

Software Estimation Demystifying The Black Art

Thank you categorically much for downloading **Software Estimation Demystifying The Black Art** .Maybe you have knowledge that, people have see numerous period for their favorite books with this Software Estimation Demystifying The Black Art , but end going on in harmful downloads.

Rather than enjoying a good PDF past a cup of coffee in the afternoon, instead they juggled in imitation of some harmful virus inside their computer. **Software Estimation Demystifying The Black Art** is to hand in our digital library an online access to it is set as public correspondingly you can download it instantly. Our digital library saves in fused countries, allowing you to get the most less latency time to download any of our books bearing in mind this one. Merely said, the Software Estimation Demystifying The Black Art is universally compatible gone any devices to read.

The Productive Programmer - Neal Ford 2008-07-03

Anyone who develops software for a living needs a proven way to produce it better, faster, and cheaper. The Productive Programmer offers critical timesaving and productivity tools that you can adopt right away, no matter what platform you use. Master developer Neal Ford not only offers advice on the mechanics of productivity-how to work smarter, spurn interruptions, get the most out your computer, and avoid repetition-he also details valuable practices that will help you elude common traps, improve your code, and become more valuable to your team. You'll learn to: Write the test before you write the code Manage the lifecycle of your objects fastidiously Build only what you need now, not what you might need later Apply ancient philosophies to software development Question authority, rather than blindly adhere to standards Make hard things easier and impossible things possible through meta-programming Be sure all code within a method is at the same level of abstraction Pick the right editor and assemble the best tools for the job This isn't theory, but the fruits of Ford's real-world experience as an Application Architect at the global IT consultancy ThoughtWorks. Whether you're a beginner or a pro with years of experience, you'll improve your work and your career with the simple and straightforward principles in The Productive Programmer. *TSP(SM) Leading a Development Team* - Watts

S. Humphrey 2005-09-06

Leaders of software-development projects face many challenges. First, you must produce a quality product on schedule and on budget. Second, you must foster and encourage a cohesive, motivated, and smoothly operating team. And third, you must maintain a clear and consistent focus on short- and long-term goals, while exemplifying quality standards and showing confidence and enthusiasm for your team and its efforts. Most importantly, as a leader, you need to feel and act responsible for your team and everything that it does. Accomplishing all these goals in a way that is rewarding for the leader and the team--while producing the results that management wants--is the motivation behind the Team Software Process (TSP). Developed by renowned quality expert Watts S. Humphrey, TSP is a set of new practices and team concepts that helps developers take the CMM and CMMI Capability Maturity Models to the next level. Not only does TSP help make software more secure, it results in an average production gain of 68 percent per project. Because of their quality, timeliness, and security, TSP-produced products can be ten to hundreds of times better than other hardware or software. In this essential guide to TSP, Humphrey uses his vast industry experience to show leaders precisely how to lead teams of software engineers trained in the Personal Software Process (PSP). He explores all aspects of effective leadership and teamwork, including

building the right team for the job, the TSP launch process, following the process to produce a quality product, project reviews, and capitalizing on both the leader's and team's capabilities. Humphrey also illuminates the differences between an ineffective leader and a superb one with the objective of helping you understand, anticipate, and correct the most common leadership failings before they undermine the team. An extensive set of appendices provides additional detail on TSP team roles and shows you how to use an organization's communication and command networks to achieve team objectives. Whether you are a new or an experienced team leader, TSPSM: Leading a Development Team provides invaluable examples, guidelines, and suggestions on how to handle the many issues you and your team face together.

Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort & Duration - Peter Hill 2010-08-29

Product verifiable, defensible, and achievable software estimates Based on data collected by the International Software Benchmarking Standards Group (ISBSG), Practical Software Project Estimation explains how to accurately forecast the size, cost, and schedule of software projects. Get expert advice on generating accurate estimates, minimizing risks, and planning and managing projects. Valuable appendixes provide estimation equations, delivery rate tables, and the ISBSG Repository demographics. Verify project objectives and requirements Determine, validate, and refine software functional size Produce indicative estimates using regression equations Predict effect and duration through comparison and analogy Build estimation frameworks Perform benchmarks using the ISBSG Repository Compare IFPUG, COSMIC, and FiSMA sizing methods Peter Hill is the chief executive officer and a director of the ISBSG. He has been in the information services industry for more than 40 years and has compiled and edited five books for the ISBSG.

Code Complete, 2nd Edition - Steve McConnell

Widely considered one of the best practical guides to programming, Steve McConnell's original CODE COMPLETE has been helping

developers write better software for more than a decade. Now this classic book has been fully updated and revised with leading-edge practices-and hundreds of new code samples-illustrating the art and science of software construction. Capturing the body of knowledge available from research, academia, and everyday commercial practice, McConnell synthesizes the most effective techniques and must-know principles into clear, pragmatic guidance. No matter what your experience level, development environment, or project size, this book will inform and stimulate your thinking-and help you build the highest quality code.

