

<<面向设计方法的操作系统>>

图书基本信息

书名：<<面向设计方法的操作系统>>

13位ISBN编号：9787506241182

10位ISBN编号：7506241188

出版时间：1999-04

出版时间：世界图书出版公司

作者：C.Crowley

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<面向设计方法的操作系统>>

书籍目录

CONTENTS

Introduction 1

1.1 Where Does an Operating System

Fit in?

1.1.1 System Levels

1.2 What Does an Operating System Do?

2.1 Hardware Resources

2.2 Resource Management

2.3 Virtual Computers

A Virtual Computer

3.1 Virtual Processor

3.2 Virtual Primary Memory

3.3 Virtual Secondary Memory 10

3.4 Virtual I/O

Do We Need an Operating System?

Summary

1.5.1 Terminology

1.5.2 Review Questions

1.5.3 Further Reading

1.6 Problems

2 The Hardware Interface

The CPU

2.2 General-Purpose Register

2.3 Control Registers

2.4 Processor Modes

Instruction Set

Machine Instructions in C

Code

Memory and Addressing

Interrupts

I/O Devices

2.4.1 Disk Controller

Summary

2.5.1 Terminology

2.5.2 Review Questions

2.5.3 Further Reading

2.6 Problems

3 The Operating System Interface

3.1 What Are System Calls?

3.1.1 How to Make a System Call

3.1.2 What Is a System Call Interface?

3.2 An Example System Call Interface

3.2.1 System Call Overview

3.2.2 Hierarchical File Naming

Systems

<<面向设计方法的操作系统>>

- 3.2.3 File and I/O System Calls
- 3.2.4 Open Files
- 3.3
- 3.4
- 3.5
- 3.2.5 Examples of File I/O
- Information and Meta-Information
- Naming Operating System Objects
- 3.5 Devices as Files
- 3.5.1 Unification of the File and Device Concepts
- 3.6 The Process Concept
- 3.6.1 Processes and Programs
- 3.6.2 Process Management System Calls
- 3.7
- 3.6.3 Process Hierarchy
- Communication between Processes
- 3.7.1 Communication-Related System Calls :
- 3.7.2 Example of Interprocess.
- 3.8 UNIX-Style Process Creation
- 3.9 Standard Input and Standard Output Communication
- 3.10 Communicating with Pipes
- 3.10.1 Naming of Pipes and Message Queues
- 3.11 Summary of System Call Interface
- 3.12 Operating System Examples
- 3.12.1 UNIX
- 3.12.2 Mach
- 3.12.3 MS/DOS
- 3.12.4 Windows NT
- 3.12.5 os/2
- 3.12.6 Macintosh OS
- 3.13 The User Interface to an Operating System*
- 3.13.1 Why You Need a Shell
- 3.13.2 The Specification of the Shell
- 3.13.3 Implementing the Shell
- 3.14 Summary
- 3.14.1 Terminology
- 3.14.2 Review Questions
- 3.14.3 Further Reading
- 3.15 Problems
- Design Techniques 1
- 4.1 Operating Systems and Design

<<面向设计方法的操作系统>>

- 4.1.1 The Design Process
- 4.1.2 Relationship to Software Engineering
- 4.1.3 A Design Example
- 4.1.4 Learning Design through Operating Systems
- 4.2 Design Problems
 - 4.2.1 Design Skills
 - 4.2.2 Design Space
 - 4.2.3 Design Levels
- 4.3 Design Techniques
- 4.4 Two-Level Implementation
 - 4.4.1 Overview
 - 4.4.2 Motivation
 - 4.4.3 Operating System Examples
 - 4.4.4 Computer Science Examples
 - 4.4.5 Applicability
 - 4.4.6 Consequences
 - 4.4.7 Implementation Issue and Variations
 - 4.4.8 Related Design Techniques
- 4.5 Interface Design
 - 4.5.1 Overview
 - 4.5.2 Motivator
 - 4.5.3 Applicability
 - 4.5.4 Consequences
 - 4.5.5 Related Design Technique
- 4.6 Connection in Protocols
 - 4.6.1 Overview
 - 4.6.2 Motivation
 - 4.6.3 Operating System Examples
 - 4.6.4 Computer Science Examples
 - 4.6.5 Applicability
 - 4.6.6 Consequences
 - 4.6.7 Implementation Issue and Variations 102
 - 4.6.8 Related Design Technique
- 4.7 Interactive and Programming Interfaces
 - 4.7.1 Overview
 - 4.7.2 Motivation
 - 4.7.3 Operating System Examples
 - 4.7.4 Computer Science Examples
 - 4.7.5 Applicability
 - 4.7.6 Consequences
 - 4.7.7 Implementation Issue -and Variations

