
Undergraduate Topics in Computer Science

Series Editor

Ian Mackie, University of Sussex, Brighton, UK

Advisory Editors

Samson Abramsky , Department of Computer Science, University of Oxford, Oxford, UK

Chris Hankin , Department of Computing, Imperial College London, London, UK

Mike Hinchey , Lero – The Irish Software Research Centre, University of Limerick, Limerick, Ireland

Joseph Migga Kizza, The University of Tennessee–Chattanooga, College of Engineering and Computer Science, Chattanooga, Tennessee, USA

Dexter C. Kozen, Department of Computer Science, Cornell University, Ithaca, NY, USA

Andrew Pitts , Department of Computer Science and Technology, University of Cambridge, Cambridge, UK

Hanne Riis Nielson , Department of Applied Mathematics and Computer Science, Technical University of Denmark, Kongens Lyngby, Denmark

Steven S. Skiena, Department of Computer Science, Stony Brook University, Stony Brook, NY, USA

Iain Stewart , Department of Computer Science, Durham University, Durham, UK

‘Undergraduate Topics in Computer Science’ (UTiCS) delivers high-quality instructional content for undergraduates studying in all areas of computing and information science. From core foundational and theoretical material to final-year topics and applications, UTiCS books take a fresh, concise, and modern approach and are ideal for self-study or for a one- or two-semester course. The texts are all authored by established experts in their fields, reviewed by an international advisory board, and contain numerous examples and problems, many of which include fully worked solutions.

The UTiCS concept relies on high-quality, concise books in softback format, and generally a maximum of 275–300 pages. For undergraduate textbooks that are likely to be longer, more expository, Springer continues to offer the highly regarded *Texts in Computer Science* series, to which we refer potential authors.

Gerard O'Regan

Concise Guide to Software Engineering

From Fundamentals to Application Methods

Second Edition



Springer

Gerard O'Regan
University of Central Asia
Naryn, Kyrgyzstan

ISSN 1863-7310 ISSN 2197-1781 (electronic)
Undergraduate Topics in Computer Science
ISBN 978-3-031-07815-6 ISBN 978-3-031-07816-3 (eBook)
<https://doi.org/10.1007/978-3-031-07816-3>

1st edition: © Springer International Publishing AG 2017
2nd edition: © Springer Nature Switzerland AG 2022

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To

*Past and present members of the Formal
Methods Group (Foundations and Methods
Group) at Trinity College Dublin, Ireland.*

Preface

Overview

The objective of this book is to provide a concise introduction to the software engineering field to students and practitioners. The principles of software engineering are discussed, and the goal is to give the reader a grasp of the fundamentals of the software engineering field, as well as guidance on how to apply the theory in an industrial environment.

Organization and Features

Chapter 1 presents a broad overview of software engineering and discusses various software lifecycles and the activities in software development. We discuss requirements gathering and specification, software design, implementation, testing and maintenance. The lightweight Agile methodology is introduced, and it has become very popular in industry.

Chapter 2 discusses the professional responsibilities of software engineers. Engineers have a responsibility to ensure that the products that they design and develop are built to the highest possible standards and are safe for the public to use. Engineers must behave ethically in their dealings with their clients, and they need to adhere to the code of ethics of the professional engineering body.

Chapter 3 discusses ethical software engineering where the ethical software engineer needs to examine both the technical and the ethical dimensions of decisions that affect wider society. We discuss the Volkswagen emissions scandal where engineers installed a “defeat device” to enable cars to pass an emissions test.

Chapter 4 introduces project management for traditional software engineering, and we discuss project estimation, project planning and scheduling, project monitoring and control, risk management, managing communication and change, and managing project quality.

Chapter 5 discusses requirements engineering and discusses activities such as requirements gathering, requirements elicitation, requirements analysis, requirements management, and requirements verification and validation.

Chapter 6 discusses software design and development, where software design is the blueprint of the solution to be developed. It is concerned with the high-level architecture of the system, as well as the detailed design that describes the algorithms and functionality of the individual programs. The detailed design is then implemented in a programming language such as C++ or Java. We discuss software development topics such as software reuse, customized-off-the-shelf software (COTS), and open-source software development.

Chapter 7 discusses software inspections, which play an important role in building quality into a product. The well-known Fagan inspection process that was developed at IBM in the 1970s is discussed, as well as lighter review and walk-through methodologies.