The Agile Samurai - Jonathan Rasmusson 2010-09-25

Printed in full color. Faced with a software project of epic proportions? Tired of over-committing and under-delivering? Enter the dojo of the agile samurai, where agile expert Jonathan Rasmusson shows you how to kick-start, execute, and deliver your agile projects. Combining cutting-edge tools with classic agile practices, The Agile Samurai gives you everything you need to deliver something of value every week and make rolling your software into production a non-event. Get ready to kick some software project butt. By learning the ways of the agile samurai you will discover: how to create plans and schedules your customer and your team can believe in what characteristics make a good agile team and how to form your own how to gather requirements in a fraction of the time using agile user stories what to do when you discover your schedule is wrong, and how to look like a pro correcting it how to execute fiercely by leveraging the power of agile software engineering practices By the end of this book you will know everything you need to set up, execute, and successfully deliver agile projects, and have fun along the way. If you're a project lead, this book gives you the tools to set up and lead your agile project from start to finish. If you are an analyst, programmer, tester, usability designer, or project manager, this book gives you the insight and foundation necessary to become a valuable agile team member. The Agile Samurai slices away the fluff and theory that make other books less-than-agile. It's packed with best practices, war stories, plenty of humor and hands-on tutorial exercises that will

get you doing the right things, the right way. This book will make a difference.

The Robert C. Martin Clean Code Collection (Collection) - Robert C. Martin 2011-11-10

The Robert C. Martin Clean Code Collection consists of two bestselling eBooks: Clean Code: A Handbook of Agile Software Craftmanship The Clean Coder: A Code of Conduct for Professional Programmers In Clean Code, legendary software expert Robert C. Martin has teamed up with his colleagues from Object Mentor to distill their best agile practice of cleaning code "on the fly" into a book that will instill within you the values of a software craftsman and make you a better programmer--but only if you work at it. You will be challenged to think about what's right about that code and what's wrong with it. More important, you will be challenged to reassess your professional values and your commitment to your craft. In The Clean Coder, Martin introduces the disciplines, techniques, tools, and practices of true software craftsmanship. This book is packed with practical advice--about everything from estimating and coding to refactoring and testing. It covers much more than technique: It is about attitude. Martin shows how to approach software development with honor, self-respect, and pride; work well and work clean; communicate and estimate faithfully; face difficult decisions with clarity and honesty; and understand that deep knowledge comes with a responsibility to act. Readers of this collection will come away understanding How to tell the difference between good and bad code How to write good code and how to transform bad code into good code How to create good names, good functions, good objects, and good classes How to format code for maximum readability How to implement complete error handling without obscuring code logic How to unit test and practice test-driven development What it means to behave as a true software craftsman How to deal with conflict, tight schedules, and unreasonable managers How to get into the flow of coding and get past writer's block How to handle unrelenting pressure and avoid burnout How to combine enduring attitudes with new development paradigms How to manage your time and avoid blind alleys, marshes, bogs, and swamps How to foster environments where programmers and

teams can thrive When to say "No"--and how to say it When to say "Yes"--and what yes really means

Software Estimation Without Guessing - George Dinwiddie 2019-12-19

Estimating software development often produces more angst than value, but it doesn't have to. Identify the needs behind estimate requests and determine how to meet those needs simply and easily. Choose estimation techniques based on current needs and available information, gaining benefit while reducing cost and effort. Detect bad assumptions that might sink your project if you don't adjust your plans. Discover what to do when an estimate is wrong, how to recover, and how to use that knowledge for future planning. Learn to communicate about estimates in a healthy and productive way, maximizing advantage to the organization and minimizing damage to the people. In a world where most developers hate estimation and most managers fear disappointment with the results, there is hope for both. It requires giving up some widely held misconceptions. Let go of the notion that "an estimate is an estimate" and estimate for the particular need you, and your organization, have. Realize that estimates have a limited shelf-life, and reestimate frequently if it's important. When reality differs from your estimate, don't lament; mine that disappointment for the gold that can be the longer-term jackpot. Estimate in comparison to past experience, by modeling the work mathematically, or a hybrid of both. Learn strategies for effective decomposition of work and aspects of the work that likely affect your estimates. Hedge your bets by comparing the results of different approaches. Find out what to do when an estimate proves wrong. And they will. They're estimates, after all. You'll discover that you can use estimates to warn you of danger so you can take appropriate action in time. Learn some crucial techniques to understand and communicate with those who need to understand. Address both the technical and sociological aspects of estimation, and you'll help your organization achieve its desired goals with less drama and more benefit. What You Need: No software needed, just your past experience and concern for the outcomes. Programming Embedded Systems - Michael Barr 2006-10-11

Authored by two of the leading authorities in the field, this guide offers readers the knowledge and skills needed to achieve proficiency with embedded software.

