

# 《软件工程》

## 图书基本信息

# 《软件工程》

## 内容概要

软件工程实践者的研究方法（英文第4版），ISBN：9787111067115，作者：（美）普莱斯曼著

## 作者简介

Roger S. Pressman is an internationally recognized consultant and author in software engineering. He received a B.S.E. (cum laude) from the University of Connecticut, an M.S. from the University of Bridgeport and a Ph.D. in engineering from the University of Connecticut, and has over 25 years of industry experience, holding both technical and management positions with responsibility for the development of software for engineered products and systems.

As an industry practitioner and manager, Dr. Pressman worked on the development of CAD/CAM systems for advanced engineering and manufacturing in aerospace applications. He has also held positions with responsibility for scientific and systems programming.

In addition to his industry experience, Dr. Pressman was Bullard Associate Professor of Computer Engineering at the University of Bridgeport and Director of the University's Computer-Aided Design and Manufacturing Center.

Dr. Pressman is President of R.S. Pressman & Associates, Inc., a consulting firm specializing in software engineering methods and training. He serves as principal consultant, specializing in helping companies establish effective software engineering practices. He developed the RSP&A software engineering assessment method, a unique blend of quantitative and qualitative analysis that helps clients assess their current state of software engineering practice.

In addition to consulting services rendered to many Fortune 500 clients, R-S. Pressman & Associates, Inc. markets a wide variety of software engineering training products and process improvement services. The company has developed a state-of-the-art video curriculum, Essential Software Engineering, which is among the industry's most comprehensive treatments of the subject. Another product, Process Advisor, is a self-directed system for software process improvement.

Dr. Pressman is author of many technical papers, is a regular contributor to industry periodicals, and is author of six books. In addition to Software Engineering: A Practitioner's Approach, he has written Making Software Engineering Happen (Prentice Hall), the first book to address the critical management problems associated with software engineering process improvement, Software Shock (Dorset House), a treatment of software and its impact on business and society, and A Manager's Guide to Software Engineering (McGraw-Hill), a book that uses a unique Q&A format to present management guidelines for instituting and understanding the technology. Dr. Pressman is on the editorial boards of American Programmer and IEEE Software, and is editor of the "Manager" column in IEEE Software. He is a member of the ACM, IEEE, and Tau Beta Pi, Phi Kappa Phi, Eta Kappa Nu, and Pi Tau Sigma.

## 书籍目录

- CONTENTS AT A GLANCE
- PREFACE
- PART ONE THE PRODUCT AND THE PROCESS
- CHAPTER 1 THE PRODUCT
- CHAPTER 2 THE PROCESS
- PART TWO MANAGING SOFTWARE PROJECTS
- CHAPTER 3 PROJECT MANAGEMENT CONCEPTS
- CHAPTER 4 SOFTWARE PROCESS AND PROJECT METRICS
- CHAPTER 5 SOFTWARE PROJECT PLANNING
- CHAPTER 6 RISK MANAGEMENT
- CHAPTER 7 PROJECT SCHEDULING AND TRACKING
- CHAPTER 8 SOFTWARE QUALITY ASSURANCE
- CHAPTER 9 SOFTWARE CONFIGURATION MANAGEMENT
- PART THREE CONVENTIONAL METHODS FOR SOFTWARE ENGINEERING
- CHAPTER 10 SYSTEM ENGINEERING
- CHAPTER 11 ANALYSIS CONCEPTS AND PRINCIPLES
- CHAPTER 12 ANALYSIS MODELING
- CHAPTER 13 DESIGN CONCEPTS AND PRINCIPLES
- CHAPTER 14 DESIGN METHODS
- CHAPTER 15 DESIGN FOR REAL-TIME SYSTEMS
- CHAPTER 16 SOFTWARE TESTING TECHNIQUES
- CHAPTER 17 SOFTWARE TESTING STRATEGIES
- CHAPTER 18 TECHNICAL METRICS FOR SOFTWARE
- PART FOUR OBJECT-ORIENTED SOFTWARE ENGINEERING
- CHAPTER 19 OBJECT-ORIENTED CONCEPTS AND PRINCIPLES
- CHAPTER 20 OBJECT-ORIENTED ANALYSIS
- CHAPTER 21 OBJECT-ORIENTED DESIGN
- CHAPTER 22 OBJECT-ORIENTED TESTING
- CHAPTER 23 TECHNICAL METRICS FOR OBJECT-ORIENTED SYSTEMS
- PART FIVE ADVANCED TOPICS IN SOFTWARE ENGINEERING
- CHAPTER 24 FORMAL METHODS
- CHAPTER 25 CLEANROOM SOFTWARE ENGINEERING
- CHAPTER 26 SOFTWARE REUSE
- CHAPTER 27 REENGINEERING
- CHAPTER 28 CLIENT/SERVER SOFTWARE ENGINEERING
- CHAPTER 29 COMPUTER-AIDED SOFTWARE ENGINEERING
- CHAPTER 30 THE ROAD AHEAD
- TABLE OF CONTENTS
- PREFACE
- PART ONE THE PRODUCT AND THE PROCESS
- CHAPTER 1 THE PRODUCT
- 1.1 THE EVOLVING ROLE OF SOFTWARE
- 1.1.1 An Industry Perspective
- 1.1.2 An Aging Software Plant
- 1.1.3 Software Competitiveness
- 1.2 SOFTWARE
- 1.2.1 Software Characteristics

