Software Maintenance Concepts And Practice

Concepts, Methodologies, Tools, and Applications
Controlling Software Projects
Software Maintenance
Software and Mind
Software Architect’s Handbook
A Practical Approach
Software and Systems Traceability
Software Engineering Design
Effective Practices for Geographically Distributed Environments
Software Maintenance Management
Software Architecture in Practice
Software Evolution and Maintenance
Software Engineering at Google
SOFTWARE DESIGN, ARCHITECTURE AND ENGINEERING
Site Reliability Engineering
Just Enough Software Architecture
Overcoming Challenges in Software Engineering Education: Delivering Non-Technical Knowledge and Skills
Software Applications: Concepts, Methodologies, Tools, and Applications
The Mechanistic Myth and Its Consequences
Agile Software Requirements
Practical Software Testing
Sustainable Forest Management
Concepts and Implementation
Measuring the Software Process
Concepts and Practice
Introduction to Software Testing
Technology Assessment in Practice and Theory
Designing Software Architectures
Feature-Oriented Software Product Lines
Software Quality Assurance
From Theory to Implementation
Concepts, Methodologies, Tools, and Applications
Software Testing and Quality Assurance
Concepts and Practice
CONCEPTS AND PRACTICE
Lessons Learned from Programming Over Time
Data Science
Concepts, Methodologies, Tools, and Applications
Software Quality
A Process-Oriented Approach

DEVYN HUGHES

Concepts, Methodologies, Tools, and Applications Addison-Wesley Professional
Includes articles in topic areas such as autonomic computing, operating system architectures, and open source software technologies and applications.
Controlling Software Projects John Wiley & Sons
Designing Software Architectures will teach you how to design any software architecture in a systematic, predictable, repeatable, and cost-effective way. This book introduces a practical methodology for architecture design that any professional software engineer can use, provides structured methods supported by reusable chunks of design knowledge, and includes rich case studies that demonstrate how to use the methods. Using realistic examples, you’ll master the powerful new version of the proven Attribute-Driven Design (ADD) 3.0 method and will learn how to use it to address key drivers, including quality attributes, such as modifiability, usability, and availability, along with functional requirements and architectural concerns. Drawing on their extensive experience, Humberto Cervantes and Rick Kazman guide you through crafting practical designs that support the full software life cycle, from requirements to maintenance and evolution. You’ll learn how to successfully integrate design in your organizational context, and how to design systems that will be built with agile methods. Comprehensive coverage includes Understanding what architecture design involves, and where it fits in the full software development life cycle Mastering core design concepts, principles, and processes Understanding how to perform the steps of the ADD method Scaling design and analysis up or down, including design for pre-sale processes or lightweight architecture reviews Recognizing and optimizing critical relationships between analysis and design Utilizing proven, reusable design primitives and adapting them to specific problems and contexts Solving design problems in new domains, such as cloud, mobile, or big data
Software Maintenance Software MaintenanceConcepts and Practice Software MaintenanceConcepts and PracticeWorld Scientific
Software and Mind World Scientific
Modern-day projects require software and

systems engineers to work together in realizing architectures of large and complex software-intensive systems. To date, the two have used their own tools and methods to deal with similar issues when it comes to the requirements, design, testing, maintenance, and evolution of these architectures. Software and Systems Architecture in Action explores practices that can be helpful in the development of architectures of large-scale systems in which software is a major component. Examining the synergies that exist between the disciplines of software and systems engineering, it presents concepts, techniques, and methods for creating and documenting architectures. The book describes an approach to architecture design that is driven from systemic quality attributes determined from both the business and technical goals of the system, rather than just its functional requirements. This architecture-centric design approach utilizes analytically derived patterns and tactics for quality attributes that inform the architect's design choices and help shape the architecture of a given system. The book includes coverage of techniques used to assess the impact of architecture-centric design on the structural complexity of a system. After reading the book, you will understand how to create architectures of systems and assess their ability to meet the business goals of your organization. Ideal for anyone involved with large and complex software-intensive systems, the book details powerful methods for engaging the software and systems engineers on your team. The book is also suitable for use in undergraduate and graduate-level courses on software and systems architecture as it exposes students to the concepts and techniques used to create and manage architectures of software-intensive systems.