Chapter 8 is concerned with software testing and discusses the various types of testing that may be carried out during the project. We discuss test planning, test case definition, test environment set-up, test execution, test tracking, test metrics, test reporting, and testing in an e-commerce environment.

Chapter 9 discusses ethics and privacy where professional ethics are a code of conduct that governs how members of a profession deal with each other and with third parties. It expresses ideals of human behaviour, and the fundamental values of the organization, and is an indication of its professionalism. Privacy is defined as “the right to be left alone,” and specifies there should be no intrusion upon seclusion, and no public disclosure of private facts or false information.

Chapter 10 is concerned with metrics and problem-solving, and this includes a discussion of the balanced score card which assists in identifying appropriate metrics for the organization. The goal, question, metrics (GQM) approach is discussed, and this allows appropriate metrics related to the organization goals to be defined. A selection of sample metrics for an organization is presented, and problem-solving tools such as fishbone diagrams, Pareto charts, trend charts are discussed.

Chapter 11 is concerned with the selection and management of a software supplier. It discusses how candidate suppliers may be identified, formally evaluated against defined selection criteria, and how the appropriate supplier is selected. We discuss how the selected supplier is managed during the project.

Chapter 12 discusses software configuration management and discusses the fundamental concept of a baseline. Configuration management is concerned with identifying those deliverables that must be subject to change control and controlling changes to them.

Chapter 13 discusses software quality assurance and the importance of process quality. It is a premise in the quality field that good processes and conformance to them is essential for the delivery of high-quality product, and this chapter discusses audits, and describes how they are carried out.

Chapter 14 discusses the Agile methodology which is a popular lightweight approach to software development. Agile provides opportunities to assess the direction of a project throughout the development lifecycle and ongoing changes to requirements are considered normal in the Agile world. It has a strong collaborative style of working, and it advocates adaptive planning and evolutionary development.

Chapter 15 discusses software reliability and dependability and covers topics such as software reliability and software reliability models; the cleanroom methodology, system availability; safety and security critical systems; and dependability engineering.

Chapter 16 discusses formal methods, which consist of a set of mathematical techniques to specify and derive a program from its specification. Formal methods may be employed to rigorously state the requirements of the proposed system. They may be employed to derive a program from its mathematical specification, and they may be used to provide a rigorous proof that the implemented program satisfies its specification. They have been mainly applied to the safety critical field.

Chapter 17 presents the Z specification language, which is one of the more popular formal methods. It was developed at the Programming Research Group at Oxford University in the early 1980s. Z specifications are mathematical, and the use of mathematics ensures precision and allows inconsistencies and gaps in the specification to be identified. Theorem provers may be employed to demonstrate that the software implementation meets its specification.

Chapter 18 presents the unified modelling language (UML), which is a visual modelling language for software systems, and I used to present several views of the system architecture. It was developed at Rational Corporation as a notation for modelling object-oriented systems. We present various UML diagrams such as use case diagrams, sequence diagrams, and activity diagrams.

Chapter 19 discusses software process improvement. It begins with a discussion of a software process and discusses the benefits that may be gained from a software process improvement initiative. Various models that support software process improvement are discussed, and these include the Capability Maturity Model Integration (CMMI), ISO 9000, Personal Software Process (PSP), and Team Software Process (TSP).

Chapter 20 gives an overview of the CMMI model and discusses its five maturity levels and their constituent process areas. We discuss both the staged and continuous representations of the CMMI and SCAMPI appraisals that indicate the extent to which the CMMI has been implemented in the organization, as well as identifying opportunities for improvement.

Chapter 21 discusses various tools to support the various software engineering activities. The focus is first to define the process, and then to find tools to support the process. Tools to support project management are discussed as well as tools to support requirements engineering, configuration management, design and development activities, and software testing.

Chapter 22 discusses innovation in the software field including miscellaneous topics such as distributed systems, service-oriented architecture, software as a service, cloud computing and embedded systems. We discuss the need for innovation in software engineering and discuss some recent innovations such as aspect-oriented software engineering.

Chapter 23 is concerned with the application of the legal system to the computing field. This includes the protection of intellectual property such as patents, copyright, trademarks and trade secrets, and the resolution of disputes between parties.