<<面向设计方法的操作系统>>

- 4.7.8 Related Design Techniques
- 4.8 Decomposition Patterns
 - 4.8.1 Overview
 - 4.8.2 Motivation
 - 4.8.3 Operating System Examples
 - 4.8.4 Computer Science Examples
 - 4.8.5 Applicability
 - 4.8.6 Consequences
 - 4.8.7 Implementation Issue. and Variations
 - 4.8.8 Related Design Technique
- 4.9 Summary
 - 4.9.1 Terminology
 - 4.9.2 Review Question
- 4.10 Problems
- Implementing Processes
- 5.1 The System Call Interface
- 5.2 Implementation of a Simple Operating System 119
 - 5.2.1 Guide to the Code
 - 5.2.2 The Architecture
 - 5.2.3 System Constants
 - 5.2.4 Global Data
- 5.3 Implementation of Processes
 - 5.3.1 Process Creation
 - 5.3.2 Process States
 - 5.3.3 Process Dispatching
 - 5.3.4 The System Stack
 - 5.3.5 Timer Interrupts
- 5.4 System Initialization
 - 5.4.1 The Initial Process
- 5.5 Process Switching
 - 5.5.1 Switching between Processes
 - 5.5.2 Flow of Control
- System Call Interrupt Handling
 - 5.6.1 Copying Messages between Address Spaces
- 5.7 Program Error Interrupts
- 5.8 Disk Driver Subsystem
 - 5.8.1 Communicating with the Disk
- 5.9 Controller
 - Implementation of Waiting
 - 5.9.1 Waiting for Messages
 - 5.9.2 Waiting inside a System Call
 - 5.9.3 Suspending System Calls
- Flow of Control through the Operating System
- 5.10 System

<<面向设计方法的操作系统>>

- 5.11 Signaling in an Operating System
- 5.12 Interrupts in the Operating System
- 5.13 Operating Systems as Event and Table Managers
- 5.14 Process Implementation
 - 5.14.1 The Process Table and Process
- 5.15 Descriptors
- 5.16 Examples of Process Implementation
 - Monoprogramming*
 - 5.16.1 Batch Systems
 - 5.16.2 Multiprogramming and I/O Overlap
 - 5.16.3 Personal Computer System
- 5.17 Summary
 - 5.17.1 Terminology
 - 5.17.2 Review Questions
 - 5.17.3 Further Reading
- Problems
- Parallel Systems
 - 6.1 Parallel Hardware
 - 6.2 An Operating System for a Two-Processor System
 - 6.2.1 Using Two Separate Operating Systems
 - 6.2.2 Sharing the Operating System
 - 6.3 Race Conditions with a Shared Process Table
 - 6.4 Atomic Actions
 - 6.4.1 Hardware Implementation of Atomic Actions
 - 6.5 A Multiprocessor Operating System
 - 6.5.1 The Current Process Variable
 - 6.5.2 Dispatching With a Shared Process Table
 - 6.5.3 Busy Waiting
 - 6.5.4 Handling the Queues
 - 6.5.5 Grouping of Shared Variables
 - 6.5.6 A General Solution
 - 6.5.7 Using Two Process Tables
- Examples of Multiprocessor Operating Systems
- 6.6 Systems
- 6.7. Threads
 - 6.7.1 The Thread Concept
 - 6.7.2 Thread System Calls
 - 6.7.3 Advantages of Threads
 - 6.7.4 Uses of Threads