Software Architect's Handbook Packt Publishing Ltd

Technological advance affects almost all areas of human life. Rapid digitization, increased mobility, new biotechnologies, and nanotechnology deeply influence, amongst others, industrial production, entertainment, work, military affairs, and individual life. Besides overwhelmingly positive effects on wealth, comfort, innovation, and development, this also raises questions of unintended effects, of tensions with democracy, of the role of citizens, and of its sustainability facing environmental issues. Tools and procedures are needed to cope with this challenging situation. Technology assessment (TA) has been developed more than fifty years ago to enable science, the economy, and society to harvest the potential of new technology to the maximum extent possible and to deal responsibly with possible adverse effects. It was developed more than 50 years ago in the U.S. Congress and has diversified considerably in the meantime. Parliamentary TA in many European states and at the international level, participatory TA at the local and regional levels worldwide, and TA as part of engineering processes are the most relevant fields today. Technology assessment is a growing field of interdisciplinary research and scientific policy advice. This volume (a) gives an overview of motivations of TA, its history and its current practices, (b) develops a fresh theoretical perspective on TA rooted in social theory and philosophy, and (c) draws conclusions from the theoretical perspective for the further development of TA's practices. It provides the first comprehensive view on the growing field of TA at the international level.

A Practical Approach IGI Global

Software systems now invade every area of daily living. Yet, we still struggle to build systems we can really rely on. If we want to work with software systems at any level, we need to get to grips with the way software evolves. This book will equip the reader with a sound understanding of maintenance and how it affects all levels of the software evolution process.

Software and Systems Traceability Pearson Education

This revised edition of Software Engineering-Principles and Practices has become more comprehensive with the inclusion of several topics. The book now offers a complete understanding of software engineering as an engineering discipline. Like its previous edition, it provides an in-depth coverage of fundamental principles, methods and applications of software engineering. In addition, it covers some advanced approaches including Computer-aided Software Engineering (CASE), Component-based Software Engineering (CBSE), Clean-room Software Engineering (CSE) and formal methods.Taking into account the needs of both students and practitioners, the book presents a pragmatic picture of the software engineering methods and tools. A thorough study of the software industry shows that there exists a substantial difference between classroom study and the practical industrial application. Therefore, earnest efforts have been made in this book to bridge the gap between theory and practical applications. The subject matter is well supported by examples and case studies representing the situations that one actually faces during the software development process.The book meets the requirements of students enrolled in various courses both at the undergraduate and postgraduate levels, such as BCA, BE, BTech, BIT, BIS, BSc, PGDCA, MCA, MIT, MIS, MSc, various DOEACC levels and so on. It will also be suitable for those software engineers who abide by scientific principles and wish to expand their knowledge. With the increasing demand of software, the software engineering discipline has become important in education and industry. This thoughtfully organized second edition of the book provides its readers a profound knowledge of software engineering concepts and principles in a simple, interesting and illustrative manner.

Software Engineering Design Taylor & Francis

This textbook aims to prepare students, as well as, practitioners for software design and production. Keeping in mind theory and practice, the book keeps a balance between theoretical foundations and practical considerations. The book by and large meets the requirements of students at all levels of computer science and engineering/information technology for their Software design and Software engineering courses. The book begins with concepts of data and object. This helps in exploring the rationale that guide high level programming language (HLL) design and object oriented frameworks. Once past this post, the book moves on to expand on software design concerns. The book emphasizes the centrality of Parnas's separation of concerns in evolving software designs and architecture. The book extensively explores modelling frameworks such as Unified Modelling Language (UML) and Petri net based methods. Next, the book covers architectural principles and software engineering practices such as Agile – emphasizing software testing during development. It winds up with case studies demonstrating how systems evolve from basic concepts to final products for quality software designs. TARGET AUDIENCE • Undergraduate/postgraduate students of Computer Science and Engineering, and Information Technology • Postgraduate students of Software Engineering/Software Systems