Code Complete - Steve McConnell 2004-06-09

Widely considered one of the best practical guides to programming, Steve McConnell's original CODE COMPLETE has been helping developers write better software for more than a decade. Now this classic book has been fully updated and revised with leading-edge practices—and hundreds of new code samples—illustrating the art and science of software construction. Capturing the body of knowledge available from research, academia, and everyday commercial practice, McConnell synthesizes the most effective techniques and must-know principles into clear, pragmatic guidance. No matter what your experience level, development environment, or project size, this book will inform and stimulate your thinking—and help you build the highest quality code. Discover the timeless techniques and strategies that help you: Design for minimum complexity and maximum creativity Reap the benefits of collaborative development Apply defensive programming techniques to reduce and flush out errors Exploit opportunities to refactor—or evolve—code, and do it safely Use construction practices that are right-weight for your project Debug problems quickly and effectively Resolve critical construction issues early and correctly Build quality into the beginning, middle, and end of your project

Agile Project Management with Scrum - Ken Schwaber 2004-02-11

The rules and practices for Scrum—a simple process for managing complex projects—are few, straightforward, and easy to learn. But Scrum's simplicity itself—its lack of prescription—can be disarming, and new practitioners often find themselves reverting to old project management habits and tools and yielding lesser results. In this illuminating series of case studies, Scrum co-creator and evangelist Ken Schwaber identifies the real-world lessons—the successes and failures—culled from his years of experience coaching companies in agile project management. Through them, you'll understand how to use Scrum to solve complex problems and drive better results—delivering

more valuable software faster. Gain the foundation in Scrum theory—and practice—you need to: Rein in even the most complex, unwieldy projects Effectively manage unknown or changing product requirements Simplify the chain of command with self-managing development teams Receive clearer specifications—and feedback—from customers Greatly reduce project planning time and required tools Build—and release—products in 30-day cycles so clients get deliverables earlier Avoid missteps by regularly inspecting, reporting on, and fine-tuning projects Support multiple teams working on a large-scale project from many geographic locations Maximize return on investment!

Rapid Development - Steve McConnell 1996 Project managers, technical leads, and Windows programmers throughout the industry share an important concern--how to get their development schedules under control. Rapid Development addresses that concern head-on with philosophy, techniques, and tools that help shrink and control development schedules and keep projects moving. The style is friendly and conversational--and the content is impressive.

Adaptive Code - Gary McLean Hall 2017-04-18 Write code that can adapt to changes. By applying this book's principles, you can create code that accommodates new requirements and unforeseen scenarios without significant rewrites. Gary McLean Hall describes Agile best practices, principles, and patterns for designing and writing code that can evolve more quickly and easily, with fewer errors, because it doesn't impede change. Now revised, updated, and expanded, *Adaptive Code, Second Edition* adds indispensable practical insights on Kanban, dependency inversion, and creating reusable abstractions. Drawing on over a decade of Agile consulting and development experience, McLean Hall has updated his best-seller with deeper coverage of unit testing, refactoring, pure dependency injection, and more. Master powerful new ways to:

- Write code that enables and complements Scrum, Kanban, or any other Agile framework
- Develop code that can survive major changes in requirements
- Plan for adaptability by using dependencies, layering, interfaces, and design patterns
- Perform unit testing and refactoring in tandem, gaining more

value from both • Use the “golden master” technique to make legacy code adaptive • Build SOLID code with single-responsibility, open/closed, and Liskov substitution principles • Create smaller interfaces to support more-diverse client and architectural needs • Leverage dependency injection best practices to improve code adaptability • Apply dependency inversion with the Stairway pattern, and avoid related anti-patterns About You This book is for programmers of all skill levels seeking more-practical insight into design patterns, SOLID principles, unit testing, refactoring, and related topics. Most readers will have programmed in C#, Java, C++, or similar object-oriented languages, and will be familiar with core procedural programming techniques.

[Programming with the Kinect for Windows Software Development Kit](#) - David Catuhe
2012-09-15

Create rich experiences for users of Windows 7 and Windows 8 Developer Preview with this pragmatic guide to the Kinect for Windows Software Development Kit (SDK). The author, a developer evangelist for Microsoft, walks you through Kinect sensor technology and the SDK—providing hands-on insights for how to add gesture and posture recognition to your apps. If you’re skilled in C# and Windows Presentation Foundation, you’ll learn how to integrate Kinect in your applications and begin writing Uis and controls that can handle Kinect interaction. This book introduces the Kinect for Windows Software Development Kit to developers looking to enrich applications they build for Windows 7 and later with human motion tracking Teaches developers with core C# and WPF skills how to program gesture and posture recognition in Kinect Describes how to integrate 3D representation on top of a real scene Provides expert insights and code samples to get you up and running

Active Directory - Joe Richards 2006-01-19
Provides information on the features, functions, and implementation of Active Directory.

