

Table of Contents

Foreword	xxv
Foreword to the 2014 Edition	xxvi
Foreword to the 2004 Edition	xxvii
Editor	xxix
Knowledge Area Editors	xxix
Contributing Editors	xxx
Steering Group	xxxi
Knowledge Area Editors of Previous SWEBOK Versions	xxxi
Review Team	xxxiii
Acknowledgements	xxxiv
IEEE Computer Society Presidents	xxxiv
Professional and Educational Activities Board, 2024 Membership	xxxiv
Motions Regarding the Approval of SWEBOK Guide V4.0	xxxv
Motions Regarding the Approval of SWEBOK Guide V3.0	xxxv
Motions Regarding the Approval of SWEBOK Guide 2004 Version	xxxvi
Introduction to the Guide	xxxvii

CHAPTER 01	
Software Requirements	1-1

Introduction	1-1
1. Software Requirements Fundamentals	1-2
1.1. Definition of a Software Requirement	1-2
1.2. Categories of Software Requirements	1-3
1.3. Software Product Requirements and Software Project Requirements	1-3
1.4. Functional Requirements	1-4
1.5. Nonfunctional Requirements	1-4
1.6. Technology Constraints	1-4
1.7. Quality of Service Constraints	1-4
1.8. Why Categorize Requirements This Way?	1-5
1.9. System Requirements and Software Requirements	1-5
1.10. Derived Requirements	1-6
1.11. Software Requirements Activities	1-6
2. Requirements Elicitation	1-6
2.1. Requirements Sources	1-6
2.2. Common Requirements Elicitation Techniques	1-7
3. Requirements Analysis	1-8
3.1. Basic Requirements Analysis	1-8
3.2. Economics of Quality of Service Constraints	1-8
3.3. Formal Analysis	1-9

3.4. Addressing Conflict in Requirements	1-10
4. Requirements Specification	1-10
4.1. Unstructured Natural Language Requirements Specification	1-11
4.2. Structured Natural Language Requirements Specification	1-12
4.3. Acceptance Criteria-Based Requirements Specification	1-12
4.4. Model-Based Requirements Specification	1-14
4.5. Additional Attributes of Requirement	1-14
4.6. Incremental and Comprehensive Requirements Specification	1-15
5. Requirements Validation	1-15
5.1. Requirements Reviews	1-15
5.2. Simulation and Execution	1-16
5.3. Prototyping	1-16
6. Requirements Management Activities	1-16
6.1. Requirements Scrubbing	1-16
6.2. Requirements Change Control	1-17
6.3. Scope Matching	1-17
7. Practical Considerations	1-17
7.1. Iterative Nature of the Requirements Process	1-17
7.2. Requirements Prioritization	1-17
7.3. Requirements Tracing	1-18
7.4. Requirements Stability and Volatility	1-19
7.5. Measuring Requirements	1-19
7.6. Requirements Process Quality and Improvement	1-19
8. Software Requirements Tools	1-20
8.1. Requirements Management Tools	1-20
8.2. Requirements Modeling Tools	1-20
8.3. Functional Test Case Generation Tools	1-20
Matrix of Topics vs. Reference Material	1-21
Further Readings	1-22
References	1-23

CHAPTER 02

Software Architecture 2-1

Introduction	2-1
1. Software Architecture Fundamentals	2-1
1.1. The Senses of “Architecture”	2-1
1.2. Stakeholders and Concerns	2-3
1.3. Uses of Architecture	2-4
2. Software Architecture Description	2-4
2.1. Architecture Views and Viewpoints	2-5
2.2. Architecture Patterns, Styles and Reference Architectures	2-6
2.3. Architecture Description Languages and Architecture Frameworks	2-7
2.4. Architecture as Significant Decisions	2-7
3. Software Architecture Process	2-8
3.1. Architecture in Context	2-8
3.1.1. Relation of Architecture to Design	2-9

3.2. Architectural Design	2-9
3.2.1. Architecture Analysis	2-9
3.2.2. Architecture Synthesis	2-9
3.2.3. Architecture Evaluation	2-10
3.3. Architecture Practices, Methods, and Tactics	2-10
3.4. Architecting in the Large	2-10
4. Software Architecture Evaluation	2-10
4.1. Goodness in Architecture	2-10
4.2. Reasoning about Architectures	2-11
4.3. Architecture Reviews	2-11
4.4. Architecture Metrics	2-11
Matrix of Topics vs. Reference Material	2-12
Further Readings	2-13
References	2-14

