

Contents

Foreword	xi
Preface	xv
Guided tour	xxii
Technology to enhance learning and teaching	xxiv
1 Software and software engineering	1
1.1 The nature of software	1
1.2 What is software engineering?	6
1.3 Software engineering as a branch of the engineering profession	8
1.4 Stakeholders in software engineering	10
1.5 Software quality	11
1.6 Software engineering projects	14
1.7 Activities common to software projects	16
1.8 The themes emphasized in this book	20
1.9 Difficulties and risks in software engineering as a whole	24
1.10 Summary	26
1.11 For more information	26
2 Review of object orientation	29
2.1 What is object orientation?	29
2.2 Classes and objects	31
2.3 Instance variables	36
2.4 Methods, operations and polymorphism	38
2.5 Organizing classes into inheritance hierarchies	39
2.6 The effect of inheritance hierarchies on polymorphism and variable declarations	45
2.7 Concepts that define object orientation	52
2.8 A program for manipulating postal codes	55
2.9 Classes for representing geometric points	57
2.10 Measuring the quality and complexity of a program	60
2.11 Difficulties and risks in programming language choice and OO programming	62
2.12 Summary	63
2.13 For more information	63

- 3 Basing software development on reusable technology
 - 3.1 Reuse: building on the work and experience of others
 - 3.2 Incorporating reusability and reuse into software engineering
 - 3.3 Frameworks: reusable subsystems
 - 3.4 The client–server architecture
 - 3.5 Technology needed to build client–server systems
 - 3.6 The Object Client–Server Framework (OCSF)
 - 3.7 Basic description of OCSF – client side
 - 3.8 Basic description of OCSF – server side
 - 3.9 An instant messaging application using the OCSF
 - 3.10 Difficulties and risks when considering reusable technology and client–server systems
 - 3.11 Summary
 - 3.12 For more information

- 4 Developing requirements
 - 4.1 Domain analysis
 - 4.2 The starting point for software projects
 - 4.3 Defining the problem and the scope
 - 4.4 What is a requirement?
 - 4.5 Types of requirements
 - 4.6 Use cases: describing how the user will use the system
 - 4.7 Some techniques for gathering requirements
 - 4.8 Types of requirements document
 - 4.9 Reviewing requirements
 - 4.10 Managing changing requirements
 - 4.11 GPS-based Automobile Navigation Assistant (GANA)
 - 4.12 Requirements for a feature of the SimpleChat instant messaging program
 - 4.13 Difficulties and risks in domain and requirements analysis
 - 4.14 Summary
 - 4.15 For more information

- 5 Modeling with classes
 - 5.1 What is UML?
 - 5.2 Essentials of UML class diagrams
 - 5.3 Associations and multiplicity
 - 5.4 Generalization
 - 5.5 Object diagrams
 - 5.6 More advanced features of class diagrams
 - 5.7 The basics of Object Constraint Language (OCL)
 - 5.8 A class diagram for genealogy
 - 5.9 The process of developing class diagrams
 - 5.10 Implementing class diagrams in Java
 - 5.11 Difficulties and risks when creating class diagrams
 - 5.12 Summary
 - 5.13 For more information

6 Using design patterns

- 6.1 Introduction to patterns
- 6.2 The Abstraction–Occurrence pattern
- 6.3 The General Hierarchy pattern
- 6.4 The Player–Role pattern
- 6.5 The Singleton pattern
- 6.6 The Observer pattern
- 6.7 The Delegation pattern
- 6.8 The Adapter pattern
- 6.9 The Façade pattern
- 6.10 The Immutable pattern
- 6.11 The Read-Only Interface pattern
- 6.12 The Proxy pattern
- 6.13 The Factory pattern
- 6.14 Enhancing OCSF to employ additional design patterns
- 6.15 Difficulties and risks when using design patterns
- 6.16 Summary
- 6.17 For more information

7 Focusing on users and their tasks

- 7.1 User-centered design
- 7.2 Characteristics of users
- 7.3 The basics of user interface design
- 7.4 Usability principles
- 7.5 Evaluating user interfaces
- 7.6 Implementing a simple GUI in Java
- 7.7 Difficulties and risks in user-centered design
- 7.8 Summary
- 7.9 For more information

8 Modeling interactions and behavior

- 8.1 Interaction diagrams
- 8.2 State diagrams
- 8.3 Activity diagrams
- 8.4 Implementing classes based on interaction and state diagrams
- 8.5 Difficulties and risks in modeling interactions and behavior
- 8.6 Summary
- 8.7 For more information

needs
Pseudocode

9 Architecting and designing software

- 9.1 The process of design
- 9.2 Principles leading to good design
 - Design Principle 1: Divide and conquer*
 - Design Principle 2: Increase cohesion where possible*
 - Design Principle 3: Reduce coupling where possible*
 - Design Principle 4: Keep the level of abstraction as high as possible*

Design Principle 5: Increase reusability where possible
Design Principle 6: Reuse existing designs and code where possible
Design Principle 7: Design for flexibility
Design Principle 8: Anticipate obsolescence
Design Principle 9: Design for portability
Design Principle 10: Design for testability
Design Principle 11: Design defensively

- 9.3 Techniques for making good design decisions
- 9.4 Model Driven Development
- 9.5 Software architecture
- 9.6 Architectural patterns
 - The Multi-Layer architectural pattern*
 - The Client-Server and other distributed architectural patterns*
 - The Broker architectural pattern*
 - The Transaction Processing architectural pattern*
 - The Pipe-and-Filter architectural pattern*
 - The Model-View-Controller (MVC) architectural pattern*
 - The Service-Oriented architectural pattern*
 - The Message-Oriented architectural pattern*
- 9.7 Writing a good design document
- 9.8 Design of a feature for the SimpleChat instant messaging application
- 9.9 Difficulties and risks in design
- 9.10 Summary
- 9.11 For more information

10 Testing and inspecting to ensure high quality

- 10.1 Basic definitions
- 10.2 Effective and efficient testing
- 10.3 Defects in ordinary algorithms
- 10.4 Defects in numerical algorithms
- 10.5 Defects in timing and co-ordination: deadlocks, livelocks and critical races
- 10.6 Defects in handling stress and unusual situations
- 10.7 Documentation defects
- 10.8 Writing formal test cases and test plans
- 10.9 Strategies for testing large systems
- 10.10 Inspections
- 10.11 Quality assurance in general
- 10.12 Test cases for phase 2 of the SimpleChat instant messaging system
- 10.13 Difficulties and risks in quality assurance
- 10.14 Summary
- 10.15 For more information

11 Managing the software process

- 11.1 What is project management?
- 11.2 Software process models

11.3	Cost estimation	435
11.4	Building software engineering teams	445
11.5	Project scheduling and tracking	449
11.6	Contents of a project plan	452
11.7	Difficulties and risks in project management	453
11.8	Summary	455
11.9	For more information	456
12	Review	459
12.1	Theme 1: Understanding the customer and user	459
12.2	Theme 2: Basing development on solid principles and reusable technology	459
12.3	Theme 3: Object orientation	464
12.4	Theme 4: Visual modeling using UML	464
12.5	Theme 5: Evaluation of alternatives in requirements and design	465
12.6	Theme 6: Incorporating quantitative and logical thinking	465
12.7	Theme 7: Iterative and agile development	466
12.8	Theme 8: Communicating effectively using documentation	467
12.9	Theme 9: Risk management in all software engineering activities	467
12.10	Where next?	469
Appendix A: Summary of the UML notation used in this book		471
Appendix B: Summary of the documentation types recommended in this book		475
Appendix C: System descriptions		479
Glossary		485