Undergraduate Topics in Computer Science

Series editor
Ian Mackie

Advisory Boards
Samson Abramsky, University of Oxford, Oxford, UK
Karin Breitman, Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil
Chris Hankin, Imperial College London, London, UK
Dexter Kozen, Cornell University, Ithaca, USA
Andrew Pitts, University of Cambridge, Cambridge, UK
Hanne Riis Nielson, Technical University of Denmark, Kongens Lyngby, Denmark
Steven Skiena, Stony Brook University, Stony Brook, USA
Iain Stewart, University of Durham, 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 include fully worked solutions.

More information about this series at http://www.springer.com/series/7592

Gerard O'Regan

Concise Guide to Software Engineering

From Fundamentals to Application Methods

Springer

Gerard O'Regan
SQC Consulting
Cork
Ireland

ISSN 1863-7310 ISSN 2197-1781 (electronic)
Undergraduate Topics in Computer Science
ISBN 978-3-319-57749-4 ISBN 978-3-319-57750-0 (eBook)
DOI 10.1007/978-3-319-57750-0

Library of Congress Control Number: 2017939621

© Springer International Publishing AG 2017
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, express 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.

Printed on acid-free paper

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

In memory of my dear godmother
Mrs. Maureen Barry

Preface

Overview

The objective of this book was 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 phases 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 provides an introduction to 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 3 discusses requirements engineering and discusses activities such as requirements gathering, requirements elicitation, requirements analysis, requirements management, and requirements verification and validation.

Chapter 4 discusses design and development, and 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 programmes. 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 5 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 6 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 7 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 8 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 9 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 10 is concerned with software 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 Metric (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 and trend charts are discussed.

Chapter 11 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 12 discusses formal methods, which consist of a set of mathematical techniques to specify and derive a programme from its specification. Formal methods may be employed to rigorously state the requirements of the proposed system. They may be employed to derive a programme from its mathematical specification, and they may be used to provide a rigorous proof that the implemented programme satisfies its specification. They have been mainly applied to the safety critical field.

Chapter 13 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 14 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 15 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 16 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 17 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 18 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 19 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 20 is the concluding chapter in which we summarize the journey that we have travelled in this book.

Audience

The main audience of this book are 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.

Acknowledgements

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 my late godmother (Mrs. Maureen Barry), who I always enjoyed visiting in Ringaskiddy, Co. Cork.

Cork, Ireland Gerard O'Regan

Contents

List of Figures

List of Tables