CHAPTER 03

Software Design

3-1

Introduction	3-1
1. Software Design Fundamentals	3-2
1.1. Design Thinking	3-2
1.2. Context of Software Design	3-2
1.3. Key Issues in Software Design	3-3
1.4. Software Design Principles	3-3
2. Software Design Processes	3-5
2.1. High-Level Design	3-6
2.2. Detailed Design	3-6
3. Software Design Qualities	3-6
3.1. Concurrency	3-6
3.2. Control and Event Handling	3-6
3.3. Data Persistence	3-7
3.4. Distribution of Components	3-7
3.5. Errors and Exception Handling, Fault Tolerance	3-7
3.6. Integration and Interoperability	3-7
3.7. Assurance, Security, and Safety	3-7
3.8. Variability	3-7
4. Recording Software Designs	3-7
4.1. Model-Based Design	3-8
4.2. Structural Design Descriptions	3-9
4.3. Behavioral Design Descriptions	3-9
4.4. Design Patterns and Styles	3-10
4.5. Specialized and Domain-Specific Languages	3-10
4.6. Design Rationale	3-11
5. Software Design Strategies and Methods	3-11
5.1. General Strategies	3-11
5.2. Function-Oriented (or Structured) Design	3-11
5.3. Data-Centered Design	3-11

5.4.	Object-Oriented Design	3-11
5.5.	User-Centered Design	3-12
5.6.	Component-Based Design (CBD)	3-12
5.7.	Event-Driven Design	3-12
5.8.	Aspect-Oriented Design (AOD)	3-12
5.9.	Constraint-Based Design	3-12
5.10.	Domain-Driven Design	3-13
5.11.	Other Methods	3-13
6.	Software Design Quality Analysis and Evaluation	3-13
6.1.	Design Reviews and Audits	3-13
6.2.	Quality Attributes	3-13
6.3.	Quality Analysis and Evaluation Techniques	3-13
6.4.	Measures and Metrics	3-14
6.5.	Verification, Validation, and Certification	3-14
	Matrix of Topics vs. Reference Material	3-14
	Further Readings	3-15
	References	3-16

CHAPTER 04

Software Construction 4-1

	Introduction	4-1
1.	Software Construction Fundamentals	4-2
1.1.	Minimizing Complexity	4-2
1.2.	Anticipating and Embracing Change	4-2
1.3.	Constructing for Verification	4-4
1.4.	Reusing Assets	4-4
1.5.	Applying Standards in Construction	4-4
2.	Managing Construction	4-4
2.1.	Construction in Life Cycle Models	4-4
2.2.	Construction Planning	4-5
2.3.	Construction Measurement	4-5
2.4.	Managing Dependencies	4-5
3.	Practical Considerations	4-6
3.1.	Construction Design	4-6
3.2.	Construction Languages	4-6
3.3.	Coding	4-7
3.4.	Construction Testing	4-7
3.5.	Reuse in Construction	4-8
3.6.	Construction Quality	4-8
3.7.	Integration	4-9
3.8.	Cross-Platform Development and Migration	4-9
4.	Construction Technologies	4-10
4.1.	API Design and Use	4-10
4.2.	Object-Oriented Runtime Issues	4-10
4.3.	Parameterization, Templates, and Generics	4-10
4.4.	Assertions, Design by Contract, and Defensive Programming	4-10
4.5.	Error Handling, Exception Handling, and Fault Tolerance	4-11

4.6.	Executable Models	4-11
4.7.	State-Based and Table-Driven Construction Techniques	4-11
4.8.	Runtime Configuration and Internationalization	4-12
4.9.	Grammar-Based Input Processing	4-12
4.10.	Concurrency Primitives	4-12
4.11.	Middleware	4-12
4.12.	Construction Methods for Distributed and Cloud-Based Software	4-13
4.13.	Constructing Heterogeneous Systems	4-13
4.14.	Performance Analysis and Tuning	4-13
4.15.	Platform Standards	4-13
4.16.	Test-First Programming	4-14
4.17.	Feedback Loop for Construction	4-14
5.	Software Construction Tools	4-14
5.1.	Development Environments	4-14
5.2.	Visual Programming and Low-Code/Zero-Code Platforms	4-14
5.3.	Unit Testing Tools	4-15
5.4.	Profiling, Performance Analysis, and Slicing Tools	4-15
	Matrix of Topics vs. Reference Material	4-15
	Further Readings	4-18
	References	4-18

CHAPTER 05

Software Testing

5-1

Introduction	5-1
1. Software Testing Fundamentals	5-3
1.1. Faults vs. Failures	5-3
1.2. Key Issues	5-4
1.2.1. Test Case Creation	5-4
1.2.2. Test Selection and Adequacy Criteria	5-4
1.2.3. Prioritization/Minimization	5-4
1.2.4. Purpose of Testing	5-4
1.2.5. Assessment and Certification	5-4
1.2.6. Testing for Quality Assurance/Improvement	5-4
1.2.7. The Oracle Problem	5-4
1.2.8. Theoretical and Practical Limitations	5-5
1.2.9. The Problem of Infeasible Paths	5-5
1.2.10. Testability	5-5
1.2.11. Test Execution and Automation	5-5
1.2.12. Scalability	5-5
1.2.13. Test Effectiveness	5-5
1.2.14. Controllability, Replication, and Generalization	5-5
1.2.15. Off-Line vs. Online Testing	5-6
1.3. Relationship of Testing to Other Activities	5-6
2. Test Levels	5-6
2.1. The Target of the Test	5-6
2.1.1. Unit Testing	5-6
2.1.2. Integration Testing	5-7