[The Art of Agile Development](#) - James Shore
2008

For those considering Extreme Programming, this book provides no-nonsense advice on agile planning, development, delivery, and management taken from the authors' many years

of experience. While plenty of books address the what and why of agile development, very few offer the information users can apply directly.

Agile Estimating and Planning - Mike Cohn
2005-11-01

Agile Estimating and Planning is the definitive, practical guide to estimating and planning agile projects. In this book, Agile Alliance cofounder Mike Cohn discusses the philosophy of agile estimating and planning and shows you exactly how to get the job done, with real-world examples and case studies. Concepts are clearly illustrated and readers are guided, step by step, toward how to answer the following questions: What will we build? How big will it be? When must it be done? How much can I really complete by then? You will first learn what makes a good plan—and then what makes it agile. Using the techniques in Agile Estimating and Planning, you can stay agile from start to finish, saving time, conserving resources, and accomplishing more. Highlights include: Why conventional prescriptive planning fails and why agile planning works How to estimate feature size using story points and ideal days—and when to use each How and when to re-estimate How to prioritize features using both financial and nonfinancial approaches How to split large features into smaller, more manageable ones How to plan iterations and predict your team's initial rate of progress How to schedule projects that have unusually high uncertainty or schedule-related risk How to estimate projects that will be worked on by multiple teams Agile Estimating and Planning supports any agile, semiagile, or iterative process, including Scrum, XP, Feature-Driven Development, Crystal, Adaptive Software Development, DSDM, Unified Process, and many more. It will be an indispensable resource for every development manager, team leader, and team member.

Software Requirement Patterns - Stephen Withall 2007-06-13

Learn proven, real-world techniques for specifying software requirements with this practical reference. It details 30 requirement “patterns” offering realistic examples for situation-specific guidance for building effective software requirements. Each pattern explains what a requirement needs to convey, offers potential questions to ask, points out potential

pitfalls, suggests extra requirements, and other advice. This book also provides guidance on how to write other kinds of information that belong in a requirements specification, such as assumptions, a glossary, and document history and references, and how to structure a requirements specification. A disturbing proportion of computer systems are judged to be inadequate; many are not even delivered; more are late or over budget. Studies consistently show one of the single biggest causes is poorly defined requirements: not properly defining what a system is for and what it's supposed to do. Even a modest contribution to improving requirements offers the prospect of saving businesses part of a large sum of wasted investment. This guide emphasizes this important requirement need—determining what a software system needs to do before spending time on development. Expertly written, this book details solutions that have worked in the past, with guidance for modifying patterns to fit individual needs—giving developers the valuable advice they need for building effective software requirements

Applying UML and Patterns Training Course - Craig Larman 2002-07-01

Second Edition of the UML video course based on the book Applying UML and Patterns. This VTC will focus on object-oriented analysis and design, not just drawing UML.

More Effective Agile - Steve McConnell 2019-08-24

In this comprehensive yet accessible overview for software leaders, the author presents an impactful, action-oriented prescription—covering the practical considerations needed to ensure you reap the full benefits of effective Agile

Emotionally Intelligent Design - Pamela Pavliscak 2018-11-21

As technology becomes deeply integrated into every aspect of our lives, we've begun to expect more emotionally intelligent interactions. But smartphones don't know if we're having a bad day, and cars couldn't care less about compassion. Technology is developing more IQ, but it still lacks EQ. In this book, Pamela Pavliscak—design researcher and advisor to Fortune 500 companies—explores new research about emotion, new technology that engages emotion, and new emotional design practices.

Drawing on her own research and the latest thinking in psychology, neuroscience, and behavioral economics, Pamela shows you how design can help promote emotional well-being. You'll learn: How design has transformed emotion and how tech is transforming it again New principles for merging emotional intelligence and design thinking How to use a relationship model for framing product interactions and personality Methods for blending well-being interventions with design patterns How emotional resonance can guide designers toward ethical futures Implications of emotionally intelligent technology as it scales from micro- to mega-emotional spheres Rapid Development - Steve McConnell 1996-07-02

Corporate and commercial software-development teams all want solutions for one important problem—how to get their high-pressure development schedules under control. In RAPID DEVELOPMENT, author Steve McConnell addresses that concern head-on with overall strategies, specific best practices, and valuable tips that help shrink and control development schedules and keep projects moving. Inside, you'll find: A rapid-development strategy that can be applied to any project and the best practices to make that strategy work Candid discussions of great and not-so-great rapid-development practices—estimation, prototyping, forced overtime, motivation, teamwork, rapid-development languages, risk management, and many others A list of classic mistakes to avoid for rapid-development projects, including creeping requirements, shortchanged quality, and silver-bullet syndrome Case studies that vividly illustrate what can go wrong, what can go right, and how to tell which direction your project is going RAPID DEVELOPMENT is the real-world guide to more efficient applications development.

