Course content

The course covers the fundamental principles and implementation methods of operating systems in managing computer system resources. The course integrates real-world examples to demonstrate these concepts. As an essential foundation for computer science and technology students, this course provides in-depth knowledge of the core theories of modern operating systems, as well as the latest developments in new concepts and technologies.

The following topics are covered in the course:

1. Process Management: Fundamentals of process scheduling, synchronization, and communication in operating systems.
2. Memory Management: Techniques for managing memory allocation, paging, segmentation, and virtual memory.
3. File Systems: File organization, directory structures, and the implementation of file systems in operating systems.
4. I/O Device Management: Managing input/output devices, including buffers, device drivers, and interrupt handling.
5. Deadlock: Analysis and prevention of deadlocks in multi-process systems, and strategies to resolve them.
6. Virtualization: The use of virtual machines and software simulation for creating isolated environments and managing resources.
7. System Design and Development: Building and testing operating system components using development tools, debugging software, and creating experimental setups.

Course objectives

Knowledge

1. Gain a comprehensive understanding of the basic principles and technologies used in operating systems, including process management, memory management, file systems, and I/O management.
2. Understand the latest developments and new technologies in operating system design and implementation.
3. Learn the concepts and methods for analyzing and evaluating operating system performance, with a focus on real-world applications.

Skills

1. Develop the ability to apply operating system principles and related knowledge to compare, integrate, and optimize solutions to complex problems in the field of operating systems.
2. Use operating system fundamentals to identify key challenges in solving complex problems and design solutions for them.
3. Analyze and compare various solutions to operating system issues, deriving effective and efficient solutions.
4. Design and implement experimental setups to research, analyze, and solve complex operating system problems.

Competencies

1. Apply knowledge of operating systems to solve complex engineering problems, leveraging both theoretical understanding and practical tools.
2. Evaluate and optimize operating systems, considering design trade-offs and performance metrics.
3. Choose appropriate virtualization software, debugging tools, and system development environments for research and development tasks in operating systems.
4. Set up and conduct experiments in operating system design, ensuring proper data collection and analysis to support system development and optimization.