2.1.3. System Testing	5-7
2.1.4. Acceptance Testing	5-7
2.2. Objectives of Testing	5-7
2.2.1. Conformance Testing	5-7
2.2.2. Compliance Testing	5-8
2.2.3. Installation Testing	5-8
2.2.4. Alpha and Beta Testing	5-8
2.2.5. Regression Testing	5-8
2.2.6. Prioritization Testing	5-8
2.2.7. Non-functional Testing	5-8
2.2.8. Security Testing	5-9
2.2.9. Privacy Testing	5-9
2.2.10. Interface and Application Program Interface (API) Testing	5-10
2.2.11. Configuration Testing	5-10
2.2.12. Usability and Human-Computer Interaction Testing	5-10
3. Test Techniques	5-10
3.1. Specification-Based Techniques	5-10
3.1.1. Equivalence Partitioning	5-10
3.1.2. Boundary-Value Analysis	5-11
3.1.3. Syntax Testing	5-11
3.1.4. Combinatorial Test Techniques	5-11
3.1.5. Decision Table	5-11
3.1.6. Cause-Effect Graphing	5-11
3.1.7. State Transition Testing	5-11
3.1.8. Scenario-Based Testing	5-12
3.1.9. Random Testing	5-12
3.1.10. Evidence-Based	5-12
3.1.11. Forcing Exception	5-12
3.2. Structure-Based Test Techniques	5-12
3.2.1. Control Flow Testing	5-13
3.2.2. Data Flow Testing	5-13
3.2.3. Reference Models for Structure-Based Test Techniques	5-13
3.3. Experience-Based Techniques	5-13
3.3.1. Error Guessing	5-13
3.3.2. Exploratory Testing	5-13
3.3.3. Further Experience-Based Techniques	5-14
3.4. Fault-Based and Mutation Techniques	5-14
3.5. Usage-Based Techniques	5-15
3.5.1. Operational Profile	5-15
3.5.2. User Observation Heuristics	5-15
3.6. Techniques Based on the Nature of the Application	5-15
3.7. Selecting and Combining Techniques	5-16
3.7.1. Combining Functional and Structural	5-16
3.7.2. Deterministic vs. Random	5-16
3.8. Techniques Based on Derived Knowledge	5-16
4. Test-Related Measures	5-16
4.1. Evaluation of the SUT	5-17
4.1.1. SUT Measurements that Aid in Planning and Designing Tests	5-17

4.1.2. Fault Types, Classification and Statistics	5-17
4.1.3. Fault Density	5-17
4.1.4. Life Test, Reliability Evaluation	5-17
4.1.5. Reliability Growth Models	5-17
4.2. Evaluation of the Tests Performed	5-17
4.2.1. Fault Injection	5-18
4.2.2. Mutation Score	5-18
4.2.3. Comparison and Relative Effectiveness of Different Techniques	5-18
5. Test Process	5-18
5.1. Practical Considerations	5-18
5.1.1. Attitudes/Egoless Programming	5-19
5.1.2. Test Guides and Organizational Process	5-19
5.1.3. Test Management and Dynamic Test Processes	5-19
5.1.4. Test Documentation	5-19
5.1.5. Test Team	5-19
5.1.6. Test Process Measures	5-20
5.1.7. Test Monitoring and Control	5-20
5.1.8. Test Completion	5-20
5.1.9. Test Reusability	5-20
5.2. Test Sub-Processes and Activities	5-21
5.2.1. Test Planning Process	5-21
5.2.2. Test Design and Implementation	5-21
5.2.3. Test Environment Set-up and Maintenance	5-21
5.2.4. Controlled Experiments and Test Execution	5-21
5.2.5. Test Incident Reporting	5-22
5.3. Staffing	5-22
6. Software Testing in the Development Processes and the Application Domains	5-22
6.1. Testing Inside Software Development Processes	5-23
6.1.1. Testing in Traditional Processes	5-23
6.1.2. Testing in Line with Shift-Left Movement	5-23
6.2. Testing in the Application Domains	5-24
7. Testing of and Testing Through Emerging Technologies	5-26
7.1. Testing of Emerging Technologies	5-26
7.2. Testing Through Emerging Technologies	5-27
8. Software Testing Tools	5-28
8.1. Testing Tool Support and Selection	5-29
8.2. Categories of Tools	5-29
Matrix of Topics vs. Reference Material	5-30
References	5-33

CHAPTER 06

Software Engineering Operations

6-1

Introduction	6-1
1. Software Engineering Operations Fundamentals	6-3
1.1. Definition of Software Engineering Operations	6-3
1.2. Software Engineering Operations Processes	6-4