Chapter 24 discusses cybersecurity and cybercrime. Cybercrime is a crime that involves a computer and a network. The computer may be the vehicle by which the crime was conducted, or it may be the target of the crime. Cybersecurity is concerned with the ability of a computer system to protect itself from attacks, and there are several characteristics of security such as confidentiality, integrity, and availability.

Chapter 25 is the concluding chapter in which we summarize the journey that we have travelled in this book.

Audience

The main audience of this book is computer science students who are interested in learning about software engineering and in learning on how to build high-quality and reliable software on time and on budget. It will also be of interest to industrialists including software engineers, quality professionals and software managers, as well as the motivated general reader.

Acknowledgments

I am deeply indebted to family and friends who supported my efforts in this endeavour, and my thanks, as always, to the team at Springer. This book is dedicated to present and past members of the Formal Methods Group (Foundations and Methods Group) at Trinity College Dublin where the author spent several happy years. I would especially like to thank Dr. Mícheál Mac An Aircinnigh, Dr. Andrew Butterfield, Dr. Hugh Gibbons, Dr. Arthur Hughes, Alexis Donnelly, Dara Gallagher, Eoin McDonnell, Gradamir Starovic, and Glenn Strong.

Cork, Ireland

Gerard O'Regan

Contents

1 Fundamentals of Software Engineering	1
1.1 Introduction	1
1.2 What is Software Engineering?	4
1.3 Challenges in Software Engineering	7
1.4 Software Processes and Lifecycles	8
1.4.1 Waterfall Lifecycle	9
1.4.2 Spiral Lifecycles	10
1.4.3 Rational Unified Process	11
1.4.4 Agile Development	12
1.4.5 Continuous Software Development	14
1.5 Activities in Software Development	15
1.5.1 Requirements Definition	15
1.5.2 Design	17
1.5.3 Implementation	18
1.5.4 Software Testing	19
1.5.5 Support and Maintenance	20
1.6 Software Inspections	21
1.7 Software Project Management	21
1.8 CMMI Maturity Model	22
1.9 Formal Methods	23
1.10 Review Questions	24
1.11 Summary	24
References	25
2 Professional Responsibility of Software Engineers	27
2.1 Introduction	27
2.2 What is a Code of Ethics?	29
2.2.1 Role of a Whistle Blower	31
2.3 IEEE Code of Ethics	33
2.4 British Computer Society Code of Conduct	34
2.5 ACM Code of Professional Conduct and Ethics	35
2.6 Precautionary Principle	37

2.7	Review Questions	38
2.8	Summary	38
3	Ethical Software Engineering	41
3.1	Introduction	41
3.2	Safety and Ethics	42
3.2.1	Therac-25 Disaster	43
3.2.2	Space Shuttle Challenger Disaster	45
3.3	Ethical Project Management.	46
3.4	Ethical Software Design and Development	48
3.4.1	Volkswagen Emissions Scandal	52
3.5	Ethical Software Testing	53
3.6	Review Questions	54
3.7	Summary	55
4	Software Project Management	57
4.1	Introduction	57
4.2	Project Start Up and Initiation	59
4.3	Estimation	60
4.3.1	Estimation Techniques	61
4.3.2	Work Breakdown Structure	61
4.4	Project Planning and Scheduling	63
4.5	Risk Management	66
4.6	People Management in Projects	67
4.7	Quality Management in Projects.	68
4.8	Project Monitoring and Control	69
4.9	Managing Issues and Change Requests.	71
4.10	Remote Project Management	71
4.11	Outsourcing	72
4.12	Project Board and Governance	73
4.13	Project Reporting	74
4.14	Project Closure	75
4.15	Prince 2 Methodology	76
4.16	Project Manager Professional	76
4.17	Project Management Office	79
4.18	Program Management	79
4.19	Project Portfolio Management	80
4.20	Project Management in the Agile World.	81
4.21	Review Questions	82
4.22	Summary	82
	References	83