<<面向设计方法的操作系统>>

- 6.7.5 Thread Implementation*
- 6.7.6 Splitting the Process Concept
- 6.7.7 Lightweight Processes and Use Threads
- 6.7.8 Examples of Threads
- 6.8 Kernel-mode Processes*
- 6.8.1 Data Structures for Kernel-Mode Processes
- 6.8.2 Process Creation with Kernel-Mode Processes
- 6.8.3 Interrupt Handler for Kernel-Mode Processes
- 6.8.4 Switching Processes for Kernel-Mode Processes
- 6.8.5 How the System Slack is Used
- 6.8.6 Waiting with Kernel-Mode Processes
- 6.8.7 Dispatching with Kernel-Mode Processes
- 6.8.8 Kernel-Mode only Processes,
- 6.8.9 Trade-Offs of Kernel-Mode Processes
- 6.8.10 Examples of Kernel-Mode Processes
- 6.9 Implementation of Mutual Exclusion
- 6.9.1 First Solution: Disabling Interrupts
- 6.9.2 Second Solution: Using ExchangeWord
- 6.9.3 Third Solution: Software Solutions
- 6.9.4 When to Use Each Solution
- 6.9.5 Examples of Implementing Mutual Exclusion
- Varieties of Computer Model
- 6.10.1 Multiprogramming
- 6.10.2 Multiprocessing
- Summary
- 6.11.1 Terminology
- 6.11.2 Review Questions
- 6.11.3 Further Reading
- Problems
- Interprocess Communication Patterns
- 7.1 Using Interprocess Communication
- 7.2 Patterns of Interprocess Communication

<<面向设计方法的操作系统>>

- 7.2.1 Competing and Cooperating
 - 7.3 Problems When Processes Compete
 - 7.4 Race Conditions and Atomic Actions
 - 7.5 New Message-Passing System Calls
 - 7.6 IPC Pattern: Mutual Exclusion
 - 7.6.1 N Process Mutual Exclusion
 - 7.6.2 Voluntary Cooperation in Mutual Exclusion
 - 7.7 IPC Pattern: Signaling
 - 7.8 IPC Pattern: Rendezvous
 - 7.8.1 Many Process Rendezvous
 - 7.9 IPC Pattern: Producer-Consumer
 - 7.9.1 The Basic Producer-Consumer Pattern
 - 7.9.2 Limiting the Number of Buffers Used
 - 7.9.3 Multiple Producer and Consumers
 - 7.10 IPC Pattern: Client-Server
 - 7.11 IPC Pattern: Multiple Servers and Clients*
 - 7.12 IPC Pattern: Database Access and Update
 - 7.12.1 Scheduling
 - 7.12.2 Priority
 - 7.12.3 Scheduling Queues
 - 7.13 Review of Interprocess Communication Patterns
 - 7.13.1 Mutual Exclusion
 - 7.13.2 Signaling
 - 7.13.3 Rendezvous
 - 7.13.4 Producer-Consumer
 - 7.13.5 Client-Server
 - 7.13.6 Multiple Servers and Clients
 - 7.13.7 Database Access and Update
 - 7.14 A Physical Analogy
 - 7.15 Failure of Processes
 - 7.15.1 Recovery from Failure
 - 7.16 Summary
 - 7.16.1 Terminology
 - 7.16.2 Review Questions
 - 7.16.3 Further Reading
 - 7.17 Problems
- Processes
- 8.1 Everyday Scheduling
 - 8.1.1 First-Come, First-Served Scheduling