- 1.2.2 Software Components
- 1.2.3 Software Applications
- 1.3 SOFTWARE: A CRISIS ON THE HORIZON
- 1.4 SOFTWARE MYTHS
- 1.5 SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND INFORMATION SOURCES
- CHAPTER 2 THE PROCESS
- 2.1 SOFTWARE ENGINEERING-A IAYERED TECHNOLOGY
- 2.1.1 Process, Methods, and Tools
- 2.1.2 A Generic View of Software Engineering
- 2.2 THE SOFTWARE PROCESS
- 2.3 SOFTWARE PROCESS MODELS
- 2.4 THE LINEAR SEQUENTIAL MODEL
- 2.5 THE PROTOTYPING MODEL
- 2.6 THERADMODEL
- 2.7 EVOLUTIONARY SOFTWARE PROCESS MODELS
- 2.7.1 The Incremental Model
- 2.7.2 The Spiral Model
- 2.7.3 The Component Assembly Model
- 2.7.4 The Concurrent Development Model
- 2.8 THE FORMAI METHODS MODEL
- 2.9 FOURTH GENERATION TECHNIQUES
- 2.10 PROCESS TECHNOLOGY
- 2.11 PRODUCT AND PROCESS
- 2.12SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES
- PART TWO MANAGING SOFTWARE PROJECTS
- CHAPTER 3 PROJECT MANAGEMENT CONCEPTS
- 3.1 THE MANAGEMENT SPECTRUM
- 3.1.1 People
- 3.1.2 TheProblem
- 3.1.3 TheProcess
- 3.2 PEOPLE
- 3.2.1 ThePlayers
- 3.2.2 Team leaders
- 3.2.3 The Software Team
- 3.2.4 Coordination and Communication Issues
- 3.3 THEPROBLEM
- 3.3.1 Soflware Scope
- 3.3.2 Problem Decomposition
- 3.4 THEPROCESS
- 3.4.1 Melding the Problem and the Process
- 3.4.2 Process Decomposition
- 3.5 THEPRQIECT
- 3.6 SUMMARY

## REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 4 SOFTWARE PROCESS AND PROJECT METRICS

### 4.1 MEASURES, METRICS, AND INDICATORS

### 4.2 METRICS IN THE PROCESS AND PROJECT DOMAINS

#### 4.2.1 Process Metrics and Software Process Improvement

#### 4.2.2 Project Metrics

### 4.3 SOFTWARE MEASUREMENT

#### 4.3.1 Size-Oriented Metrics

#### 4.3.2 Function-Oriented Metrics

#### 4.3.3 Extended Function Point Metrics

### 4.4 RECONCILING DIFFERENT METRICS APPROACHES

### 4.5 METRICS FOR SOFTWARE QUALITY

#### 4.5.1 An Overview of Factors That Affect Quality

#### 4.5.2 Measuring Quality

#### 4.5.3 Defect Removal Efficiency

### 4.6 INTEGRATING METRICS WITHIN THE SOFTWARE PROCESS

### 4.7 SUMMARY

## REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 5 SOFTWARE PROJECT PIANNING

### 5.1 OBSERVATIONS ON ESTIMATING

### 5.2 PROJECT PIANNING OBJECTIVES

### 5.3 SOFTWARE SCOPE

#### 5.3.1 Obtaining Information Necessary for Scope

#### 5.3.2 A Scoping Example

### 5.4 RESOURCES

#### 5.4.1 Human Resources

#### 5.4.2 Reusable Software Resources

#### 5.4.3 Environmental Resources

### 5.5 SOFTWARE PROJECT ESTIMATION

### 5.6 DECOMPOSITION TECHNIQUES

#### 5.6.1 Software Sizing

#### 5.6.2 Problem-Based Estimation

#### 5.6.3 An Example of LOC-Based Estimation

#### 5.6.4 An Example of FP-Based Estimation

#### 5.6.5 Process-Based Estimation

#### 5.6.6 An Example of Process-Based Estimation

### 5.7 EMPIRICAL ESTIMATION MODELS

#### 5.7.1 The Structure of Estimation Models

#### 5.7.2 The COCOMO Model

#### 5.7.3 The Software Equation

### 5.8 THE MAKE-BUY DECISION

#### 5.8.1 Creating a Decision Tree

#### 5.8.2 Outsourcing

### 5.9 AUTOMATED ESTIMATION TOOLS

### 5.10 SUMMARY

## REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 6 RISK MANAGEMENT

### 6.1 REACTIVE VS. PROACTIVE RISK STRATEGIES

### 6.2 SOFTWARE RISKS

### 6.3 RISK IDENTIFICATION

#### 6.3.1 Product Size Risks

#### 6.3.2 Business Impact Risks

#### 6.3.3 Customer-Related Risks

#### 6.3.4 Process Risks

#### 6.3.5 Technology Risk

#### 6.3.6 Development Environment Risks

#### 6.3.7 Risks Associated with Staff Size and Experience

#### 6.3.8 Risk Components and Drivers

### 6.4 RISK PROJECTION

#### 6.4.1 Developing a Risk Table

#### 6.4.2 Assessing Risk Impact

#### 6.4.3 Risk Assessment

### 6.5 RISK MITIGATION, MONITORING, AND MANAGEMENT

### 6.6 SAFETY RISKS AND HAZARDS

### 6.7 THERMMMPIAN

### 6.8 SUMMARY

## REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 7 PROJECT SCHEDULING AND TRACKING

### 7.1 BASIC CONCEPTS

#### 7.1.1 Comments on "Lateness"

#### 7.1.2 Basic Principles

### 7.2 THE RELATIONSHIP BETWEEN PEOPLE AND EFFORT

#### 7.2.1 An Example

#### 7.2.2 An Empirical Relationship

#### 7.2.3 Effort Distribution

### 7.3 DEFINING A TASK SET FOR THE SOFTWARE PROJECT

#### 7.3.1 Degree of Rigor

#### 7.3.2 Defining Adaptation Criteria

#### 7.3.3 Computing a Task Set Selector Value

#### 7.3.4 Interpreting the TSS Value and Selecting the Task Set

### 7.4 SELECTING SOFTWARE ENGINEERING TASKS

### 7.5 REFINEMENT OF MAJOR TASKS

### 7.6 DEFINING A TASK NETWORK

### 7.7 SCHEDULING

#### 7.7.1 Timeline Charts

#### 7.7.2 Tracking the Schedule

### 7.8 THE PROJECT PLAN

### 7.9 SUMMARY

## REFERENCES

### PROBLEMS AND POINTS TO PONDER

## FURTHER READINGS AND OTHER INFORMATION SOURCES

### CHAPTER 8 SOFTWARE QUALITY ASSURANCE

#### 8.1 QUALITY CONCEPTS

##### 8.1.1 Quality

##### 8.1.2 Quality Control

##### 8.1.3 Quality Assurance

##### 8.1.4 Cost of Quality

#### 8.2 THE QUALITY MOVEMENT

#### 8.3 SOFTWARE QUALITY ASSURANCE

##### 8.3.1 Background Issues

##### 8.3.2 SQA Activities

#### 8.4 SOFTWARE REVIEWS

##### 8.4.1 Cost Impact of Software Defects

##### 8.4.2 Defect Amplification and Removal

#### 8.5 FORMAL TECHNICAL REVIEWS

##### 8.5.1 The Review Meeting

##### 8.5.2 Review Reporting and Record Keeping

##### 8.5.3 Review Guidelines

#### 8.6 FORMAL APPROACHES TO SQA

#### 8.7 STATISTICAL QUALITY ASSURANCE

#### 8.8 SOFTWARE RELIABILITY

##### 8.8.1 Measures of Reliability and Availability

##### 8.8.2 Software Safety and Hazard Analysis

#### 8.9 THE SQAPIAN

#### 8.10 THE ISO 9000 QUALITY STANDARDS

##### 8.10.1 The