5 Requirements Engineering	85
5.1 Introduction	85
5.2 Requirements Process	86
5.2.1 Requirements Elicitation and Specification	89
5.2.2 Requirements Analysis	92
5.2.3 Requirements Verification and Validation	92
5.2.4 Requirements Management	93
5.2.5 Requirements Traceability	94
5.3 System Modelling	95
5.4 Requirements Definition in the Agile World	97
5.5 Review Questions	97
5.6 Summary	98
References	99
6 Software Design and Development	101
6.1 Introduction	101
6.2 Architecture Design	102
6.3 Low-Level Design and Development	106
6.3.1 Function-Oriented Design	107
6.3.2 Object-Oriented Design	107
6.3.3 User-Interface Design	109
6.3.4 Open-Source Development	109
6.3.5 Customized-off-the-Shelf Software	110
6.3.6 Software Reuse	110
6.3.7 Design Patterns	111
6.3.8 Object-Oriented Programming	111
6.4 Software Maintenance and Evolution	112
6.5 Software Design and Development in the Agile World	113
6.6 Review Questions	114
6.7 Summary	114
References	115
7 Software Inspections	117
7.1 Introduction	117
7.2 Economic Benefits of Software Inspections	119
7.3 Informal Reviews	120
7.4 Structured Walkthrough	120
7.5 Semi-formal Review Meeting	121
7.6 Fagan Inspections	124
7.6.1 Fagan Inspection Guidelines	125
7.6.2 Inspectors and Roles	126
7.6.3 Inspection Entry Criteria	126
7.6.4 Preparation	128
7.6.5 The Inspection Meeting	128

7.6.6	Inspection Exit Criteria	130
7.6.7	Issue Severity	130
7.6.8	Defect Type	130
7.7	Automated Software Inspections	133
7.8	Review Questions	133
7.9	Summary	134
	References	135
8	Software Testing	137
8.1	Introduction	137
8.2	Test Process	139
8.3	Test Planning	143
8.4	Test Case Design and Definition	144
8.5	Test Execution	145
8.6	Test Reporting and Project Sign-Off	146
8.7	Testing and Quality Improvement	147
8.8	Traceability of Requirements	148
8.9	Test Tools	148
8.10	E-Commerce Testing	150
8.11	Testing in the Agile World	151
8.12	Review Questions	152
8.13	Summary	152
9	Ethics and Privacy	155
9.1	Introduction	155
9.2	Business Ethics	157
9.3	What is Computer Ethics?	159
9.3.1	Ethical Problems in Computing	160
9.3.2	The Ethical Software Engineer	161
9.3.3	Ethics in Data Science	162
9.4	Privacy	166
9.4.1	Social Media	172
9.4.2	Internet of Things	175
9.4.3	AI and Facial Recognition	176
9.4.4	Privacy and the Law	177
9.4.5	EU GDPR Privacy Law	178
9.5	Review Questions	179
9.6	Summary	180
	References	180
10	Software Metrics and Problem Solving	181
10.1	Introduction	181
10.2	The Goal Question Metric Paradigm	182
10.3	The Balanced Scorecard	184

10.4	Metrics for an Organization	187
10.4.1	Customer Satisfaction Metrics	187
10.4.2	Process Improvement Metrics	188
10.4.3	Human Resources and Training Metrics	190
10.4.4	Project Management Metrics	191
10.4.5	Development Quality Metrics	193
10.4.6	Quality Audit Metrics	195
10.4.7	Customer Care Metrics	197
10.4.8	Miscellaneous Metrics	199
10.5	Implementing a Metrics Program	201
10.5.1	Data Gathering for Metrics	202
10.6	Problem-Solving Techniques	203
10.6.1	Fishbone Diagram	205
10.6.2	Histograms	206
10.6.3	Pareto Chart	207
10.6.4	Trend Graphs	209
10.6.5	Scatter Graphs	209
10.6.6	Metrics and Statistical Process Control	210
10.7	Review Questions	211
10.8	Summary	212
	References	212
11	Supplier Selection and Management	213
11.1	Introduction	213
11.2	Planning and Requirements	216
11.3	Identifying Suppliers	216
11.4	Prepare and Issue RFP	217
11.5	Evaluate Proposals and Select Supplier	217
11.6	Formal Agreement	218
11.7	Managing the Supplier	219
11.8	Acceptance of Software	220
11.9	Rollout and Customer Support	220
11.10	Ethical Software Outsourcing	220
11.11	Legal Breach of Contact	222
11.12	Review Questions	224
11.13	Summary	225
12	Configuration Management	227
12.1	Introduction	227
12.2	Configuration Management System	231
12.2.1	Identify Configuration Items	232
12.2.2	Document Control Management	232
12.2.3	Source Code Control Management	233
12.2.4	Configuration Management Plan	233

