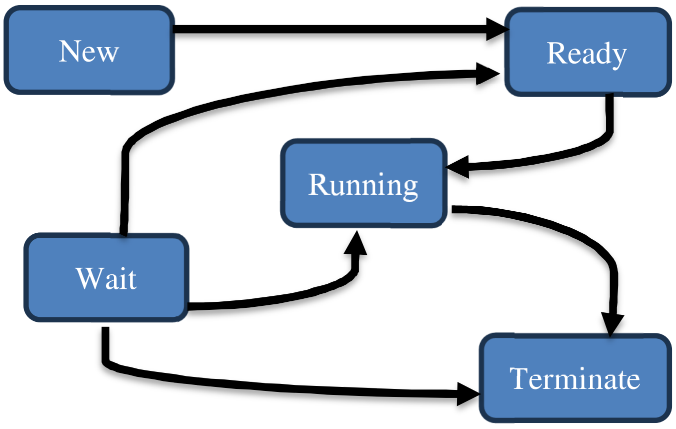

Fig. 1 Process Life Cycle

B. *Process Control Block (PCB):*

The operating system keeps track of processes using the PCB as a data structure. Essential details stored include process ID, current status, CPU registers, program counter, memory usage, file handles, and accounting info. During changes in programs, the computer system stores and then returns the program control block data to start a task properly.

C. *Threads:*

A thread is the smallest part that runs in a program. A number of streams may coexist in a single program, exchanging its software and assets yet operating separately. Threads enhance performance by better managing resources across multiple processors. Parallelism within applications can be achieved without needing separate processes. Operating systems use user-level, kernel-level, or mixed-threading methods; these mix the speed of user-level threads with the security of kernel support.

D. *Scheduling:*

Scheduling decides what task gets executed first by the CPU. Long-term plans decide when tasks go into the system. – Medium-term scheduling suspends or resumes processes. Short-term scheduling chooses which task will run next when all tasks are available. Common algorithms are FCFS, SJF, priority scheduling, and round robin. Modern operating systems frequently employ multilevel feedback queues or mixed strategies to achieve optimal performance in terms of speed, equality, and efficiency.

E. *E. Context Switching:*

involves temporarily storing the current program's data and then restoring it when needed for another task. It allows doing many things at once and works well for tasks and programs that run on computers with just one processor or ones with multiple processors. Context switching in systems is crucial for efficiency; modern kernels enhance it by reducing data storage needs and utilizing CPU features like distinct registers.

IV. CONCLUSIONS

Efficiently managing processes is crucial for the effectiveness and reliability of all systems. The operating system manages the process lifecycle, maintains printed circuit boards, supports threading, uses proper scheduling rules, performs quick switch contexts, ensuring balanced resources, high CPU usage, and system efficiency. Modern multitasking systems rely on these components as their foundation, which continually adapt to meet demands for real-time processing, multiple cores, and virtualized environments.

ACKNOWLEDGMENT

The author wishes to thank the Department of Computer Engineering at P.E.S Polytechnic College for their support. Chh. Sambhaji Nagar in Maharashtra provided continuous encouragement and support throughout the writing of this paper. Special thanks go to Nisha Pawar Mam for her helpful ideas and comments that made this project better.

REFERENCES

1] A. Silberschatz, P. B. Galvin and G. Gagne, *Operating System Concepts*, 10th Edition, Wiley, 2023. https://www.wiley.com/en-in/Operating+System+Concepts%2C+10th+Edition-p-9781119800361

[2] A. S. Tanenbaum and H. Bos, *Modern Operating Systems*, 5th Edition, Pearson, 2023. https://www.pearson.com/store/p/modern-operating-systems/P100000955314

[3] M. Singhal and N. G. Shivaratri, *Advanced Concepts in Operating Systems*, McGraw-Hill, 2011. https://www.mheducation.com/highered/product/advanced-concepts-operating-systems-singhal-shivaratri/M9780071142438.html

[4] S. Kumar and R. Gupta, "Analysis of CPU Scheduling Algorithms in Multiprocessor Systems," *International Journal of Computer Applications*, vol. 182, no. 32, pp. 25–29, 2024. https://www.ijcaonline.org/archives/volume182/number32/31897-2024918375/