ISO Approach to Quality Assurance Systems

##### 8.10.2 The ISO 9001 Standard

#### 8.11 SUMMARY

#### REFERENCES

#### PROBLEMS AND POINTS TO PONDER

## FURTHER READINGS AND OTHER INFORMATION SOURCES

### CHAPTER 9 SOFTWARE CONFIGURATION MANAGEMENT

#### 9.1 SOFTWARE CONFIGURATION MANAGEMENT

##### 9.1.1 Baselines

##### 9.1.2 Software Configuration Items

#### 9.2 THE SCM PROCESS

#### 9.3 IDENTIFICATION OF OBJECTS IN THE SOFTWARE CONFIGURATION

#### 9.4 VERSION CONTROL

#### 9.5 CHANGE CONTROL

#### 9.6 CONFIGURATION AUDIT

#### 9.7 STATUS REPORTING

#### 9.8 SCM STANDARDS

#### 9.9 SUMMARY

#### REFERENCES

#### PROBLEMS AND POINTS TO PONDER

## FURTHER READINGS AND OTHER INFORMATION SOURCES

### PART THREE CONVENTIONAL METHODS FOR SOFTWARE DEVELOPMENT

### CHAPTER 10 SYSTEM ENGINEERING

#### 10.1 COMPUTER-BASED SYSTEMS



## 10.2 THE SYSTEM ENGINEERING HIERARCHY

### 10.2.1 System Modeling

### 10.2.2 Information Engineering: An Overview

### 10.2.3 Product Engineering: An Overview

## 10.3 INFORMATION ENGINEERING

## 10.4 INFORMATION STRATEGY PIANNING

### 10.4.1 Enterprise Modeling

### 10.4.2 Business-level Dala Modeling

## 10.5 BUSINESS AREA ANALYSIS

### 10.5.1 Process Modeling

### 10.5.2 Information Flow Modeling

## 10.6 PRODUCT ENGINEERING

### 10.6.1 System Analysis

### 10.6.2 Identification of Need

### 10.6.3 Feasibility Study

### 10.6.4 EconomicAnalysis

### 10.6.5 Technical Analysis

## 10.7 MODELNG THE SYSTEM ARCHITECTURE

## 10.8 SYSTEM MODEUNG AND SIMUIATION

## 10.9 SYSTEM SPECIFICATION

## 10.10SUMMARY

## REFERENCES

## PROBLEMS AND POINTS TO PONDER

## FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 11 ANALYSIS CONCEPTS AND PRINCIPLES

### 11.1 REQUIREMENTS ANALYSIS

### 11.2 COMMUNICATION TECHNIQUES

#### 11.2.1 Initiating the Process

#### 11.2.2 Facilitaed Application Specification Techniques

#### 11.2.3 Qualily Function Deployment

### 11.3 ANALYSIS PRINCIPLES

#### 11.3.1 The Information Domain

#### 11.3.2 Modeling

#### 11.3.3 Partitioning

#### 11.3.4 Essential and Implementation Views

### 11.4 SOFTWARE PROTOTYPING

#### 11.4.1 Selecting the Prototyping Approach

#### 11.4.2 Prototyping Methods and Tools

### 11.5 SPECIFICATION

#### 11.5.1 Specification Principles

#### 11.5.2 Representation

#### 11.5.3 The Software Requirements Specification

### 11.6 SPECIFICATION REVIEW

### 11.7 SUMMARY

## REFERENCES

## PROBIEMS AND POINTS TO PONDER

## FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 12 ANALYSIS MODELING

### 12.1 A BRIEF HISTORY

- 12.2 THE ELEMENTS OF THE ANALYSIS MODEL
- 12.3 DATAMODELING
  - 12.3.1 Data Objects, Attributes, and Relationships
  - 12.3.2 Cardinality and Modality
  - 12.3.3 Entity-Relationship Diagrams
- 12.4 FUNCTIONAL MODELING AND INFORMATION FLOW
  - 12.4.1 Data Flow Diagrams
  - 12.4.2 Extensions for Real-Time Systems
  - 12.4.3 Ward and Mellor Extensions
  - 12.4.4 Hatley and Pirbhai Extensions
- 12.5 BEHAVIORAL MODELING
- 12.6 THE MECHANICS OF STRUCTURED ANALYSIS
  - 12.6.1 Creating an Entity-Relationship Diagram
  - 12.6.2 Creating a Data Flow Model
  - 12.6.3 Creating a Control Flow Model
  - 12.6.4 The Control Specification
  - 12.6.5 The Process Specification
- 12.7 THE DATA DICTIONARY
- 12.8 AN OVERVIEW OF OTHER CLASSICAL ANALYSIS METHODS
  - 12.8.1 Data Structured Systems Development
  - 12.8.2 Jackson System Development
  - 12.8.3 SADT
- 12.9 SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES
- CHAPTER 13 DESIGN CONCEPTS AND PRINCIPLES
  - 13.1 SOFTWARE DESIGN AND SOFTWARE ENGINEERING
  - 13.2 THE DESIGN PROCESS
    - 13.2.1 Design and Software Quality
    - 13.2.2 The Evolution of Software Design
  - 13.3 DESIGN PRINCIPLES
  - 13.4 DESIGN CONCEPTS
    - 13.4.1 Abstraction
    - 13.4.2 Refinement
    - 13.4.3 Modularity
    - 13.4.4 Software Architecture
    - 13.4.5 Control Hierarchy
    - 13.4.6 Structural Partitioning
    - 13.4.7 Data Structure
    - 13.4.8 Software Procedure
    - 13.4.9 Information Hiding
  - 13.5 EFFECTIVE MODULAR DESIGN
    - 13.5.1 Functional Independence
    - 13.5.2 Cohesion
    - 13.5.3 Coupling
  - 13.6 DESIGN HEURISTICS FOR EFFECTIVE MODULARITY
  - 13.7 THE DESIGN MODEL
  - 13.8 DESIGN DOCUMENTATION