<<面向设计方法的操作系统>>

- Shortest-Job-First Scheduling
- Highest-Response-Ratio-Next Scheduling
- Priority Scheduling
- Deadline Scheduling
- 8.1.6 Round-Robin Scheduling
- 8.1.7 Summary
- 8.2 Preemptive Scheduling Methods
- 8.2.1 Scheduling Overview
- 8.2.2 Round-Robin Scheduling
- 8.2.3 Heavily Loaded Systems
- 8.2.4 Two Queues
- 8.2.5 Multiple Queues
- 8.3 Policy versus Mechanism in Scheduling
- 8.4 A Scheduling Example
- 8.5 Scheduling in Real Operating Systems
- 8.5.1 Scheduling in UNIX SVR4
- 8.5.2 Scheduling in Solaris
- 8.5.3 Scheduling in OS/2.0
- 8.5.4 Scheduling in Windows NT 3.51
- 8.5.5 Scheduling in Other Operating Systems
- 8.6 Deadlock
- 8.7 Why Deadlock Is a Problem
- 8.8 Conditions for Deadlock to Occur
- 8.9 How to Deal with Deadlock
- 8.9.1 Deadlock Prevention
- 8.9.2 Deadlock Avoidance
- 8.9.3 Deadlock Recovery
- 8.10 A Sequence of Approaches to the Deadlock Problem
- 8.11 Two-Phase Locking
- Starvation
- 8.12 Message Passing Variations
- 8.13.1 Using PIDs as Message Addresses
- 8.13.2 Message Passing with Nonblocking Receives
- 8.13.3 Message Passing with Blocking Sends
- 8.13.4 Remote Procedure Call,
- 8.14 Synchronization
- 8.14.1 Definition of Synchronization
- 8.14.2 Review of Synchronization
- 8.15 Separating Data Transfer and

<<面向设计方法的操作系统>>

8.16 Synchronization

Semaphores

8.16.1 Specification of Semaphore

Operations

8.16.2 Implementation of Semaphore

8.16.3 An Analogy

8.16.4 Mutual Exclusion with

Semaphores

8.16.5 Rendezvous with Semaphores

8.16.6 Producer-Consumer (one buffer) with

Semaphores

8.16.7 Counting Semaphores

8.16.8 Producer-Consumer (N buffers) with

Semaphores

8.16.9 Semaphores and Message.

8.17 Implementing Semaphores*

8.17.1 System Constants

Using Semaphores in the Simple Operating

System

8.18 Programming-Language-Based

8.19 Synchronization Primitives

8.19.1 Monitors

8.19.2 Synchronization Primitives in

Ada 95

8.20 Message Passing Design Issues

8.20.1 Copying Messages

8.20.2 Longer Messages

8.21 IPC in Mach 337

8.21.1 Tasks and Threads

8.21.2 Ports and Messages

8.21.3 Objects

8.22 IPC and Synchronization Example.

8.22.1 Signals 338

8.22.2 SVR4 UNIX

8.22.3 Windows NT

8.22.4 OS/2

8.22.5 Solaris

8.23 Summary

8.23.1 Terminology

8.23.2 Review Questions

8.23.3 Further Reading

8.24 Problems

Design Techniques 11

Indirection

Overview

9.1 Motivation

9.1.1 Operating System Examples

<<面向设计方法的操作系统>>

- 9.1.2 Computer Science Examples
- 9.1.3 Discussion
- 9.1.4 Applicability
- Consequences
- 9.2 Using State Machines
- 9.2.1 Overview
- 9.2.2 Operating System Example .
- 9.2.3 Computer Science Example
- 9.2.4 Applicability
- 9.2.5 Consequences
- 9.2.6 Implementation Issue, and Variations
- 9.3 Win Big. Then Give Some Back
- 9.3.1 Overview
- 9.3.2 Motivation
- 9.3.3 Operating System Examples
- 9.3.4 Computer Science Examples
- 9.3.5 Applicability
- 9.4 Consequences
- Separation of Concepts
- 9.4.1 Overview
- 9.4.2 Motivation
- 9.4.3 Operating System Examples
- 9.4.4 Computer Science Examples
- 9.4.5 Applicability
- 9.4.6 Consequences
- 9.4.7 Implementation Issue and Variations
- 9.4.8 Related Design Techniques . Variations
- 9.5 Reducing a Problem to a Special Case
- 9.5.1 Overview
- 9.5.2 Motivation
- 9.5.3 Operating System Examples
- 9.5.4 Computer Science Examples
- 9.5.5 Applicability
- 9.5.6 Consequences
- 9.5.7 Implementation Issue and Variations
- 9.6 Reentrant Programs
- 9.6.1 Overview
- 9.6.2 Motivation
- 9.6.3 Operating System Examples
- 9.6.4 Computer Science Examples
- 9.6.5 Applicability
- 9.6.6 Consequences
- 9.6.7 Implementation Issue and Variations

