

Contents

Part 1 Introduction

Chapter 1	Introduction
1.1	Software products
1.2	The software process
1.3	Boehm's spiral model
1.4	Process visibility
1.5	Professional responsibility

All covered in 6/e but split between Chapter 1 (SW Engineering FAQs) and Chapter 3 (Software Processes)

Chapter 2	Computer-based System Engineering
2.1	Systems and their environment
2.2	System procurement
2.3	The system engineering process
2.4	System architecture modelling
2.5	Human factors
2.6	System reliability engineering

More or less unchanged in 6th edition

Chapter 3	Project Management
3.1	Management activities
3.2	Project planning
3.3	Activity organisation
3.4	Project scheduling

All material in 6th edition plus additional material on risk management has been added. Now Ch. 4

Part 2 Requirements and Specification

Chapter 4	Requirements Engineering
4.1	The requirements engineering process
4.2	The software requirements document
4.3	Requirements validation
4.4	Requirements evolution

Chapter 5	Requirements Analysis
5.1	Viewpoint-oriented analysis
5.2	Method-based analysis
5.3	System contexts
5.4	Social and organisational factors

All included in the 6th edition but organised in a different and (I hope) more coherent way with Ch. 5 focusing on the requirements themselves and the requirements document (incl discussion of IEEE standard) and with Ch. 6 focusing on requirements engineering processes. An intro to RE processes is in Ch. 3.

Chapter 6	System Models
6.1	Data-flow models
6.2	Semantic data models
6.3	Object models
6.4	Data dictionaries

Topics covered in Ch. 7 6/e but in the UML and with additional material on event models. Probably a bit less on data-flow

Chapter 7	Requirements Definition and Specification	
7.1	Requirements definition	Material split between Ch. 5 and Ch. 6 in the 6/e
7.2	Requirements specification	
7.3	Non-functional requirements	
Chapter 8	Software Prototyping	
8.1	Prototyping in the software process	Virtually unchanged in 6/e
8.2	Prototyping techniques	
8.3	User interface prototyping	
Chapter 9	Formal Specification	
9.1	Formal specification on trial	One of the major 6/e changes. These 3 chapters have been summarised in a single chapter in the 6/e so there is less on algebraic and model-based specification. Examples are from critical systems as this seems to be the only area where these techniques are likely to be viable.
9.2	Transformational development	
9.3	Specifying functional abstractions	
Chapter 10	Algebraic Specification	
10.1	Systematic algebraic specification	One of the major 6/e changes. These 3 chapters have been summarised in a single chapter in the 6/e so there is less on algebraic and model-based specification. Examples are from critical systems as this seems to be the only area where these techniques are likely to be viable.
10.2	Structured specification	
10.3	Error specification	
Chapter 11	Model-based Specification	
11.1	Z schemas	One of the major 6/e changes. These 3 chapters have been summarised in a single chapter in the 6/e so there is less on algebraic and model-based specification. Examples are from critical systems as this seems to be the only area where these techniques are likely to be viable.
11.2	The Z specification process	
11.3	Specifying ordered collections	
Part 3 Software Design		
Chapter 12	Software Design	
12.1	The design process	Some material in this chapter now covered in Chapter 3 (SW processes); Other material e.g. on design metrics has gone
12.2	Design strategies	
12.3	Design quality	
Chapter 13	Architectural Design	
13.1	System structuring	All material in corresponding chapter in 6/e (Ch. 10)
13.2	Control models	
13.3	Modular decomposition	
13.4	Domain-specific architectures	
Chapter 14	Object-oriented Design	
14.1	Objects, object classes and inheritance	All covered in Ch. 12 with a UML flavour. I've taken a more process-based approach to the description
14.2	Object identification	
14.3	An object-oriented design example	
14.4	Concurrent objects	
Chapter 15	Function-oriented Design	
15.1	Data-flow design	Covered but in less detail in Ch. 26. Legacy systems (I know that its still used for new systems too but OO design dominates in university courses)
15.2	Structural decomposition	
15.3	Detailed design	

- 15.4 A comparison of design strategies
- Chapter 16 Real-time Systems Design**
- 16.1 System design
- 16.2 State machine modelling
- 16.3 Real-time executives
- 16.4 Monitoring and control systems
- 16.5 Data acquisition systems

Covered in Ch 13 more or less unchanged except I've used Java instead of Ada

- Chapter 17 User Interface Design**
- 7.1 Design principles
- 17.2 User-system interaction
- 17.3 Information presentation
- 17.4 User guidance
- 17.5 Interface evaluation