Effective Practices for Geographically Distributed Environments World Scientific

Sustainable Forest Management provides the necessary material to educate

students about forestry and the contemporary role of forests in ecosystems and society. This comprehensive textbook on the concept and practice of sustainable forest management sets the standard for practice worldwide. Early chapters concentrate on conceptual aspects, relating sustainable forestry management to international policy. In particular, they consider the concept of criteria and indicators and how this has determined the practice of forest management, taken here to be the management of forested lands and of all ecosystems present on such lands. Later chapters are more practical in focus, concentrating on the management of the many values associated with forests. Overall the book provides a major new synthesis which will serve as a textbook for undergraduates of forestry as well as those from related disciplines such as ecology or geography who are taking a course in forests or natural resource management.

Software Maintenance Management "O'Reilly Media, Inc."

Controlling Software Projects shows managers how to organize software projects so they are objectively measurable, and prescribes techniques for making early and accurate projections of time and cost to deliver.

Software Architecture in Practice John Wiley & Sons

Extensively class-tested, this textbook takes an innovative approach to software testing: it defines testing as the process of applying a few well-defined, general-purpose test criteria to a structure or model of the software. It incorporates the latest innovations in testing, including techniques to test modern types of software such as OO, web applications, and embedded software. The book contains numerous examples throughout. An instructor's solution manual, PowerPoint slides, sample syllabi, additional examples and updates, testing tools for students, and example software programs in Java are available on an extensive website.

Software Evolution and Maintenance Marshall & Brainerd

Computer science graduates often find software engineering knowledge and skills are more in demand after they join the industry. However, given the lecture-based curriculum present in academia, it is not an easy undertaking to deliver industry-standard knowledge and skills in a software engineering classroom as such lectures hardly engage or convince students. Overcoming Challenges in Software Engineering Education: Delivering Non-Technical Knowledge and Skills combines recent advances and best practices to improve the curriculum of software engineering education. This book is an essential reference source for researchers and educators seeking to bridge the gap between industry expectations and what academia can provide in software engineering education.

Software Engineering at Google Routledge

"While it is usually helpful to launch improvement programs, many such programs soon get bogged down in detail. They either address the wrong problems, or they keep beating on the same solutions, wondering why things don't improve. This is when you need an objective way to look at the problems. This is the time to get some data." Watts S. Humphrey, from the Foreword This book, drawing on work done at the Software Engineering Institute and other organizations, shows how to use measurements to manage and improve software processes. The authors explain specifically how quality characteristics of software products and processes can be quantified, plotted, and analyzed so the performance of software development activities can be predicted, controlled, and guided to achieve both business and technical goals. The measurement methods presented, based on the principles of statistical quality control, are illuminated by application examples taken from industry. Although many of the methods discussed are applicable to individual projects, the book's primary focus is on the steps software development organizations can take toward broad-reaching, long-term success. The book particularly addresses the needs of software managers and practitioners who have already set up some kind of basic measurement process and are ready to take the next step by collecting and analyzing software data as a basis for making process decisions and predicting process performance. Highlights of the book include: Insight into developing a clear framework for measuring process behavior Discussions of process performance, stability, compliance, capability, and improvement Explanations of what you want to measure (and why) and instructions on how to collect your data Step-by-step guidance on how to get started using statistical process control If you have responsibilities for product quality or process performance and you are ready to use measurements to manage, control, and predict your software processes, this book will be an invaluable resource.