Practical Project Initiation - Karl Wieggers 2007-08-08

Zero in on key project-initiation tasks—and build a solid foundation for successful software development. In this concise guide, critically-acclaimed author Karl E. Wieggers fills a void in project management literature by focusing on the activities that are essential—but often overlooked—for launching any project. Drawing

on his extensive experience, Karl shares lessons learned, proven practices, and tools for getting your project off to the right start—and steering it to ultimate success. Lay a foundation for project success—discover how to: Effectively charter a project Define meaningful criteria for project success and product releases Negotiate achievable commitments for project teams and stakeholders Identify and document potential barriers to success—and manage project risks Apply the Wideband Delphi method for more accurate estimation Measure project performance and avoid common metrics traps Systematically apply lessons learned to future projects Companion Web site includes: Worksheets from inside the book Project document templates Resources for project initiation and process improvement *Safe 5.0 Distilled: Achieving Business Agility with the Scaled Agile Framework* - RICHARD. LEFFINGWELL KNASTER (DEAN.) 2020-08-08

Operations Strategy - Nigel Slack 2008
Operation Strategy Second Edition Nigel Slack and Michael Lewis Ideal for Advanced Undergraduate and Postgraduate students, this book builds on concepts from Strategic Management, Operations Management, Marketing and HRM to give students a comprehensive understanding of Operations Strategy. Features Comprehensive and accessible with authoritative authorship and an excellent blend of theory and practice A European context Engaging case studies Teaching resources including an Instructor's Manual with extensive case notes and PowerPoint slides at www.pearsoned.co.uk/slack. What's New? This new edition has been focused to concentrate on the most significant topics in the subject, with 10 chapters replacing the previous 15. New material has been added and coverage of some older topics has been revised (see new table of contents). End-of-chapter case exercises have been replaced by a major end-of-book section of 'Harvard-type' cases. New to the Instructor's resources online: additional cases and a set of questions and answers for class use / exam use. New coverage of hot topics, such as the implications of ERP and Six Sigma on ops strategy, agility and it's inter-relationship with lean, supply management issues, operations

strategy for competitive advantage and SCM, and implementation.
Software Estimation Best Practices, Tools & Techniques - Murali Chemuturi 2009-07-15
Almost every software project begins with the utterances, "What will this cost?" and "When will this project be done?" Once those words are spoken, project stakeholders begin to wrestle with how to produce an estimate. Accurately estimating the cost or time to complete a software project is a serious problem for many software engineers, developers and project managers who struggle with costs running double original estimates, putting their careers at risk. It is reported that nearly 50% of all software projects are shelved and that one of the major causes is poor estimation practices. If developing software for internal use, poor estimates can represent a significant drain on corporate profits. Worldwide growth in the number of companies specializing in the development of software for use by other companies is staggering. India alone has nearly 20,000 such companies. Intense competition has led to an increased demand for fixed-bid pricing in client/vendor relationships, and has made effective cost estimation even more important and, in many cases, critical to a firm's survival. There are many methods of estimation. Each method has its strengths and weaknesses, proponents and opponents. Knowing how and which one to use on a given project is key to developing acceptable estimates for either internal or external projects. Software Estimation Best Practices, Tools, & Techniques covers all facets of software estimation. It provides a detailed explanation of the various methods for estimating software size, development effort, cost, and schedule, including a comprehensive explanation of Test Effort Estimation. Emphasizing that software estimation should be based on a well-defined process, it presents software estimation best practices and shows how to avoid common pitfalls. This guide offers direction on which methods are most appropriate for each of the different project types commonly executed in the software development space and criteria for selecting software estimation tools. This comprehensive desk reference explains software estimation from scratch to help the beginner and features

advanced techniques for more experienced estimators. It details project scheduling, including resource leveling and the concept of productivity, as applicable to software estimators, demonstrating the many benefits of moving from the current macro-productivity approach to a micro-productivity approach in software estimation. *Software Estimation Best Practices, Tools, & Techniques: A Complete Guide for Software Project Estimators* caters to the needs of all software project stakeholders, from novice to expert. It provides the valuable guidance needed to estimate the cost and time required to complete software projects within a reasonable margin of error for effective software development.

Technology Strategy Patterns - Eben Hewitt
2018-10-15

Technologists who want their ideas heard, understood, and funded are often told to speak the language of business—without really knowing what that is. This book's toolkit provides architects, product managers, technology managers, and executives with a shared language—in the form of repeatable, practical patterns and templates—to produce great technology strategies. Author Eben Hewitt developed 39 patterns over the course of a decade in his work as CTO, CIO, and chief architect for several global tech companies. With these proven tools, you can define, create, elaborate, refine, and communicate your architecture goals, plans, and approach in a way that executives can readily understand, approve, and execute. This book covers: Architecture and strategy: Adopt a strategic architectural mindset to make a meaningful material impact Creating your strategy: Define the components of your technology strategy using proven patterns Communicating the strategy: Convey your technology strategy in a compelling way to a variety of audiences Bringing it all together: Employ patterns individually or in clusters for specific problems; use the complete framework for a comprehensive strategy