<<面向设计方法的操作系统>>

- 9.6.8 Related Design Techniques
- 9.7 Using Models for Inspiration
 - 9.7.1 Overview
 - 9.7.2 Motivation
 - 9.7.3 Operating System Examples
 - 9.7.4 Computer Science Examples
 - 9.7.5 Applicability
 - 9.7.6 Consequences
- 9.8 Adding a New Facility to a System
 - 9.8.1 Overview
 - 9.8.2 Motivation
 - 9.8.3 Operating System Examples
 - 9.8.4 Computer Science Examples
 - 9.8.5 Applicability
 - 9.8.6 Consequences
 - 9.8.7 Related Design Technique.
- 9.9 Summary
 - 9.9.1 Terminology
 - 9.9.2 Review Question
- 9.10 Problems
- Memory Management
 - 10.1 Levels of Memory Management
 - 10.2 Linking and Loading a Process
 - 10.2.1 Creating a Load Module
 - 10.2.2 Loading a Load Module
 - 10.2.3 Allocating Memory in a Running Process
 - 10.3 Variations in Program Loading
 - 10.3.1 Load Time Dynamic Linking
 - 10.3.2 Run Time Dynamic Linking
 - 10.4 Why Use Dynamic Memory Allocation?
 - 10.5 The Memory Management Design Problem*
 - 10.6 Solutions to the Memory Management Design Problem*
 - 10.6.1 Static Division into a Fixed Number of Blocks
 - 10.6.2 Buddy Systems
 - 10.6.3 Powers-of-two Allocation
 - 10.7 Dynamic Memory Allocation*
 - 10.8 Keeping Track of the Blocks*
 - 10.8.1 The List Method
 - 10.8.2 Keeping Allocated Blocks on the Block List
 - 10.8.3 Where Is the Block List Kept?
 - 10.8.4 Using Block Headers as Free List