13.9 SUMMARY

REFERENCES

PROBLEMS AND POINTS TO PONDER

FURTHER READINGS AND OTHER INFORMATION SOURCES

CHAPTER 14 DESIGN METHODS

14.1 DATA DESIGN

14.2 ARCHITECTURAL DESIGN

14.2.1 Contributors

14.2.2 Areas of Application

14.3 THE ARCHITECTURAL DESIGN PROCESS

14.3.1 Transform Flow

14.3.2 Transaction Flow

14.4 TRANSFORM MAPPING

14.4.1 An Example

14.4.2 Design Steps

14.5 TRANSACTION MAPPING

14.5.1 An Example

14.5.2 Design Steps

14.6 DESIGN POSTPROCESSING

14.7 ARCHITECTURAL DESIGN OPTIMIZATION

14.8 INTERFACE DESIGN

14.8.1 Internal and External Interface Design

14.8.2 User Interface Design

14.9 HUMAN-COMPUTER INTERFACE DESIGN

14.9.1 Interface Design Models

14.9.2 Task Analysis and Modeling

14.9.3 Design Issues

14.9.4 Implementation Tools

14.9.5 Design Evaluation

14.10 INTERFACE DESIGN GUIDELINES

14.10.1 General Interaction

14.10.2 Information Display

14.10.3 Data Input

14.11 PROCEDURAL DESIGN

14.11.1 Structured Programming

14.11.2 Graphical Design Notation

14.11.3 Tabular Design Notation

14.11.4 Program Design Language

14.11.5 APDL Example

14.12 SUMMARY

REFERENCES

PROBLEMS AND POINTS TO PONDER

FURTHER READINGS AND OTHER INFORMATION SOURCES

CHAPTER 15 DESIGN FOR REAL-TIME SYSTEMS

15.1 SYSTEM CONSIDERATIONS

15.2 REAL-TIME SYSTEMS

15.2.1 Integration and Performance Issues

15.2.2 Interrupt Handling

15.2.3 Real-Time Data Bases

15.2.4. Real-Time Operating Systems

15.2.5 Real-Time Languages

15.2.6 Task Synchronization and Communication

15.3 ANALYSIS AND SIMULATION OF REAL-TIME SYSTEMS

15.3.1 Mathematical Tools for Real-Time System Analysis

15.3.2 Simulation and Modeling Techniques

15.4 REAL-TIME DESIGN

15.5 SUMMARY

REFERENCES

PROBLEMS AND POINTS TO PONDER

FURTHER READINGS AND OTHER INFORMATION SOURCES

CHAPTER 16 SOFTWARE TESTING METHODS

16.1 SOFTWARE TESTING FUNDAMENTALS

16.1.1 Testing Objectives

16.1.2 Testing Principles

16.1.3 Testability

16.2 TEST CASE DESIGN

16.3 WHITE BOX TESTING

16.4 BASIS PATH TESTING

16.4.1 Flow Graph Notation

16.4.2 Cyclomatic Complexity

16.4.3 Deriving Test Cases

16.4.4 Graph Matrices

16.5 CONTROL STRUCTURE TESTING

16.5.1 Condition Testing

16.5.2 Data Flow Testing

16.5.3 Loop Testing

16.6 BLACK-BOX TESTING

16.6.1 Graph-Based Testing Methods

16.6.2 Equivalence Partitioning

16.6.3 Boundary Value Analysis

16.6.4 Comparison Testing

16.7 TESTING FOR SPECIALIZED ENVIRONMENTS

16.7.1 Testing GUIs

16.7.2 Testing of Client/Server Architectures

16.7.3 Testing Documentation and Help Facilities

16.7.4 Testing for Real-Time Systems

16.8 SUMMARY

REFERENCES .