1.3.	Software Installation	6-5
1.4.	Scripting and Automating	6-5
1.5.	Effective Testing and Troubleshooting	6-5
1.6.	Performance, Reliability and Load Balancing	6-6
2.	Software Engineering Operations Planning	6-6
2.1.	Operations Plan and Supplier Management	6-6
2.1.1.	Operations Plan	6-6
2.1.2.	Supplier Management	6-7
2.2.	Development and Operational Environments	6-7
2.3.	Software Availability, Continuity, and Service Levels	6-8
2.4.	Software Capacity Management	6-8
2.5.	Software Backup, Disaster Recovery, and Failover	6-8
2.6.	Software and Data Safety, Security, Integrity, Protection, and Controls	6-9
3.	Software Engineering Operations Delivery	6-9
3.1.	Operational Testing, Verification, and Acceptance	6-9
3.2.	Deployment/Release Engineering	6-10
3.3.	Rollback and Data Migration	6-10
3.4.	Problem Resolution	6-11
4.	Software Engineering Operations Control	6-11
4.1.	Incident Management	6-11
4.2.	Change Management	6-11
4.3.	Monitor, Measure, Track, and Review	6-11
4.4.	Operations Support	6-12
4.5.	Service Reporting	6-12
5.	Practical Considerations	6-12
5.1.	Incident and Problem Prevention	6-12
5.2.	Operational Risk Management	6-12
5.3.	Automating Software Engineering Operations	6-12
5.4.	Software Engineering Operations for Small Organizations	6-13
6.	Software Engineering Operations Tools	6-13
6.1.	Containers and Virtualization	6-13
6.2.	Deployment	6-13
6.3.	Automated Test	6-14
6.4.	Monitoring and Telemetry	6-14
	Matrix of Topics vs. Reference Material	6-14
	References	6-15

CHAPTER 07

Software Maintenance

7-1

	Introduction	7-1
1.	Software Maintenance Fundamentals	7-2
1.1.	Definitions and Terminology	7-2
1.2.	Nature of Software Maintenance	7-2
1.3.	Need for Software Maintenance	7-3
1.4.	Majority of Maintenance Costs	7-3
1.5.	Evolution of Software	7-3

1.6. Categories of Software Maintenance	7-4
2. Key Issues in Software Maintenance	7-5
2.1. Technical Issues	7-5
2.1.1 Limited Understanding	7-5
2.1.2 Testing	7-5
2.1.3 Impact Analysis	7-6
2.1.4 Maintainability	7-6
2.2. Management Issues	7-7
2.2.1. Alignment with Organizational Objectives	7-7
2.2.2. Staffing	7-7
2.2.3. Process	7-8
2.2.4. Supplier Management	7-8
2.2.5. Organizational Aspects of Maintenance	7-8
2.3. Software Maintenance Costs	7-9
2.3.1. Technical Debt Cost Estimation	7-9
2.3.2. Maintenance Cost Estimation	7-9
2.4. Software Maintenance Measurement	7-10
3. Software Maintenance Processes	7-11
3.1. Software Maintenance Processes	7-11
3.2. Software Maintenance Activities and Tasks	7-11
3.2.1. Supporting and Monitoring Activities	7-12
3.2.2. Planning Activities	7-12
3.2.3. Configuration Management	7-13
3.2.4. Software Quality	7-13
4. Software Maintenance Techniques	7-13
4.1. Program Comprehension	7-13
4.2. Software Reengineering	7-13
4.3. Reverse Engineering	7-14
4.4. Continuous Integration, Delivery, Testing, and Deployment	7-14
4.5. Visualizing Maintenance	7-15
5. Software Maintenance Tools	7-15
Matrix of Topics vs. Reference Material	7-16
Further Readings	7-17
References	7-17

CHAPTER 08

Software Configuration Management

8-1

Introduction	8-1
1. Management of the SCM Process	8-2
1.1. Organizational Context for SCM	8-2
1.2. Constraints and Guidance for the SCM Process	8-3
1.3. Planning for SCM	8-3
1.3.1. SCM Organization and Responsibilities	8-4
1.3.2. SCM Resources and Schedules	8-4
1.3.3. Tool Selection and Implementation	8-4
1.3.4. Vendor/Subcontractor Control	8-5

1.3.5. Interface Control	8-5
1.4. SCM Plan	8-5
1.5. Monitoring of Software Configuration Management	8-5
1.5.1 SCM Measures and Measurement	8-6
1.5.2 In-Process Audits of SCM	8-6
2. Software Configuration Identification	8-6
2.1. Identifying Items to Be Controlled	8-6
2.1.1 Software Configuration	8-6
2.1.2 Software Configuration Item	8-6
2.2. Configuration Item Identifiers and Attributes	8-7
2.3. Baseline Identification	8-7
2.4. Baseline Attributes	8-7
2.5. Relationships Scheme Definition	8-7
2.6. Software Libraries	8-8
3. Software Configuration Change Control	8-9
3.1. Requesting, Evaluating, and Approving Software Changes	8-9
3.1.1 Software Configuration Control Board	8-10
3.1.2 Software Change Request Process	8-10
3.1.3 Software Change Request Forms Definition	8-10
3.2. Implementing Software Changes	8-10
3.3. Deviations and Waivers	8-11
4. Software Configuration Status Accounting	8-11
4.1. Software Configuration Status Information	8-11
4.2. Software Configuration Status Reporting	8-11
5. Software Configuration Auditing	8-12
5.1. Software Functional Configuration Audit	8-12
5.2. Software Physical Configuration Audit	8-12
5.3. In-Process Audits of a Software Baseline	8-12
6. Software Release Management and Delivery	8-13
6.1. Software Building	8-13
6.2. Software Release Management	8-13
7. Software Configuration Management Tools	8-14
Matrix of Topics vs. Reference Material	8-15
Further Readings	8-16
References	8-17