Applied Software Measurement - Capers Jones
2008-05-11

Effectively forecast, manage, and control software across the entire project lifecycle Accurately size, estimate, and administer software projects with real-world guidance from

an industry expert. Fully updated to cover the latest tools and techniques, *Applied Software Measurement, Third Edition* details how to deploy a cost-effective and pragmatic analysis strategy. You will learn how to use function points and baselines, implement benchmarks and tracking systems, and perform efficiency tests. Full coverage of the latest regulations, metrics, and standards is included. Measure performance at the requirements, coding, testing, and installation phases Set function points for efficiency, cost, market share, and customer satisfaction Analyze quality and productivity using assessments, benchmarks, and baselines Design and manage project cost, defect, and quality tracking systems Use object-oriented, reusable component, Agile, CMM, and XP methods Assess defect removal efficiency using unit tests and multistage test suites
Software Estimation - Steve McConnell
2006-02-22

Often referred to as the “black art” because of its complexity and uncertainty, software estimation is not as difficult or puzzling as people think. In fact, generating accurate estimates is straightforward—once you understand the art of creating them. In his highly anticipated book, acclaimed author Steve McConnell unravels the mystery to successful software estimation—distilling academic information and real-world experience into a practical guide for working software professionals. Instead of arcane treatises and rigid modeling techniques, this guide highlights a proven set of procedures, understandable formulas, and heuristics that individuals and development teams can apply to their projects to help achieve estimation proficiency. Discover how to: Estimate schedule and cost—or estimate the functionality that can be delivered within a given time frame Avoid common software estimation mistakes Learn estimation techniques for you, your team, and your organization * Estimate specific project activities—including development, management, and defect correction Apply estimation approaches to any type of project—small or large, agile or traditional Navigate the shark-infested political waters that surround project estimates When many corporate software projects are failing, McConnell shows you what works for successful

software estimation.

How to Get Prepared for Managing a Remote Team - Hugo Messer 2014

The main focus of this eBook is "How to get prepared for managing a remote team." Typical questions that come up while preparing are: Which country shall we outsource our work to?; What project shall we choose to start with?; Which company suits our needs best?; Shall we set up our own captive center or outsource to a partner?; Are we organized well enough to start offshoring work? Most people tend to focus a lot on these 'initiation' questions at the expense of wondering 'how to organize'. Preparation is seen as selecting the right country and partner and then 'just get going'. Many problems can be prevented by investing time in the right organization before the 'real work' starts. In this eBook, we try to provide advice on both perspectives, based on experiences from several experts around the globe. The first chapter is written by Hugo Messer, he describes how to get started. The main questions he answered in this chapter are related to 'initiation' and the questions above. Hugo has gained substantial experience in setting up and managing remote teams, with suppliers, freelancers and own offices. Then, Patrick van Dun, an experienced 'offshore founder', provides guidelines on the choice of setting up your own remote office versus engaging a partner. Zhenya Rozinskiy discusses his best practices for getting the right people on your team. Zhenya has set up several teams around the world. Amanda Crouch from the UK has over 20 years of experience as a management consultant and researcher. She looks at stimulating collaboration at the company and individual level. Ove Holmberg, an IT project manager and agile coach from Sweden, describes his concept of the virtual teamroom. Andreas Brilling from Germany works as engagement manager for CapGemini and has led a large offshoring initiative from Australia. In the final chapter Hugo shares his personal story of how he got started with setting up his own offices in India and Ukraine. This is the second eBook in a series of eBooks that will be published within a couple of month's interval and later on into one printed book. These eBooks are being written through a crowdwriting project and the authors are experts from all over

the world.

Why Programs Fail - Andreas Zeller 2009-06-12
This fully updated second edition includes 100+ pages of new material, including new chapters on Verifying Code, Predicting Errors, and Preventing Errors. Cutting-edge tools such as FindBUGS and AGITAR are explained, techniques from integrated environments like Jazz.net are highlighted, and all-new demos with ESC/Java and Spec#, Eclipse and Mozilla are included. This complete and pragmatic overview of debugging is authored by Andreas Zeller, the talented researcher who developed the GNU Data Display Debugger(DDD), a tool that over 250,000 professionals use to visualize the data structures of programs while they are running. Unlike other books on debugging, Zeller's text is product agnostic, appropriate for all programming languages and skill levels. *Why Programs Fail* explains best practices ranging from systematically tracking error reports, to observing symptoms, reproducing errors, and correcting defects. It covers a wide range of tools and techniques from hands-on observation to fully automated diagnoses, and also explores the author's innovative techniques for isolating minimal input to reproduce an error and for tracking cause and effect through a program. It even includes instructions on how to create automated debugging tools. The new edition of this award-winning productivity-booster is for any developer who has ever been frustrated by elusive bugs. Brand new chapters demonstrate cutting-edge debugging techniques and tools, enabling readers to put the latest time-saving developments to work for them. Learn by doing. New exercises and detailed examples focus on emerging tools, languages and environments, including AGITAR, FindBUGS, Python and Eclipse. The text includes exercises and extensive references for further study, and a companion website with source code for all examples and additional debugging resources.