PROBLEMS AND POINTS TO PONDER

FURTHER READINGS AND OTHER INFORMATION SOURCES

CHAPTER 17 SOFTWARE TESTING STRATEGIES

17.1 A STRATEGIC APPROACH -TO SOFTWARE TESTING

17.1.1 Verification and Validation

17.1.2 Organizing for Software Testing

17.1.3 A Software Testing Strategy

17.1.4 Criteria for Completion or Testing

17.2 STRATEGIC ISSUES

17.3 UNIT TESTING

- 17.3.1 Unit Test Considerations
- 17.3.2 Unit Test Procedures
- 17.4 INTEGRATION TESTING
- 17.4.1 Top-Down Integration
- 17.4.2 Bottom-Up Integration
- 17.4.3 Regression Testing
- 17.4.4 Comments on Integration Testing
- 17.4.5 Integration Test Documentation
- 17.5 VALIDATION TESTING
- 17.5.1 Validation Test Criteria
- 17.5.2 Configuration Review
- 17.5.3 Alpha and Beta Testing
- 17.6 SYSTEM TESTING
- 17.6.1 Recovery Testing
- 17.6.2 Security Testing
- 17.6.3 Stress Testing
- 17.6.4 Performance Testing
- 17.7 THE ART OF DEBUGGING
- 17.7.1 The Debugging Process
- 17.7.2 Psychological Considerations
- 17.7.3 Debugging Approaches
- 17.8 SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES
- CHAPTER 18 TECHNICAL METRICS FOR SOFTWARE
- 18.1 SOFTWARE QUALITY
- 18.1.1 McCall's Quality Factors
- 18.1.2 FURPS
- 18.1.3 The Transition to a Quantitative View
- 18.2 A FRAMEWORK FOR TECHNICAL SOFTWARE METRICS
- 18.2.1 The Challenge of Technical Metrics
- 18.2.2 Measurement Principles
- 18.2.3 The Attributes of Effective Software Metrics
- 18.3 METRICS FOR THE ANALYSIS MODEL
- 18.3.1 Function-Based Metrics
- 18.3.2 The Bang Metric
- 18.3.3 Metrics for Specification Quality
- 18.4 METRICS FOR THE DESIGN MODEL
- 18.4.1 Highlevel Design Metrics
- 18.4.2 Component level Design Metrics
- 18.4.3 Interface Design Metrics
- 18.5 METRICS FOR SOURCE CODE
- 18.6 METRICS FOR TESTING
- 18.7 METRICS FOR MAINTENANCE
- 18.8 SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES

## PART FOUR OBJECT-ORIENTED SOFTWARE ENGINEERING CHAPTER 19 OBJECT-ORIENTED CONCEPTS AND PRINCIPLES

### 19.1 THE OBJECT-ORIENTED PARADIGM

### 19.2 OBJECT-ORIENTED CONCEPTS

#### 19.2.1 Classes and Objects

#### 19.2.2 Attributes

#### 19.2.3 Operations, Methods and Services

#### 19.2.4 Messages

#### 19.2.5 Encapsulation, Inheritance, and Polymorphism

### 19.3 IDENTIFYING THE ELEMENTS OF AN OBJECT MODEL

#### 19.3.1 Identifying Classes and Objects

#### 19.3.2 Specifying Attributes

#### 19.3.3 Defining Operations

#### 19.3.4 Finalizing the Object Definition

### 19.4 MANAGEMENT OF OBJECT-ORIENTED SOFTWARE PROJECTS

#### 19.4.1 The Common Process Framework for OO

#### 19.4.2 Object-Oriented Project Metrics and Estimation

#### 19.4.3 An OO Estimating and Scheduling Approach

#### 19.4.4 Progress for an Object-Oriented Project

### 19.5 SUMMARY

### REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 20 OBJECT-ORIENTED ANALYSIS

### 20.1 OBJECT-ORIENTED ANALYSIS

#### 20.1.1 Conventional vs. OO Approaches

#### 20.1.2 The OOA Landscape

### 20.2 DOMAIN ANALYSIS

#### 20.2.1 Reuse and Domain Analysis

#### 20.2.2 The Domain Analysis Process

### 20.3 GENERIC COMPONENTS OF THE OO ANALYSIS MODEL

### 20.4 THE OOA PROCESS

#### 20.4.1 Use Cases

#### 20.4.2 Class-Responsibility-Collaborator Modeling

#### 20.4.3 Defining Structures and Hierarchies

#### 20.4.4 Defining Subjects and Subsystems

### 20.5 THE OBJECT-RELATIONSHIP MODEL

### 20.6 THE OBJECT-BEHAVIOR MODEL

#### 20.6.1 Event Identification with Use Cases

#### 20.6.2 State Representations

### 20.7 SUMMARY

### REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 21 OBJECT-ORIENTED DESIGN

### 21.1 DESIGN FOR OBJECT-ORIENTED SYSTEMS

#### 21.1.1 Conventional vs. OO Approaches

#### 21.1.2 Design Issues

#### 21.1.3 The OOD Landscape

## 21.2 THE GENERIC COMPONENTS OF THE OO DESIGN MODEL

### 21.3 THE SYSTEM DESIGN PROCESS

#### 21.3.1 Partitioning the Analysis Model

#### 21.3.2 Concurrency and Subsystem Allocation

#### 21.3.3 The Task Management Component

#### 21.3.4 The Data Management Component

#### 21.3.5 The Resource Management Component

#### 21.3.6 The Human-Computer Interface Component

#### 21.3.7 Inter-Subsystem Communication

### 21.4 THE OBJECT DESIGN PROCESS

#### 21.4.1 Object Descriptions

#### 21.4.2 Designing Algorithms and Data Structures

#### 21.4.3 Program Components and Interfaces

### 21.5 DESIGN PATTERNS

#### 21.5.1 Describing a Design Pattern

#### 21.5.2 Using Patterns in Design

### 21.6 OBJECT-ORIENTED PROGRAMMING

### 21.7 SUMMARY

### REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 22 OBJECT-ORIENTED TESTING

### 22.1 BROADENING THE VIEW OF TESTING

### 22.2 TESTING OOA AND OOD MODELS

#### 22.2.1 Correctness of OOA and OOD Models

#### 22.2.2 Consistency of OOA and OOD Models

### 22.3 OBJECT-ORIENTED TESTING STRATEGIES

#### 22.3.1 Unit Testing in the OO Context

#### 22.3.2 Integration Testing in the OO Context

#### 22.3.3 Validation Testing in an OO Context

### 22.4 TEST CASE DESIGN FOR OO SOFTWARE

#### 22.4.1 The Test Case Design Implications of OO Concepts

#### 22.4.2 Applicability of Conventional Test Case Design Methods

#### 22.4.3 Fault-Based Testing

#### 22.4.4 The Impact of OO Programming on Testing

#### 22.4.5 Test Cases and the Class Hierarchy

#### 22.4.6 Scenario-Based Test Design

#### 22.4.7 Testing Surface Structure and Deep Structure

### 22.5 TESTING METHODS APPLICABLE AT THE CLASS LEVEL

#### 22.5.1 Random Testing for OO Classes

#### 22.5.2 Partition Testing at the Class Level

### 22.6 INTERCLASS TEST CASE DESIGN

#### 22.6.1 Multiple Class Testing

#### 22.6.2 Tests Derived from Behavior Models

### 22.7 SUMMARY

### REFERENCES

### PROBLEMS AND POINTS TO PONDER

### FURTHER READINGS AND OTHER INFORMATION SOURCES

## CHAPTER 23 TECHNICAL METRICS FOR OBJECT-ORIENTED SYSTEMS

- 23.1 THE INTENT OF OBJECTORIENTED METRICS
- 23.2 THE DISTINGUISHING CHARACTERISTICS
  - 23.2.1 Localization
  - 23.2.2 Encapsulation
  - 23.2.3 Information hiding
  - 23.2.4 Inheritance
  - 23.2.5 Abstraction
- 23.3 METRICS FOR THE OO DESIGN MODEL
- 23.4 CLASS-ORIENTED METRICS
  - 23.4.1 The CK Metrics Suite
  - 23.4.2 Metrics Proposed by Lorenz and Kidd
- 23.5 OPERATION-ORIENTED METRICS
- 23.6 METRICS FOR OBJECT-ORIENTED TESTING
- 23.7 METRICS FOR OBJECTORIENTED PROJECTS
- 23.8 SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES
- CHAPTER 24 FORMAL METHODS
  - 24.1 BASIC CONCEPTS
    - 24.1.1 Deficiencies of Less Formal Approaches
    - 24.1.2 Mathematics in Software Development
    - 24.1.3 Formal Methods Concepts
  - 24.2 MATHEMATICAL PRELIMINARIES
    - 24.2.1 Sets and Constructive Specification
    - 24.2.2 Set Operators
    - 24.2.3 Logic Operators
  - 24.3 APPLYING MATHEMATICAL NOTATION FOR FORMAL SPECIFICATION
  - 24.4 FORMAL SPECIFICATION LANGUAGES
  - 24.5 USING Z TO REPRESENT AN EXAMPLE SOFTWARE COMPONENT
  - 24.6 THE TEN COMMANDMENTS OF FORMAL METHODS
  - 24.7 FORMAL METHODS-THE ROAD AHEAD
  - 24.8 SUMMARY
  - REFERENCES
  - PROBLEMS AND POINTS TO PONDER
  - FURTHER READINGS AND OTHER INFORMATION SOURCES
- CHAPTER 25 CLEANROOM SOFTWARE ENGINEERING
  - 25.1 THE CLEANROOM APPROACH
    - 25.1.1 The Cleanroom Strategy
    - 25.1.2 What Makes Cleanroom Different?
  - 25.2 FUNCTIONAL SPECIFICATION
    - 25.2.1 Black-Box Specification
    - 25.2.2 State-Box Specification
    - 25.2.3 Clear-Box Specification
  - 25.3 DESIGN REFINEMENT AND VERIFICATION
    - 25.3.1 Design Refinement and Verification
    - 25.3.2 Advantages of Design Verification
  - 25.4 CLEANROOM TESTING
    - 25.4.1 Statistical Use Testing