CHAPTER 09

Software Engineering Management 9-1

Introduction	9-1
1. Initiation and Scope Definition	9-6
1.1. Determination and Negotiation of Requirements	9-6
1.2. Feasibility Analysis	9-6
1.3. Process for the Review and Revision of Requirements	9-7
2. Software Project Planning	9-7
2.1. Process Planning	9-8
2.2. Determine Deliverables	9-8

2.3.	Effort, Schedule, and Cost Estimation	9-8
2.4.	Resource Allocation	9-9
2.5.	Risk Management	9-9
2.6.	Quality Management	9-9
2.7.	Plan Management	9-10
3.	Software Project Execution	9-11
3.1.	Implementation of Plans	9-11
3.2.	Software Acquisition and Supplier Contract Management	9-11
3.3.	Implementation of Measurement Process	9-12
3.4.	Monitor Process	9-12
3.5.	Control Process	9-12
3.6.	Reporting	9-13
4.	Software Review and Evaluation	9-13
4.1.	Determining Satisfaction of Requirements	9-13
4.2.	Reviewing and Evaluating Performance	9-13
5.	Closure	9-13
5.1.	Determining Closure	9-13
5.2.	Closure Activities	9-14
6.	Software Engineering Measurement	9-14
6.1.	Establish and Sustain Measurement Commitment	9-14
6.2.	Plan the Measurement Process	9-15
6.3.	Perform the Measurement Process	9-15
6.4.	Evaluate Measurement	9-16
7.	Software Engineering Management Tools	9-16
	Matrix of Topics vs. Reference Material	9-17
	Further Readings	9-18
	References	9-18

CHAPTER 10

Software Engineering Process

10-1

Introduction	10-1
1. Software Engineering Process Fundamentals	10-1
1.1. Introduction	10-1
1.2. Software Engineering Process Definition	10-3
2. Life Cycles	10-3
2.1. Life Cycle Definition, Process Categories, and Terminology	10-3
2.2. Rationale for Life Cycles	10-4
2.3. The Concepts of Process Models and Life Cycle Models	10-5
2.4. Some Paradigms for Development Life Cycle Models	10-5
2.5. Development Life Cycle Models and Their Engineering Dimension	10-6
2.6. The Management of SLCPs	10-7
2.7. Software Engineering Process Management	10-8
2.8. Software Life Cycle Adaptation	10-8
2.9. Practical Considerations	10-8
2.10. Software Process Infrastructure, Tools, Methods	10-9
2.11. Software Engineering Process Monitoring and	

its Relationship with the Software Product	10-9
3. Software Process Assessment and Improvement	10-9
3.1. Overview of Software Process Assessment and Improvement	10-9
3.2. Goal-Question-Metric (GQM)	10-10
3.3. Framework-Based Methods	10-10
3.4. Process Assessment and Improvement in Agile	10-10
Matrix of Topics vs. Reference Material	10-10
References	10-11

CHAPTER 11

Software Engineering Models and Methods 11-1

Introduction	11-1
1. Modeling	11-1
1.1. Modeling Principles	11-2
1.2. Properties and Expression of Models	11-3
1.3. Syntax, Semantics, and Pragmatics	11-3
1.4. Preconditions, Postconditions, and Invariants	11-4
2. Types of Models	11-4
2.1. Structural Modeling	11-5
2.2. Behavioral Modeling	11-5
3. Analysis of Models	11-5
3.1. Analyzing for Completeness	11-6
3.2. Analyzing for Consistency	11-6
3.3. Analyzing for Correctness	11-6
3.4. Analyzing for Traceability	11-6
3.5. Analyzing for Interaction	11-6
4. Software Engineering Methods	11-7
4.1. Heuristic Methods	11-7
4.2. Formal Methods	11-8
4.3. Prototyping Methods	11-9
4.4. Agile Methods	11-9
Matrix of Topics vs. Reference Material	11-11
References	11-12

CHAPTER 12

Software Quality 12-1

Introduction	12-1
1. Software Quality Fundamentals	12-3
1.1. Software Engineering Culture and Ethics	12-3
1.2. Value and Costs of Quality	12-4
1.3. Standards, Models, and Certifications	12-4
1.4. Software Dependability and Integrity Levels	12-5
1.4.1 Dependability	12-5
1.4.2. Integrity Levels of Software	12-6