Pro Git - Scott Chacon 2009-10-06

Git is the version control system developed by Linus Torvalds for Linux kernel development. It took the open source world by storm since its inception in 2005, and is used by small development shops and giants like Google, Red Hat, and IBM, and of course many open source projects. A book by Git experts to turn you into a

Git expert Introduces the world of distributed version control Shows how to build a Git development workflow

Software Project Survival Guide - Steve McConnell 1998

Looks at a successful software project and provides details for software development for clients using object-oriented design and programming.

The Economics of Software Quality - Capers Jones 2012

Poor quality continues to bedevil large-scale development projects, but few software leaders and practitioners know how to measure quality, select quality best practices, or cost-justify their usage. In *The Economics of Software Quality*, leading software quality experts Capers Jones and Jitendra Subramanyam show how to systematically measure the economic impact of quality and how to use this information to deliver far more business value. Using empirical data from hundreds of software organizations, Jones and Subramanyam show how integrated inspection, static analysis, and testing can achieve defect removal rates exceeding 95 percent. They offer innovative guidance for predicting and measuring defects and quality; choosing defect prevention, pre-test defect removal, and testing methods; and optimizing post-release defect reporting and repair. This book will help you Prove that improved software quality translates into strongly positive ROI and greatly reduced TCO Drive better results from current investments in debugging and prevention Use quality techniques to stay on schedule and on budget Avoid "hazardous" metrics that lead to poor decisions Important note: The audio and video content included with this enhanced eBook can be viewed only using iBooks on an iPad, iPhone, or iPod touch.

Applied Software Project Management - Andrew Stellman 2005-11-18

"If you're looking for solid, easy-to-follow advice on estimation, requirements gathering, managing change, and more, you can stop now: this is the book for you."--Scott Berkun, Author of *The Art of Project Management* What makes software projects succeed? It takes more than a good idea and a team of talented programmers. A project manager needs to know how to guide the team through the entire software project.

There are common pitfalls that plague all software projects and rookie mistakes that are made repeatedly--sometimes by the same people! Avoiding these pitfalls is not hard, but it is not necessarily intuitive. Luckily, there are tried and true techniques that can help any project manager. In *Applied Software Project Management*, Andrew Stellman and Jennifer Greene provide you with tools, techniques, and practices that you can use on your own projects right away. This book supplies you with the information you need to diagnose your team's situation and presents practical advice to help you achieve your goal of building better software. Topics include: Planning a software project Helping a team estimate its workload Building a schedule Gathering software requirements and creating use cases Improving programming with refactoring, unit testing, and version control Managing an outsourced project Testing software Jennifer Greene and Andrew Stellman have been building software together since 1998. Andrew comes from a programming background and has managed teams of requirements analysts, designers, and developers. Jennifer has a testing background and has managed teams of architects, developers, and testers. She has led multiple large-scale outsourced projects. Between the two of them, they have managed every aspect of software development. They have worked in a wide range of industries, including finance, telecommunications, media, nonprofit, entertainment, natural-language processing, science, and academia. For more information about them and this book, visit stellman-greene.com

Software Cost Estimation, Benchmarking, and Risk Assessment - Adam Trendowicz 2013-01-10

Software effort estimation is a key element of software project planning and management. Yet, in industrial practice, the important role of effort estimation is often underestimated and/or misunderstood. In this book, Adam Trendowicz presents the CoBRA method (an abbreviation for Cost Estimation, Benchmarking, and Risk Assessment) for estimating the effort required to successfully complete a software development project, which uniquely combines human judgment and measurement data in order to

systematically create a custom-specific effort estimation model. CoBRA goes far beyond simply predicting the development effort; it supports project decision-makers in negotiating the project scope, managing project risks, benchmarking productivity, and directing improvement activities. To illustrate the method's practical use, the book reports several real-world cases where CoBRA was applied in various industrial contexts. These cases represent different estimation contexts in terms of software project environment, estimation objectives, and estimation constraints. This book is the result of a successful collaboration between the process management division of Fraunhofer IESE and many software companies in the field of software engineering technology transfer. It mainly addresses software practitioners who deal with planning and managing software development projects as part of their daily work, and is also of interest for students or courses specializing in software engineering or software project management.