25.4.2 Certification

25.5 SUMMARY .

REFERENCES

PROBLEMS AND POINTS TO PONDER

FURTHER READINGS AND OTHER INFORMATION SOURCES

CHAPTER 26 SOFTWARE REUSE

26.1 MANAGEMENT ISSUES

26.1.1 Roadblocks to Reuse

26.1.2 A Hardware Analogy

26.1.3 Some Suggestions for Establishing an Approach to Reuse

26.2 THE REUSE PROCESS

26.2.1 Reusable Artifacts

26.2.2 A Process Model

26.3 DOMAIN ENGINEERING

26.3.1 The Domain Analysis Process

26.3.2 Characterization Functions

26.3.3 Structural Modeling and Structure Points

26.4 BUILDING REUSABLE COMPONENTS

26.4.1 Analysis and Design for Reuse

26.4.2 Construction Methods

26.4.3 Component-Based Development

26.5 CLASSIFYING AND RETRIEVING COMPONENTS

26.5.1 Describing Reusable Components

26.5.2 The Reuse Environment

26.6 ECONOMICS OF SOFTWARE REUSE

26.6.1 Impact on Quality, Productivity and Cost

26.6.2 Cost Analysis Using Structure Points

26.6.3 Reuse Metrics

26.7 SUMMARY

REFERENCES

PROBLEMS AND POINTS TO PONDER

FURTHER READINGS AND OTHER INFORMATION SOURCES

CHAPTER 27 REENGINEERING

27.1 BUSINESS PROCESS REENGINEERING

27.1.1 Business Processes

27.1.2 Principles of Business Process Reengineering

27.1.3 ABPR Model

27.1.4 Words of Warning

27.2 SOFTWARE REENGINEERING

27.2.1 Software Maintenance

27.2.2 A Software Reengineering Process Model

27.3 REVERSE ENGINEERING

27.3.1 Reverse Engineering to Understand Processing

27.3.2 Reverse Engineering to Understand Data

27.3.3 Reverse Engineering User Interfaces

27.4 RESTRUCTURING

27.4.1 Code Restructuring

27.4.2 Data Restructuring

27.5 FORWARD ENGINEERING

- 27.5.1 Forward Engineering for Client/Server Architectures
- 27.5.2 Forward Engineering for Object-Oriented Architectures
- 27.5.3 Forward Engineering User Interfaces
- 27.6 THE ECONOMICS OF REENGINEERING
- 27.7 SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES
- CHAPTER 28 CLIENT/SERVER SOFTWARE ENGINEERING
- 28.1 THE STRUCTURE OF CLIENT/SERVER SYSTEMS
  - 28.1.1 Software Components for C/S Systems
  - 28.1.2 The Distribution of Software Components
  - 28.1.3 Guidelines for Distributing Application Components
  - 28.1.4 linking C/S Software Components
  - 28.1.5 Middleware and Object Request Broker
- 28.2 SOFTWARE ENGINEERING FOR C/S SYSTEMS
- 28.3 ANALYSIS MODEUNG iSSUES
- 28.4 DESIGN FOR C/S SYSTEMS
  - 28.4.1 Conventional Design Approaches
  - 28.4.2 Database Design
  - 28.4.3 An Overview of a Design Approach
  - 28.4.4 Process Design iteration
- 28.5 TESTING iSSUES
  - 28.5.1 Overall C/S Testing Strategy
  - 28.5.2 C/S Testing Tactics
- 28.6 SUMMARY
- REFERENCES
- PROBLEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES
- CHAPTER 29 COMPUTER-AIDED SOFTWARE ENGINEERING
- 29.1 WHATISCASE?
- 29.2 BUILDING BLOCKS FOR CASE
- 29.3 A TAXONOMY OF CASE TOOLS
- 29.4 INTEGRATED CASE ENVIRONMENTS
- 29.5 THE INTEGRATION ARCHITECTURE
- 29.6 THE CASE REPOSITORY
  - 29.6.1 The Roe of the Repository in I-CASE
  - 29.6.2 Features and Content
- 29.7 SUMMARY
- REFERENCES
- PROBIEMS AND POINTS TO PONDER
- FURTHER READINGS AND OTHER INFORMATION SOURCES
- CHAPTER 30 THE ROAD AHEAD
- 30.1 THE IMPORTANCE OF SOFTWARE-REVISITED
- 30.2 THE SCOPE OF CHANGE
- 30.3 PEOPLE AND THE WAY THEY BUILD SYSTEMS
- 30.4 THE "NEW" SOFTWARE PROCESS
- 30.5 NEW MODES FOR REPRESENTING INFORMATION
- 30.6 TECHNOLOGY AS A DRIVER

30.7 A CONCLUDING COMMENT

REFERENCES

PROBLEMS AND POINTS TO PONDER

FURTHER READINGS AND OTHER INFORMATION SOURCES

## 精彩短评

1、目前图书馆少数能找到的关于软件工程的书

## 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：[www.tushu000.com](http://www.tushu000.com)