SOFTWARE DESIGN, ARCHITECTURE AND ENGINEERING Addison-Wesley Professional

While standardization has empowered the software industry to substantially scale software development and to provide affordable software to a broad market, it often does not address smaller market segments, nor the needs and wishes of individual customers. Software product lines reconcile mass production and standardization with mass customization in software engineering. Ideally, based on a set of reusable parts, a software manufacturer can generate a software product based on the requirements of its customer. The concept of features is central to achieving this level of automation, because features bridge the gap between the requirements the customer has and the functionality a product provides. Thus features are a central concept in all phases of product-line development. The authors take a developer’s viewpoint, focus on the development, maintenance, and implementation of product-line variability, and especially concentrate on automated product derivation based on a user’s feature selection. The book consists of three parts. Part I provides a general introduction to feature-oriented software product lines, describing the product-line approach and introducing the product-line development process with its two elements of domain and application engineering. The pivotal part II covers a wide variety of implementation techniques including design patterns, frameworks, components, feature-oriented programming, and aspect-oriented programming, as well as tool-based approaches including preprocessors, build systems, version-control systems, and virtual separation of concerns. Finally, part III is devoted to advanced topics related to feature-oriented product lines like refactoring, feature interaction, and analysis tools specific to product lines. In addition, an appendix lists various helpful tools for software product-line development, along with a description of how they relate to the topics covered in this book. To tie the book together, the authors use two running examples that are well documented in the product-line literature: data management for embedded systems, and variations of graph data structures. They start every chapter by explicitly stating the respective learning goals and finish it with a set of exercises; additional teaching material is also available online. All these features make the book ideally suited for teaching – both for academic classes and for professionals interested in self-study.