12.3	Change Control	234
12.4	Configuration Management Audits	236
12.5	Review Questions	237
12.6	Summary	238
13	Software Quality Assurance	239
13.1	Introduction	239
13.2	Audit Planning	242
13.3	Audit Meeting	243
13.4	Audit Reporting	244
13.5	Follow Up Activity	245
13.6	Audit Escalation	245
13.7	Review of Audit Activities	245
13.8	Other Audits	245
13.9	Review Questions	246
13.10	Summary	246
14	Agile Methodology	247
14.1	Introduction	247
14.2	Scrum Methodology	250
14.3	User Stories	251
14.4	Estimation in Agile	252
14.5	Test Driven Development	252
14.6	Pair Programming	253
14.7	Review Questions	254
14.8	Summary	254
	Reference	255
15	Software Reliability and Dependability	257
15.1	Introduction	257
15.2	Software Reliability	258
15.2.1	Software Reliability and Defects	259
15.2.2	Cleanroom Methodology	261
15.2.3	Software Reliability Models	262
15.3	Dependability	264
15.4	Computer Security	266
15.5	System Availability	267
15.6	Safety Critical Systems	268
15.7	Review Questions	269
15.8	Summary	269
	References	270
16	Formal Methods	271
16.1	Introduction	271
16.2	Why Should We Use Formal Methods?	274

16.3	Applications of Formal Methods	275
16.4	Tools for Formal Methods	276
16.5	Approaches to Formal Methods	277
16.5.1	Model-Oriented Approach	277
16.5.2	Axiomatic Approach	279
16.6	Proof and Formal Methods	279
16.7	The Future of Formal Methods	280
16.8	The Vienna Development Method	281
16.9	VDM [®] , the Irish School of VDM	282
16.10	The Z Specification Language	283
16.11	The B Method	284
16.12	Predicate Transformers and Weakest Preconditions	285
16.13	The Process Calculi	286
16.14	Finite State Machines	287
16.15	The Parnas Way	288
16.16	Usability of Formal Methods	288
16.16.1	Why are Formal Methods Difficult?	290
16.16.2	Characteristics of a Usable Formal Method	290
16.17	Review Questions	291
16.18	Summary	292
	References	292
17	Z Specification Language	295
17.1	Introduction	295
17.2	Sets	298
17.3	Relations	299
17.4	Functions	301
17.5	Sequences	302
17.6	Bags	303
17.7	Schemas and Schema Composition	305
17.8	Reification and Decomposition	308
17.9	Proof in Z	309
17.10	Review Questions	309
17.11	Summary	310
	References	311
18	Unified Modelling Language	313
18.1	Introduction	313
18.2	Overview of UML	314
18.3	UML Diagrams	316
18.4	Object Constraint Language	322
18.5	Tools for UML	322
18.6	Rational Unified Process	323
18.7	Review Questions	325

18.8	Summary	325
	References	326
19	Software Process Improvement	327
19.1	Introduction	327
19.2	What is a Software Process?	328
19.3	What is Software Process Improvement?	330
19.4	Benefits of Software Process Improvement	331
19.5	Software Process Improvement Models	332
19.6	Process Mapping	335
19.7	Process Improvement Initiatives	336
19.8	Barriers to Success	337
19.9	Setting Up an Improvement Initiative	337
19.10	Appraisals	340
19.11	Review Questions	341
19.12	Summary	341
	References	342
20	Capability Maturity Model Integration	343
20.1	Introduction	343
20.2	The CMMI	346
20.3	CMMI Maturity Levels	349
20.3.1	CMMI Representations	352
20.4	Categories of CMMI Processes	354
20.5	CMMI Process Areas	355
20.6	Components of CMMI Process Areas	357
20.7	SCAMPI Appraisals	362
20.8	Review Questions	362
20.9	Summary	363
	References	363
21	Software Engineering Tools	365
21.1	Introduction	365
21.2	Tools for Project Management	366
21.3	Tools for Requirements	370
21.4	Tools for Design and Development	373
21.5	Tools for Agile Development	376
21.6	Tools for Configuration Management and Change Control	376
21.7	Tools for Code Analysis and Code Inspections	377
21.8	Tools for Testing	379
21.9	Review Questions	380
21.10	Summary	381
	References	382