2.	Software Quality Management Process	12-6
2.1.	Software Quality Improvement	12-7
2.2.	Plan Quality Management	12-7
2.3.	Evaluate Quality Management	12-8
2.3.1	Software Quality Measurement	12-8
2.4.	Perform Corrective and Preventive Actions	12-9
2.4.1.	Defect Characterization	12-9
3.	Software Quality Assurance Process	12-9
3.1.	Prepare for Quality Assurance	12-9
3.2.	Perform Process Assurance	12-10
3.3.	Perform Product Assurance	12-11
3.4.	V&V and Testing	12-12
3.4.1	Static Analysis Techniques	12-12
3.4.2.	Dynamic Analysis Techniques	12-13
3.4.3.	Formal Analysis Techniques	12-13
3.4.4.	Software Quality Control and Testing	12-13
3.4.5.	Technical Reviews and Audits	12-13
4.	Software Quality Tools	12-14
	Matrix of Topics vs. Reference Material	12-15
	Further Readings	12-16
	References	12-17

CHAPTER 13

Software Security

13-1

	Introduction	13-1
1.	Software Security Fundamentals	13-1
1.1.	Software Security	13-1
1.2.	Information Security	13-1
1.3.	Cybersecurity	13-2
2.	Security Management and Organization	13-2
2.1.	Capability Maturity Model	13-2
2.2.	Information Security Management System	13-2
2.3.	Agile Practice for Software Security	13-3
3.	Software Security Engineering and Processes	13-3
3.1.	Security Engineering and Secure Development Life Cycle (SDLC)	13-3
3.2.	Common Criteria for Information Technology Security Evaluation	13-3
4.	Security Engineering for Software Systems	13-3
4.1.	Security Requirements	13-3
4.2.	Security Design	13-4
4.3.	Security Patterns	13-4
4.4.	Construction for Security	13-4
4.5.	Security Testing	13-5
4.6.	Vulnerability Management	13-5
5.	Software Security Tools	13-5
5.1.	Security Vulnerability Checking Tools	13-5
5.2.	Penetration Testing Tools	13-6

6. Domain-Specific Software Security	13-6
6.1. Security for Container and Cloud	13-6
6.2. Security for IoT Software	13-6
6.3. Security for Machine Learning-Based Application	13-6
Matrix of Topics vs. Reference Material	13-7
Further Readings	13-7
References	13-8

CHAPTER 14

Software Engineering Professional Practice 14-1

Introduction	14-1
1. Professionalism	14-2
1.1. Accreditation, Certification and Qualification, and Licensing	14-2
1.1.1. Accreditation	14-2
1.1.2. Certification and Qualification	14-3
1.1.3. Licensing	14-3
1.2. Codes of Ethics and Professional Conduct	14-3
1.3. Nature and Role of Professional Societies	14-4
1.4. Nature and Role of Software Engineering Standards	14-4
1.5. Economic Impact of Software	14-5
1.6. Employment Contracts	14-5
1.7. Legal Issues	14-6
1.7.1. Standards	14-6
1.7.2. Trademarks	14-6
1.7.3. Patents	14-6
1.7.4. Copyrights	14-6
1.7.5. Trade Secrets	14-6
1.7.6. Professional Liability	14-7
1.7.7. Legal Requirements	14-7
1.7.8. Trade Compliance	14-7
1.7.9. Cybercrime	14-7
1.7.10. Data Privacy	14-8
1.8. Documentation	14-8
1.9. Trade-Off Analysis	14-9
2. Group Dynamics and Psychology	14-9
2.1. Dynamics of Working in Teams/Groups	14-9
2.2. Individual Cognition	14-10
2.3. Dealing with Problem Complexity	14-10
2.4. Interacting with Stakeholders	14-10
2.5. Dealing with Uncertainty and Ambiguity	14-11
2.6. Dealing with Equity, Diversity, and Inclusivity	14-11
3. Communication Skills	14-11
3.1. Reading, Understanding, and Summarizing	14-12
3.2. Writing	14-12
3.3. Team and Group Communication	14-12
3.4. Presentation Skills	14-12

Matrix of Topics vs. Reference Material	14-13
Further Readings	14-14
References	14-14