All covered in Ch. 15

Part 4 Dependable Systems

- Chapter 18 Software Reliability**
- 18.1 Software reliability metrics
- 18.2 Software reliability specification
- 18.3 Statistical testing
- 18.4 Reliability growth modelling

Coverage split between Ch 16 (Dependability), Ch 17 (Critical Systems Spec.), Ch. 21 (Crit Sys Validation)

- Chapter 19 Programming for Reliability**
- 19.1 Fault avoidance
- 19.2 Fault tolerance
- 19.3 Exception handling
- 19.4 Defensive programming

All covered in Ch. 18 (Critical systems development)

- Chapter 20 Software Reuse**
- 20.1 Software development with reuse
- 20.2 Software development for reuse
- 20.3 Generator-based reuse
- 20.4 Application system portability

The new chapter on reuse is completely different and covers component based development, application families and patterns. Portability stuff has gone.

- Chapter 21 Safety-critical Software**
- 21.1 An insulin delivery system
- 21.2 Safety specification
- 21.3 Safety assurance

Coverage split between Ch 16 (Dependability), Ch 17 (Critical Systems Spec.), Ch. 21 (Crit Sys Validation)

Part 5 Verification and Validation

- Chapter 22 Verification and Validation**
- 22.1 The testing process
- 22.2 Test planning
- 22.3 Testing strategies

Three V & V chapters have been cut to 2 plus additional material in Ch. 3 on the testing process. All 5/e material still covered but summarised in places. More on object-oriented testing.

Chapter 23 Defect Testing

- 23.1 Black-box testing
- 23.2 Structural testing
- 23.3 Interface testing

Covered in Ch. 20

Chapter 24 Static Verification

- 24.1 Program inspections
- 24.2 Mathematically-based verification
- 24.3 Static analysis tools
- 24.4 Cleanroom software development

Covered in Ch. 19 except for mathematically-based verification which has been cut

Part 6 CASE**Chapter 25 Computer-aided Software Engineering**

- 25.1 CASE classification
- 25.2 Integrated CASE
- 25.3 The CASE life cycle

Chapter 26 CASE Workbenches

- 26.1 Programming workbenches
- 26.2 Analysis and design workbenches
- 26.3 Testing workbenches
- 26.4 Meta-CASE workbenches

Chapter 27 Software Engineering Environments

- 27.1 Integrated environments
- 27.2 Platform services
- 27.3 Framework services
- 27.4 PCTE

Another major change. More or less all gone. CASE classification is covered in Ch 3 under process support, and design workbenches in Ch. 7 (System modeling) and Testing workbenches in Ch. 20 (Testing). The rest I have cut.

Part 7 Management**Chapter 28 Managing People**

- 28.1 Cognitive fundamentals
- 28.2 Management implications
- 28.3 Project staffing
- 28.4 Group working
- 28.5 Working environments

All covered in Ch. 22. New material on people capability maturity model

Chapter 29 Software Cost Estimation

- 29.1 Productivity
- 29.2 Estimation techniques
- 29.3 Algorithmic cost modelling
- 29.4 Project duration and staffing

All covered in Ch. 23 but updated to COCOMO 2

Chapter 30	Quality Management	Virtually unchanged in content in Ch. 24 but some reordering of presentation.
30.1	Process quality assurance	
30.2	Quality reviews	
30.3	Software standards	
30.4	Documentation standards	
30.6	Product quality metrics	

Chapter 31	Process Improvement	Virtually unchanged - Ch. 25
31.1	Process and product quality	
31.2	Process analysis and modelling	
31.3	Process measurement	
31.4	The SEI process maturity model	
31.5	Process classification	

Part 8 Evolution

Chapter 32	Software Maintenance	Covered in Ch. 27, Software Change. New material in Ch. 27 on architectural evolution.
32.1	The maintenance process	
32.2	System documentation	
32.3	Program evolution dynamics	
32.4	Maintenance costs	
32.5	Maintainability measurement	

Chapter 33	Configuration Management	All covered in Ch. 29
33.1	Configuration management planning	
33.2	Change management	
33.3	Version and release management	
33.4	System building	

Chapter 34	Software Re-engineering	All covered in Ch. 28
34.1	Source code translation	
34.2	Program re-structuring	
34.3	Data re-engineering	
34.4	Reverse engineering	

References

Index

Material covered in 6/e that isn't in the 5/e

Distributed systems architectures (Client-server, distributed object), CORBA
 Component based software engineering, application families, patterns
 Risk management
 Dependability (in general)
 Security, security specification and security validation (not in any detail)
 Legacy systems
 IEEE/ACM Code of Ethics