22 A Miscellany of Innovation	383
22.1 Introduction	383
22.2 Distributed Systems	384
22.3 Service-Oriented Architecture	385
22.4 Software as a Service	386
22.5 Cloud Computing	387
22.6 Embedded Systems	388
22.7 Software Engineering and Innovation	389
22.7.1 Aspect-Oriented Software Engineering	389
22.8 Review Questions	390
22.9 Summary	390
References	391
23 Legal Aspects of Software Engineering	393
23.1 Introduction	393
23.2 Intellectual Property	395
23.2.1 Patents	396
23.2.2 Copyright	399
23.2.3 Copyright of Software	402
23.2.4 Software Licensing	404
23.3 Lawsuits	405
23.3.1 Tort in Software Engineering	406
23.3.2 Software Licenses and Failure	407
23.3.3 Legal Aspects of Outsourcing	408
23.4 Computer Crime	410
23.5 Review Questions	412
23.6 Summary	412
Reference	413
24 Cybersecurity and Cybercrime	415
24.1 Introduction	415
24.1.1 Scams	418
24.1.2 Malware	419
24.1.3 Cyberextortion and Ransomware	419
24.2 Hacking	421
24.3 Cybersecurity	423
24.4 Review Questions	429
24.5 Summary	429
References	430
25 Epilogue	431
25.1 The Future of Software Engineering	434
Glossary	435
Index	441

List of Figures

Fig. 1.1	Standish report—Results of 1995 and 2009 survey	3
Fig. 1.2	Standish 1998 report—Estimation accuracy	7
Fig. 1.3	Waterfall V lifecycle model	9
Fig. 1.4	Spiral lifecycle model ... public domain	10
Fig. 1.5	Rational unified process	12
Fig. 2.1	Whistle blower	30
Fig. 3.1	A radiotherapy machine	43
Fig. 3.2	Space challenger disaster	45
Fig. 3.3	Bridge over the River Kwai in Kanchanburi, Thailand	49
Fig. 3.4	Balancing an ethical life against a feather in Egyptian religion	50
Fig. 3.5	Volkswagen Beetle Type 82E	52
Fig. 4.1	Simple process map for project planning	63
Fig. 4.2	Sample microsoft project schedule	64
Fig. 4.3	Simple process map for project monitoring and control	70
Fig. 4.4	Prince 2 project board	74
Fig. 4.5	Project management triangle	76
Fig. 4.6	Prince 2 processes	77
Fig. 5.1	Requirements process	90
Fig. 6.1	C.A.R Hoare. (Public domain)	104
Fig. 6.2	David Parnas	105
Fig. 7.1	Michael Fagan	118
Fig. 7.2	Template for semi-formal review	123
Fig. 7.3	Template for Fagan inspection	129
Fig. 7.4	Sample-defect types in a project (ODC)	132
Fig. 8.1	Simplified test process	140
Fig. 8.2	Sample test status	142
Fig. 8.3	Cumulative defects	146
Fig. 9.1	Corrupt legislation. 1896. Public domain	157
Fig. 9.2	Bentham's panopticon prison	167
Fig. 9.3	Cardinals eavesdropping in the Vatican	170
Fig. 9.4	Young peoples on smart phones and social media. Public domain	173