CHAPTER 15

Software Engineering Economics 15-1

Introduction	15-1
1. Software Engineering Economics Fundamentals	15-3
1.1. Proposals	15-3
1.2. Cash Flow	15-3
1.3. Time-Value of Money	15-3
1.4. Equivalence	15-4
1.5. Bases for Comparison	15-4
1.6. Alternatives	15-4
1.7. Intangible Assets	15-4
1.8. Business Model	15-5
2. The Engineering Decision-Making Process	15-5
2.1. Process Overview	15-5
2.2. Understand the Real Problem	15-5
2.3. Identify All Reasonable Technically Feasible Solutions	15-6
2.4. Define the Selection Criteria	15-6
2.5. Evaluate Each Alternative Against the Selection Criteria	15-6
2.6. Select the Preferred Alternative	15-6
2.7. Monitor the Performance of the Selected Alternative	15-7
3. For-Profit Decision-Making	15-7
3.1. Minimum Acceptable Rate of Return	15-7
3.2. Economic Life	15-7
3.3. Planning Horizon	15-8
3.4. Replacement Decisions	15-8
3.5. Retirement Decisions	15-9
3.6. Advanced For-Profit Decision Considerations	15-9
4. Nonprofit Decision-Making	15-9
4.1. Benefit-Cost Analysis	15-9
4.2. Cost-Effectiveness Analysis	15-9
5. Present Economy Decision-Making	15-9
5.1. Break-Even Analysis	15-9
5.2. Optimization Analysis	15-9
6. Multiple-Attribute Decision-Making	15-10
6.1. Compensatory Techniques	15-10
6.2. Non-Compensatory Techniques	15-10
7. Identifying and Characterizing Intangible Assets	15-10
7.1. Identify Processes and Define Business Goals	15-10
7.2. Identify Intangible Assets Linked with Business Goal	15-11
7.3. Identify Software Products That Support Intangible Assets	15-11
7.4. Define and Measure Indicators	15-11
7.5. Intangible Asset Characterization	15-11

7.6. Link Specific Intangible Assets with the Business Model	15-13
7.7. Decision-Making	15-13
8. Estimation	15-13
8.1. Expert Judgment	15-14
8.2. Analogy	15-15
8.3. Decomposition	15-15
8.4. Parametric	15-15
8.5. Multiple Estimates	15-15
9. Practical Considerations	15-16
9.1. Business Case	15-16
9.2. Multiple-Currency Analysis	15-16
9.3. Systems Thinking	15-16
10. Related Concepts	15-16
10.1. Accounting	15-16
10.2. Cost and Costing	15-16
10.3. Finance	15-17
10.4. Controlling	15-17
10.5. Efficiency and Effectiveness	15-17
10.6. Productivity	15-18
10.7. Product or Service	15-18
10.8. Project	15-18
10.9. Program	15-18
10.10. Portfolio	15-18
10.11. Product Life Cycle	15-19
10.12. Project Life Cycle	15-19
10.13. Price and Pricing	15-19
10.14. Prioritization	15-19
Matrix of Topics vs. Reference Material	15-20
Further Readings	15-22
References	15-22

CHAPTER 16

Computing Foundations 16-1

Introduction	16-2
1. Basic Concepts of a System or Solution	16-2
2. Computer Architecture and Organization	16-3
2.1. Computer Architecture	16-3
2.2. Types of Computer Architectures	16-3
2.2.1. Von Neumann Architecture	16-3
2.2.2. Harvard Architecture	16-4
2.2.3. Instruction Set Architecture	16-4
2.2.4. Flynn's Architecture or Taxonomy	16-5
2.2.5. System Architecture	16-5
2.3. Microarchitecture or Computer Organization	16-5
2.3.1. Arithmetic Logic Unit	16-5
2.3.2. Memory Unit	16-6

2.3.3. Input/Output Devices	16-6
2.3.4. Control Unit	16-6
3. Data Structures and Algorithms	16-6
3.1. Types of Data Structures	16-6
3.2. Operations on Data Structures	16-7
3.3. Algorithms and Attributes of Algorithms	16-7
3.4. Algorithm Complexity	16-8
3.5. Measurement of Complexity	16-8
3.6. Designing Algorithms	16-8
3.7. Sorting Techniques	16-9
3.8. Searching Techniques	16-10
3.9. Hashing	16-10
4. Programming Fundamentals and Languages	16-10
4.1. Programming Language Types	16-10
4.2. Programming Syntax, Semantics, Type Systems	16-11
4.3. Subprograms and Coroutines	16-11
4.4. Object-Oriented Programming	16-12
4.5. Distributed Programming and Parallel Programming	16-13
4.6. Debugging	16-13
4.7. Standards and Guidelines	16-13
5. Operating Systems	16-15
5.1. Processor Management	16-15
5.2. Memory Management	16-16
5.3. Device Management	16-16
5.4. Information Management	16-16
5.5. Network Management	16-16
6. Database Management	16-17
6.1. Schema	16-17
6.2. Data Models and Storage Models	16-17
6.3. Database Management Systems	16-18
6.4. Relational Database Management Systems and Normalization	16-18
6.5. Structured Query Language	16-19
6.6. Data Mining and Data Warehousing	16-19
6.7. Database Backup and Recovery	16-20
7. Computer Networks and Communications	16-20
7.1. Types of Computer Networks	16-20
7.2. Layered Architectures of Networks	16-21
7.3. Open Systems Interconnection Model	16-21
7.4. Encapsulation and Decapsulation	16-22
7.5. Application Layer Protocols	16-22
7.6. Design Techniques for Reliable and Efficient Network	16-22
7.7. Internet Protocol Suite	16-23
7.8. Wireless and Mobile Networks	16-23
7.9. Security and Vulnerabilities	16-23
8. User and Developer Human Factors	16-24
8.1. User Human Factors	16-24
8.2. Developer Human Factors	16-24
9. Artificial Intelligence and Machine Learning	16-25