Site Reliability Engineering Addison-Wesley Professional
Learn the basics of Data Science through an easy to understand conceptual framework and immediately practice using RapidMiner platform. Whether you are brand new to data science or working on your tenth project, this book will show you how to analyze data, uncover hidden patterns and relationships to aid important decisions and predictions. Data Science has become an essential tool to extract value from data for any organization that collects, stores and processes data as part of its operations. This book is ideal for business users, data analysts, business analysts, engineers, and analytics professionals and for anyone who works with data. You'll be able to: Gain the necessary knowledge of different data science techniques to extract value from data. Master the concepts and inner workings of 30 commonly used powerful data science algorithms. Implement step-by-step data science process using using RapidMiner, an open source GUI based data science platform Data Science techniques covered: Exploratory data analysis, Visualization, Decision trees, Rule induction, k-nearest neighbors, Naïve Bayesian classifiers, Artificial neural networks, Deep learning, Support vector machines, Ensemble models, Random forests, Regression, Recommendation engines, Association analysis, K-Means and Density based clustering, Self organizing maps, Text mining, Time series forecasting, Anomaly detection, Feature selection and more... Contains fully updated content on data science, including tactics on how to mine business data for information Presents simple explanations for over twenty powerful data science techniques Enables the practical use of data science algorithms without the need for programming Demonstrates processes with practical use cases Introduces each algorithm or technique and explains the workings of a data science algorithm in plain language Describes the commonly used setup options for the open source tool RapidMiner
Just Enough Software Architecture Andsor Books
"We need better approaches to understanding and managing software requirements, and Dean provides them in this book. He draws ideas from three very useful intellectual pools: classical management practices, Agile methods, and lean product development. By combining the strengths of these three approaches, he has produced something that works better than any one in isolation." –From the Foreword by Don Reinertsen, President of Reinertsen & Associates; author of Managing the Design Factory; and leading expert on rapid product development Effective requirements discovery and analysis is a critical best practice for serious application development. Until now, however, requirements and Agile methods have rarely coexisted peacefully. For many enterprises considering Agile approaches, the absence of effective and scalable Agile requirements processes has been a showstopper for Agile adoption. In Agile Software Requirements, Dean Leffingwell shows exactly how to create effective requirements in Agile environments. Part I presents the "big picture" of Agile requirements in the enterprise, and describes an overall process model for Agile requirements at the project team, program, and portfolio levels Part II describes a simple and lightweight, yet comprehensive model that Agile project teams can use to manage requirements Part III shows how to develop Agile requirements for complex systems that require the cooperation of multiple teams Part IV guides enterprises in developing Agile requirements for ever-larger "systems of systems," application suites, and product portfolios This book will help you leverage the benefits of Agile without sacrificing the value of effective requirements discovery and analysis. You'll find proven solutions you can apply right now–whether you're a software developer or tester, executive, project/program manager, architect, or team leader.
Overcoming Challenges in Software Engineering Education: Delivering Non-Technical Knowledge and Skills Springer Science & Business Media
Software maintenance, the work done on a software system after it becomes operational, consumes at least half of all technical and management resources expended in the software area. This volume supplies an overview of software maintenance : what it is, how to do it, how to manage it, and trends in current research. The thirty-one papers included are frequently requested from their authors, from hard-to-find sources, cover the foundations of current thinking on this topic, and extend the frontiers of research.
Software Applications: Concepts, Methodologies, Tools, and Applications Tata McGraw-Hill Education
"Readership: Researchers, graduate students and undergraduates in software engineering, programming, information engineering, health informatics and medical informatics; practitioners and industrialists in software development and maintenance."--BOOK JACKET.Title Summary field provided by Blackwell North America, Inc. All Rights Reserved
The Mechanistic Myth and Its Consequences IEEE Computer Society
The most comprehensive General, Organic, and Biochemistry book available, Introduction to General, Organic, and Biochemistry, 11th Edition continues its tradition of a solid development of problem-solving skills, numerous examples and practice problems, along with coverage of current applications. Written by an experienced author team, they skillfully anticipate areas of difficulty and pace the book accordingly. Readers will find the right mix of general chemistry compared to the discussions on organic and biochemistry. Introduction to General, Organic, and Biochemistry, 11th Edition has clear & logical explanations of chemical concepts and great depth of coverage as well as a clear, consistent writing style which provides great readability. An emphasis on Real-World aspects of chemistry makes the reader comfortable in seeing how the chemistry will apply to their career.
Agile Software Requirements John Wiley & Sons
Taking a learn-by-doing approach, Software Engineering Design: Theory and Practice uses examples, review questions, chapter exercises, and case study assignments to provide students and practitioners with the understanding required to design complex software systems. Explaining the concepts that are immediately relevant to software designers, it begins with a review of software design fundamentals. The text presents a formal top-down design process that consists of several design activities with varied levels of detail, including the macro-, micro-, and construction-design levels. As part of the top-down approach, it provides in-depth coverage of applied architectural, creational, structural, and behavioral design patterns. For each design issue covered, it includes a step-by-step breakdown of the execution of the design solution, along with an evaluation, discussion, and justification for using that particular solution. The book outlines industry-proven software design practices for leading large-scale software design efforts, developing reusable and high-quality software systems, and producing technical and customer-driven design documentation. It also: Offers one-stop guidance for mastering the Software Design & Construction sections of the official Software Engineering Body of Knowledge (SWEBOK®) Details a

collection of standards and guidelines for structuring high-quality code Describes techniques for analyzing and evaluating the quality of software designs Collectively, the text supplies comprehensive coverage of the software design concepts students will need to succeed as professional design leaders.

The section on engineering leadership for software designers covers the necessary ethical and leadership skills required of software developers in the public domain. The section on creating software design documents (SDD) familiarizes students with the software design notations, structural descriptions, and behavioral models required for SDDs. Course notes, exercises with answers, online resources, and an instructor’s manual are available upon qualified course adoption. Instructors can contact the author about these resources via the author's website: http://softwareengineeringdesign.com/

Related with Software Maintenance Concepts And Practice:
• Intentional Torts Practice Multiple Choice Questions : click here