Fig. 9.5	Fitbit Surge. Smart-watch activity tracker. Creative commons	175
Fig. 9.6	EU GDPR 2016/679	179
Fig. 10.1	GQM example	183
Fig. 10.2	The balanced scorecard	185
Fig. 10.3	Balanced score card and implementing strategy	185
Fig. 10.4	Customer survey arrivals	187
Fig. 10.5	Customer satisfaction measurements	188
Fig. 10.6	Process improvement measurements	188
Fig. 10.7	Status of process improvement suggestions	189
Fig. 10.8	Age of open process improvement suggestions	189
Fig. 10.9	Process improvement productivity	190
Fig. 10.10	Employee headcount in current year	190
Fig. 10.11	Employee turnover in current year	191
Fig. 10.12	Schedule timeliness metric	191
Fig. 10.13	Effort timeliness metric	192
Fig. 10.14	Requirements delivered	192
Fig. 10.15	Total number of issues in project	193
Fig. 10.16	Open issues in project	193
Fig. 10.17	Age of open defects in project	194
Fig. 10.18	Problem arrivals per month	194
Fig. 10.19	Phase containment effectiveness	195
Fig. 10.20	Annual audit schedule	196
Fig. 10.21	Status of audit actions	196
Fig. 10.22	Audit action types	196
Fig. 10.23	Customer queries (arrivals/closures)	198
Fig. 10.24	Outage time per customer	198
Fig. 10.25	Availability of system per month	199
Fig. 10.26	Configuration management	199
Fig. 10.27	CMMI maturity in current year	200
Fig. 10.28	Cost of poor quality (COPQ)	201
Fig. 10.29	Fishbone cause-and-effect diagram	205
Fig. 10.30	Histogram	207
Fig. 10.31	Pareto chart outages	208
Fig. 10.32	Trend chart estimation accuracy	209
Fig. 10.33	Scatter graph amount inspected rate/error density	210
Fig. 10.34	Estimation accuracy and control charts	211
Fig. 11.1	Legal contract	218
Fig. 12.1	Simple process map for change requests	235
Fig. 12.2	Simple process map for configuration management	236
Fig. 13.1	Sample audit process	241
Fig. 16.1	Deterministic finite state machine	288
Fig. 17.1	Specification of positive square root	296
Fig. 17.2	Specification of a library system	297

Fig. 17.3	Specification of borrow operation	298
Fig. 17.4	Specification of vending machine using bags	304
Fig. 17.5	Schema inclusion	305
Fig. 17.6	Merging schemas ($S_1 \vee S_2$)	305
Fig. 17.7	Schema composition	307
Fig. 17.8	Refinement commuting diagram	308
Fig. 18.1	Simple object diagram	318
Fig. 18.2	Use-case diagram of ATM machine	319
Fig. 18.3	UML sequence diagram for balance enquiry	320
Fig. 18.4	UML activity diagram	321
Fig. 18.5	UML state diagram	321
Fig. 18.6	Iteration in rational unified process	324
Fig. 18.7	Phases and workflows in rational unified process	325
Fig. 19.1	Process as glue for people, procedures and tools	329
Fig. 19.2	Sample process map	330
Fig. 19.3	Steps in process improvement	331
Fig. 19.4	ISO 9001 quality management system	334
Fig. 19.5	Continuous improvement cycle	338
Fig. 19.6	Appraisals	340
Fig. 20.1	Process as glue for people, procedures and tools	344
Fig. 20.2	ISO/IEC 12207 standard for software engineering processes	345
Fig. 20.3	CMMI worldwide maturity 2013	348
Fig. 20.4	CMMI maturity levels	349
Fig. 20.5	CMMI capability levels	352
Fig. 20.6	CMMI—continuous representation	352
Fig. 20.7	CMMI staged model	357
Fig. 20.8	Specific practices for SG1—manage requirements	358
Fig. 21.1	Dashboard views in Planview Enterprise	369
Fig. 21.2	Planview process builder	370
Fig. 21.3	IBM Rational Doors tool	372
Fig. 21.4	IBM Rational Software Modeler	374
Fig. 21.5	Sparx Enterprise Architect	375
Fig. 21.6	LDRA code coverage analysis report	378
Fig. 21.7	HP Quality Center	380
Fig. 22.1	A distributed system	384
Fig. 22.2	Service-oriented architecture	386
Fig. 22.3	Cloud computing. Creative Commons	387
Fig. 22.4	Example of an embedded system	388
Fig. 23.1	Patent for an invention	397
Fig. 23.2	St. Colomba's Cathach	400
Fig. 23.3	Legal contract. Creative Commons	409
Fig. 23.4	Dandy Pickpockets (1818)	411

Fig. 24.1	Trojan horse at Troy	420
Fig. 24.2	Hacker at work on blacklit keyboard. Creative Commons	421
Fig. 24.3	Sun Tzu Wu	427