<<面向设计方法的操作系统>>

Nodes

10.8.5 The Bitmap Method

10.8.6 Comparing Free List Methods

10.9 Which Free Block to Allocate?*

10.10 Examples of Dynamic Memory Allocation

10.11 Logical and Physical

Memory

10.12 Allocating Memory to

Processes

10.12.1 Static Memory Management

10.12.2 Handling Variable-Sized

Processes

10.13 Multiprogramming Issues

10.14 Memory Protection

10.15 Memory Management System Calls

10.15.1 Static Allocation of Memory to

Processes

10.15.2 Dynamic Allocation of Memory to

Processes

10.15.3 What about New and Malloc?

10.15.4 Freeing Memory at Each

Level

10.15.5 A Different Memory Management System Call

10.16 Example Code for Memory

Allocation*

10.17 Summary

10.17.1 Terminology

10.17.2 Review Question

10.17.3 Further Reading

10.18 Problems

Virtual Memory

11.1 Fragmentation and Compaction*

11.2 Dealing with Fragmentation

11.2.1 Separate Code and Data Space

11.2.2 Segments

11.2.3 Noncontiguous Logical Address

Spaces

11.2.4 Page Tables in Hardware

Registers

11.2.5 Page Tables in Memory

11.2.6 Using a Page Table Cache

11.2.7 Analysis Models of Paging with Caching

11.2.8 Memory Allocation with Paging

11.2.9 Terminology: Page and Page

<<面向设计方法的操作系统>>

Frame

11.2.10 Page Tables

11.2.11 Paging Summary

11.3 Memory Allocation Code with Page.

11.4 Sharing the Processor and Sharing

Memory*

11.5 Swapping*

11.5.1 Efficient Resource Use and Use

Needs

11.6 Overlays*

11.6.1 Overlays in PCs

11.7 Implementing Virtual Memory

11.7.1 Hardware Required to Support Virtual

Memory

11.7.2 Software Required to Support Virtual

Memory

11.8 What is the Cost of Virtual Memory'?

11.8.1 Paging More Than One Process

11.8.2 Locality

11.9 Virtual Memory Management

11.10 Daemons and Events

11.11 File Mapping

11.11.1 The System Call Interface

11.11.2 An Example of Using File

Mapping

11.11.3 Advantages of File Mapping

11.11.4 Memory and File Mapping on the

IBM 801

11.11.5 File Mapping Example.

11.12 Summary

11.12.1 Terminology

11.12.2 Review Questions

11.12.3 Further Reading

11.13 Problems

12 Virtual Memory Systems

12.1 Page Replacement

12.2 Global Page Replacement Algorithms

12.2.1 Measuring the Performance of a Page

12.2.2 Replacement Algorithm

12.2.3 Optimal Page Replacement

12.2.4 Theories of Program Paging

12.2.5 Behavior

Random Page Replacement

First-In, First-Out FIFO Page

Replacement

Least Recently Used Page

Replacement

<<面向设计方法的操作系统>>

- 12.2.7 Approximations of LRU
- 12.2.8 Clock Algorithms
- 12.3 Page Replacement Examples
- 12.4 Local Page Replacement Algorithm.
 - 12.4.1 What Is a Working Set?
 - 12.4.2 Program Phases
 - 12.4.3 Variable Resident Set Size.
 - 12.4.4 The Working Set Paging Algorithm
 - 12.4.5 Approximating the working Set
 - 12.4.6 WSClock Paging Algorithm
- 12.5 Evaluating Paging Algorithms*
 - 12.5.1 Methodology for Paging Simulation
 - 12.5.2 Some Page Simulation Results
- 12.6 Thrashing and Load Control
 - 12.6.1 How Thrashing Occurs
 - 12.6.2 Load control
 - 12.6.3 Swapping
 - 12.6.4 Scheduling and Swapping
 - 12.6.5 Load Control and Paging Algorithms
 - 12.6.6 Predictive Load Control
 - 12.6.7 Preloading of Pages
- 12.7 Dealing with Large Page Tables
 - 12.7.1 What Is the Problem?
 - 12.7.2 Two-Level Paging
 - 12.7.3 Benefits of Two-Level Paging
 - 12.7.4 Problems with Two-Level Paging
 - 12.7.5 Software Page Table Lookup.
- 12.8 Recursive Address Spaces*
- 12.9 Paging the Operating System Address Space
 - 12.9.1 Locking Pages in Memory
- 12.10 Page Size*
 - 12.10.1 Reasons for a Large Page Size
 - 12.10.2 Reasons for a Small Page Size
 - 12.10.3 Clustering Pages
- 12.11 Segmentation
 - 12.11.1 What Is a Segment?
 - 12.11.2 Virtual Memory with Segmentation
 - 12.11.3 Segmentation with Paging
 - 12.11.4 History of Segmentation
 - 12.11.5 Segment Terminology
- 12.12 Sharing Memory

<<面向设计方法的操作系统>>

- 12.12.1 Reasons for Sharing Memory
- 12.12.2 Shared Memory System Calls
- 12.13 Examples of Virtual Memory Systems
 - 12.13.1 SwapArea
 - 12.13.2 Page Initialization
 - 12.13.3 PageSharing
 - 12.13.4 Double-Handed Clock Algorithm
 - 12.13.5 Standby Page Lists
 - 12.13.6 Clustering Pages
 - 12.13.7 FileMapping
 - 12.13.8 Portable Virtual Memory Systems
 - 12.13.9 Sparse Address Space
 - 12.13.10 OS/2 Version 2.0
 - 12.13.11 WindowsNT
 - 12.13.12 Mach and OSF/1
 - 12.13.13 System V Release 4
 - 12.13.14 Other Systems
- 12.14 Very Large Address Spaces
- 12.15 Summary
 - 12.15.1 Terminology
 - 12.15.2 Review Questions
 - 12.15.3 Further Reading
- 12.16 Problems
- 13 Design Techniques III
 - 13.1 Multiplexing
 - 13.1.1 Overview
 - 13.1.2 Motivation
 - 13.1.4 Operating System Examples
 - 13.1.4 Computer Science Examples
 - 13.1.4 Applicability
 - 13.1.4 Consequences
 - 13.2 Late Binding
 - 13.2.1 Overview
 - 13.2.2 Motivation
 - 13.2.3 Operating System Examples
 - 13.2.4 Computer Science Examples
 - 13.2.5 Applicability
 - 13.2.6 Consequences
 - 13.2.7 Implementation Issue. and Variations
 - 13.2.8 Related Design Technique
 - 13.3 Static Versus Dynamic
 - 13.3.1 Overview
 - 13.3.2 Motivation