9.1. Reasoning	16-25
9.2. Learning	16-26
9.3. Models	16-26
9.4. Perception and Problem-Solving	16-27
9.5. Natural Language Processing	16-27
9.6. AI and Software Engineering	16-27
Matrix of Topics vs. Reference Material	16-28
References	16-32

CHAPTER 17

Mathematical Foundations 17-1

Introduction	17-1
1. Basic Logic	17-1
1.1. Propositional Logic	17-1
1.2. Predicate Logic	17-3
2. Proof Techniques	17-3
2.1. Direct Proof	17-4
2.2. Proof by Contradiction	17-4
2.3. Proof by Induction	17-4
2.4. Proof by Example	17-5
3. Set, Relation, Function	17-5
3.1. Set Operations	17-6
3.2. Properties of Sets	17-6
3.3. Relation and Function	17-7
4. Graph and Tree	17-8
4.1. Graph	17-8
4.2. Tree	17-10
5. Finite-State Machine	17-12
6. Grammar	17-13
6.1. Language Recognition	17-14
7. Number Theory	17-14
7.1. Types of Numbers	17-15
7.2. Divisibility	17-15
7.3. Prime Number	17-15
7.4. Greatest Common Divisor	17-16
8. Basics of Counting	17-16
9. Discrete Probability	17-17
10. Numerical Precision, Accuracy, and Error	17-18
11. Algebraic Structures	17-19
11.1. Group	17-19
11.2. Ring	17-20
12. Engineering Calculus	17-21
13. New Advancements	17-21
13.1. Computational Neurosciences	17-21
13.2. Genomics	17-21
Matrix of Topics vs. Reference Material	17-22
References	17-22

CHAPTER 18**Engineering Foundations****18-1**

Introduction	18-1
1. The Engineering Process	18-1
2. Engineering Design	18-2
2.1. Engineering Design in Engineering Education	18-2
2.2. Design as a Problem-Solving Activity	18-3
3. Abstraction and Encapsulation	18-3
3.1. Levels of Abstraction	18-4
3.2. Encapsulation	18-4
3.3. Hierarchy	18-4
3.4. Alternate Abstractions	18-4
4. Empirical Methods and Experimental Techniques	18-4
4.1. Designed Experiment	18-5
4.2. Observational Study	18-5
4.3. Retrospective Study	18-5
5. Statistical Analysis	18-5
5.1. Unit of Analysis (Sampling Units), Population, and Sample	18-5
5.2. Correlation and Regression	18-8
6. Modeling, Simulation, and Prototyping	18-8
6.1. Modeling	18-8
6.2. Simulation	18-9
6.3. Prototyping	18-9
7. Measurement	18-10
7.1. Levels (Scales) of Measurement	18-10
7.2. Implications of Measurement Theory for Programming Languages	18-12
7.3. Direct and Derived Measures	18-13
7.4. Reliability and Validity	18-14
7.5. Assessing Reliability	18-14
7.6. Goal-Question-Metric Paradigm: Why Measure?	18-15
8. Standards	18-15
9. Root Cause Analysis	18-16
9.1. Root Cause Analysis Techniques	18-16
9.2. Root Cause–Based Improvement	18-17
10. Industry 4.0 and Software Engineering	18-17
Matrix of Topics vs. Reference Material	18-18
Further Readings	18-19
References	18-20

APPENDIX A**Knowledge Area Description Specifications****A-1**

Introduction	A-1
The Swebok Guide is a Foundational Document for the IEEE Computer Society	
Suite of Software Engineering Products	A-1
Baseline and Change Control	A-1

Criteria and Requirements for the Breakdown of Topics Within a Knowledge Area	A-2
Criteria and Requirements for Describing Topics	A-2
Criteria and Requirements for Reference Material	A-2
Common Structure	A-4
What Do We Mean by “Generally Recognized Knowledge”?	A-4
Length of KA Description	A-5
Important Related Documents	A-5
Other Detailed Guidelines	A-6
Editing	A-6
Release of Copyright	A-6
References	A-6

APPENDIX B

IEEE and ISO/IEC Standards Supporting the Software Engineering Body of Knowledge (SWEBOK) B-1

1. Overview	B-1
1.1. The SWEBOK and standards	B-1
1.2. Types of Standards	B-2
1.3. Sources of Software Engineering Standards	B-2
2. The software engineering standards landscape	B-3
3. Life cycle process standards	B-4
4. Extensions and specialized applications of ISO/IEC/IEEE 12207	B-5
4.1. Explanations of concepts and several processes	B-5
4.2. More specialized extensions	B-8
4.3. SoS standards	B-9
5. Single Process Standards	B-9
6. Standards for product line, methods, and tools	B-9
7. Process assessment standards	B-10
8. Professional Skills and Knowledge Standards	B-11
9. Selected Software Engineering Standards	B-11

APPENDIX C

Consolidated Reference List C-1

Consolidated Reference List	C-1
-----------------------------	-----