List of Tables

Table 2.1	Professional responsibilities of software engineers and testers	28
Table 2.2	Types of professional codes	29
Table 2.3	Steps in whistle blowing	32
Table 2.4	IEEE code of ethics	33
Table 2.5	BCS code of conduct	35
Table 2.6	ACM code of conduct	36
Table 4.1	Estimation techniques	62
Table 4.2	Example work-breakdown structure	62
Table 4.3	Sample project management checklist	65
Table 4.4	Risk management activities	66
Table 4.5	Activities in managing issues and change requests	71
Table 4.6	Project board roles and responsibilities	74
Table 4.7	Key processes in Prince 2	77
Table 4.8	PMBOK process groups	78
Table 4.9	PMBOK knowledge areas	78
Table 4.10	Functions of project management office	79
Table 5.1	Characteristics of good requirements	87
Table 5.2	Symptoms of poor requirements process	88
Table 5.3	Managing change requests	94
Table 5.4	Sample trace matrix	95
Table 6.1	Views of system architecture	103
Table 6.2	Object-oriented paradigm	108
Table 7.1	Informal review	120
Table 7.2	Structured walkthroughs	121
Table 7.3	Activities for semi-formal review meeting	122
Table 7.4	Overview Fagan inspection process	124
Table 7.5	Strict Fagan inspection guidelines	125
Table 7.6	Tailored (Relaxed) Fagan inspection guidelines	126
Table 7.7	Inspector roles	127
Table 7.8	Fagan entry criteria	127
Table 7.9	Inspection meeting	128
Table 7.10	Fagan exit criteria	130
Table 7.11	Issue severity	131

Table 7.12	Classification of defects in Fagan inspections	131
Table 7.13	Classification of ODC defect types	131
Table 8.1	Types of testing	141
Table 8.2	Simple test schedule	144
Table 9.1	Ten commandments on computer ethics	159
Table 9.2	Some ethical problems in computing	161
Table 9.3	Sources of information	168
Table 10.1	BSC objectives and measures for IT service organization	186
Table 10.2	Cost of quality categories	200
Table 10.3	Implementing metrics	201
Table 10.4	Goals and questions	202
Table 10.5	Phase containment effectiveness	202
Table 11.1	Supplier selection and management	215
Table 11.2	Possible breaches of contract	224
Table 12.1	Features of good configuration management	228
Table 12.2	Symptoms of poor configuration management	229
Table 12.3	Software configuration management activities	230
Table 12.4	Build plan for project	230
Table 12.5	CMMI requirements for configuration management	231
Table 12.6	Sample configuration management audit checklist	237
Table 13.1	Auditing activities	240
Table 13.2	Sample auditing checklist	243
Table 13.3	Sample audit report	244
Table 15.1	Adam's 1984 study of software failures of IBM products	260
Table 15.2	New and old version of software	261
Table 15.3	Cleanroom results in IBM	262
Table 15.4	Characteristics of good software reliability model	263
Table 15.5	Software reliability models	264
Table 15.6	Dimensions of dependability	265
Table 16.1	Criticisms of formal methods	273
Table 16.2	Parnas's contributions to software engineering	289
Table 16.3	Techniques for validation of formal specification	290
Table 16.4	Why are formal methods difficult?	290
Table 16.5	Characteristics of a usable formal method	291
Table 17.1	Schema composition	307
Table 18.1	Classification of UML things	315
Table 18.2	UML diagrams	316
Table 18.3	Simple class diagram	317
Table 18.4	Advantages of UML	322
Table 18.5	OCL constraints	323
Table 18.6	UML tools	323
Table 19.1	Benefits of software process improvement (CMMI)	332
Table 19.2	Continuous improvement cycle	339
Table 19.3	Teams in improvement program	339

Table 19.4	Phases in an appraisal	341
Table 20.1	Motivation for CMMI implementation	347
Table 20.2	Benefits of CMMI implementation	349
Table 20.3	CMMI maturity levels	350
Table 20.4	CMMI capability levels for continuous representation	353
Table 20.5	CMMI process categories	354
Table 20.6	CMMI process areas	355
Table 20.7	CMMI generic practices	359
Table 20.8	Implementation of generic practices	361
Table 21.1	Tool evaluation table	366
Table 21.2	Key capabilities of Planview Enterprise	369
Table 21.3	Tools for requirements development and management	371
Table 21.4	Tools for software design	373
Table 21.5	Integrated development environment	375
Table 21.6	Features of Jira for Agile project management	376
Table 23.1	Process for obtaining a patent	398
Table 23.2	Types of lawsuits	406
Table 24.1	Computer crimes	417