<<面向设计方法的操作系统>>

- 13.3.3 Operating System Examples
- 13.3.4 Computer Science Examples
- 13.3.5 Applicability
- 13.3.6 Consequences
- 13.3.7 Implementation Issue and Variations
- 13.3.8 Related Design Technique
- Space-Time Tradeoffs
- 13.4
- 13.4.1 Overview
- 13.4.2 Motivation
- 13.4.3 Computer Science Example
- 13.4.4 Applicability
- 13.4.5 Consequences
- 13.4.6 Implementation Issue and Variations
- 13.4.7 Related Design Technique
- 13.5 Simple Analytic Models
- 13.5.1 Overview
- 13.5.2 Motivation
- 13.5.3 Operating System Example.
- 13.5.4 Applicability
- 13.5.5 Consequences
- 13.5.6 Implementation Issue and Variations
- 13.6 Summary
- 13.6.1 Terminology
- 13.6.2 Review Question.
- Problems
- I/O Devices
- 14.1 Devices and Controllers
- 14.1.1 Device Controllers
- 14.2 Terminal Devices*
- 14.2.1 Basic Terminals
- 14.2.2 Display Commands
- 14.2.3 Example Display Command.
- 14.2.4 Keyboard Events
- 14.2.5 Terminal Capability Database.
- 14.2.6 Virtual Terminals
- 14.2.7 Terminal Interfaces
- 14.2.8 Mouse Devices
- 14.2.9 Event Streams
- 14.2.10 Varieties of Two-Processing
- 14.2.11 Graphics Terminals
- 14.2.12 Color and Color Maps
- 14.2.13 Command Oriented Graphic

<<面向设计方法的操作系统>>

- 14.2.14 X Terminals
- 14.2.15 Terminal Emulators
- 14.2.16 Virtual Terminals and Terminal Emulators
- 14.2.17 PPP; a Network Emulator
- 14.2.18 Modems
- 14.3 Communication Devices*
- 14.3.1 Serial Ports
- 14.3.2 Parallel Ports
- 14.3.3 Ethernet Devices
- 14.3.4 Other Network Devices
- 14.4 Disk Devices
- 14.4.1 Timing of a Disk Access
- 14.4.2 Floppy Disks
- 14.4.3 RAID Devices
- 14.5 Disk Controllers
- 14.6 SCSI Interfaces*
- 14.7 Tape Devices*
- 14.8 CD Devices*
- 14.9 Summary
- 14.9.1 Terminology
- 14.9.2 Review Questions
- 14.9.3 Further Reading
- 14.10 Problems 613
- 15 I/O Systems
- 15.1 I/O System Software
- 15.1.1 Device Drivers
- 15.1.2 Device Driver Interfaces
- 15.1.3 The Two Categories of Device Drivers
- 15.1.4 The Block Device Interface
- 15.1.5 The Character Device Interface
- 15.1.6 Disk Device Driver Access Strategies
- 15.2 Handling Disk Requests
- 15.2.1 Efficiently
- 15.2.2 Double Buffering -- An Aside
- 15.2.3 A Disk Scheduling Example
- 15.2.4 Sector Scheduling within Cylinder Scheduling
- 15.2.5 Combined Sector and Cylinder Scheduling
- 15.2.6 Real-Life Disk Head Scheduling
- 15.3 Modeling of Disks*
- 15.3.1 A Disk Scheduling Anomaly
- 15.3.2 Cylinder Correlations
- 15.3.3 A More Accurate Disk Model
- 15.4 Device numbers

<<面向设计方法的操作系统>>

- 15.5 Unification of Files and I/O Devices
- 15.6 Generalized Disk Device Drivers
 - 15.6.1 Partitioning Large Disks
 - 15.6.2 Combining Disks into a Large Logical Disk
 - 15.6.3 RAMDisks
 - 15.6.4 Memory as a Device
 - 15.6.5 Pseudo-ttys
- 15.7 DiskCaching
- 15.8 Two-Level Structure of Device Drivers
- 15.9 SCSI Device Drivers
- 15.10 Examples of I/O Systems
- 15.11 Summary
 - 15.11.1 Terminology
 - 15.11.2 Review Questions
 - 15.11.3 Further Reading
- 15.12 Problems
- 19.8.2 System Authentication
- 19.8.3 Other Methods of Authentication
- 19.8.4 Password Variants
- 19.8.5 Identifying Objects
- 19.8.6 Identifying a Person
- Mechanisms for Protecting Hardware
- 19.9 Resources
 - 19.9.1 Processor Modes
 - 19.9.2 Protecting Hardware Resource Representation of Protection
- 19.10 Information
 - 19.10.1 Object Types
 - 19.10.2 Operations On Objects
 - 19.10.3 The Protection Database
 - 19.10.4 Access Control Lists
 - 19.10.5 Capability Lists
 - 19.10.6 Modifying the Protection Database
 - 19.10.7 Protection Domains
 - 19.10.8 Protection in Distributed Systems
 - 19.10.9 Caching Protection Data
 - 19.10.10 Operations on Protection Objects
- 19.11 Mechanisms for Software Protection
 - 19.11.1 File Protection Example
 - 19.11.2 Implementation of Protection
 - 19.11.3 Protection Mechanisms and Security Policies
 - 19.11.4 Variations in File Security
- 19.12 Examples of Protection Attacks

<<面向设计方法的操作系统>>

- 19.12.1 Browsing for Information
- 19.12.2 Wiretapping of Communication Lines
- 19.12.3 Trial and Error
- 19.12.4 Password Guessing
- 19.12.5 Searching Trash
- 19.12.6 TrapDoors
- 19.12.7 Running Other People's Programs
- Government Security Level
- 19.14 Protection Examples
 - 19.14.1 Protection In Windows NT
 - 19.14.2 Protection In OSF/1
 - 19.14.3 Protection In UNIX
- 19.15 External Security
 - 19.15.1 Physical Security
 - 19.15.2 Operational Security
 - 19.15.3 NonTechnical Security Threats
- The Use of Cryptography in Computer Security
- 19.16 Security
 - 19.16.1 What Is Cryptography?
 - 19.16.2 Some Basic Definition
 - 19.16.3 Public and Private Key Cryptosystems
 - 19.16.4 Using Cryptography for Privacy
 - 19.16.5 Using Cryptography for Authentication
 - 19.16.6 Authenticating Public Key
- 19.17 Summary
 - 19.17.1 Terminology
 - 19.17.2 Review Questions
 - 19.17.3 Further Reading
- 19.18 Problems
- 20 The Client-Server Model
 - 20.1 Three Modes of Communication
 - 20.2 System Processes
 - 20.2.1 Overview
 - 20.2.2 The Initial Process
 - 20.2.3 System Constants
 - 20.2.4 Initialization
 - 20.2.5 Interrupt Handling
 - 20.2.6 Handling System Calls
 - 20.2.7 The System Call Handling Code
 - 20.2.8 User Knowledge of Message Queues
 - Identifiers
 - 20.2.9 Protection of Resources
 - 20.2.10 Disk Interrupt Handler

<<面向设计方法的操作系统>>

20.2.11 Disk I/O System Process
20.2.12 Server Data Structures
20.3 Micro-Kernel Operating Systems
20.3.1 Tradeoffs of the Client Server
Model
20.3.2 Object-Oriented Operating
Systems
20.4 The Developments toward a Distributed
System
20.4.1 Networked Operating Systems
20.4.2 Distributed Operating Systems
20.5 Summary
20.5.1 Terminology
20.5.2 Review Questions
20.5.3 Further Reading
20.6 Problems
Glossary G-1
References R-1
Index 1-1

<<面向设计方法的操作系统>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>