Estimating Software Costs - Capers Jones
2007-05-10

Deliver bug-free software projects on schedule and within budget Get a clear, complete understanding of how to estimate software costs, schedules, and quality using the real-world information contained in this comprehensive volume. Find out how to choose the correct hardware and software tools, develop an appraisal strategy, deploy tests and prototypes, and produce accurate software cost estimates. Plus, you'll get full coverage of cutting-edge estimating approaches using Java, object-oriented methods, and reusable components. Plan for and execute project-, phase-, and activity-level cost estimations Estimate regression, component, integration, and stress tests Compensate for inaccuracies in data collection, calculation, and analysis Assess software deliverables and data complexity Test design principles and operational characteristics using software prototyping Handle configuration change, research, quality control, and documentation costs "Capers Jones' work offers a unique contribution to the understanding of the economics of software production. It provides deep insights into why our advances in computing are not matched with corresponding

improvements in the software that drives it. This book is absolutely required reading for an understanding of the limitations of our technological advances." --Paul A. Strassmann, former CIO of Xerox, the Department of Defense, and NASA

Trends in Advanced Intelligent Control, Optimization and Automation - Wojciech Mitkowski 2017-06-06

This volume contains the proceedings of the KKA 2017 - the 19th Polish Control Conference, organized by the Department of Automatics and Biomedical Engineering, AGH University of Science and Technology in Kraków, Poland on June 18-21, 2017, under the auspices of the Committee on Automatic Control and Robotics of the Polish Academy of Sciences, and the Commission for Engineering Sciences of the Polish Academy of Arts and Sciences. Part 1 deals with general issues of modeling and control, notably flow modeling and control, sliding mode, predictive, dual, etc. control. In turn, Part 2 focuses on optimization, estimation and prediction for control. Part 3 is concerned with autonomous vehicles, while Part 4 addresses applications. Part 5 discusses computer methods in control, and Part 6 examines fractional order calculus in the modeling and control of dynamic systems. Part 7 focuses on modern robotics. Part 8 deals with modeling and identification, while Part 9 deals with problems related to security, fault detection and diagnostics. Part 10 explores intelligent systems in automatic control, and Part 11 discusses the use of control tools and techniques in biomedical engineering. Lastly, Part 12 considers engineering education and teaching with regard to automatic control and robotics. [Benefit/Cost-Driven Software Development](#) - Jo Erskine Hannay 2021-07-26

This open access book presents a set of basic techniques for estimating the benefit of IT development projects and portfolios. It also offers methods for monitoring how much of that estimated benefit is being achieved during projects. Readers can then use these benefit estimates together with cost estimates to create a benefit/cost index to help them decide which functionalities to send into construction and in what order. This allows them to focus on constructing the functionality that offers the

best value for money at an early stage. Although benefits management involves a wide range of activities in addition to estimation and monitoring, the techniques in this book provides a clear guide to achieving what has always been the goal of project and portfolio stakeholders: developing systems that produce as much usefulness and value as possible for the money invested. The techniques can also help deal with vicarious motives and obstacles that prevent this happening. The book equips readers to recognize when a project budget should not be spent in full and resources be allocated elsewhere in a portfolio instead. It also provides development managers and upper management with common ground as a basis for making informed decisions.

Coder to Developer - Mike Gunderloy
2006-02-20

"Two thumbs up" —Gregory V. Wilson, Dr. Dobbs Journal (October 2004) No one can disparage the ability to write good code. At its highest levels, it is an art. But no one can confuse writing good code with developing good software. The difference—in terms of challenges, skills, and compensation—is immense. Coder to Developer helps you excel at the many non-coding tasks entailed, from start to finish, in just about any successful development project. What's more, it equips you with the mindset and self-assurance required to pull it all together, so that you see every piece of your work as part of a coherent process. Inside, you'll find plenty of technical guidance on such topics as: Choosing and using a source code control system Code generation tools—when and why Preventing bugs with unit testing Tracking, fixing, and learning from bugs Application activity logging Streamlining and systematizing the build process Traditional installations and alternative approaches To pull all of this together, the author has provided the source code for Download Tracker, a tool for organizing your collection of downloaded code, that's used for examples throughout this book. The code is provided in various states of completion, reflecting every stage of development, so that you can dig deep into the actual process of building software. But you'll also develop "softer" skills, in areas such as team management, open source collaboration, user

and developer documentation, and intellectual property protection. If you want to become someone who can deliver not just good code but also a good product, this book is the place to start. If you must build successful software projects, it's essential reading.

The Pragmatic Programmer - David Thomas
2019-07-30

"One of the most significant books in my life."
—Obie Fernandez, Author, The Rails Way
"Twenty years ago, the first edition of The Pragmatic Programmer completely changed the trajectory of my career. This new edition could do the same for yours." —Mike Cohn, Author of Succeeding with Agile, Agile Estimating and Planning, and User Stories Applied ". . . filled with practical advice, both technical and professional, that will serve you and your projects well for years to come." —Andrea Goulet, CEO, Corgibytes, Founder, LegacyCode.Rocks ". . . lightning does strike twice, and this book is proof." —VM (Vicky) Brasseur, Director of Open Source Strategy, Juniper Networks
The Pragmatic Programmer is one of those rare tech books you'll read, re-read, and read again over the years. Whether you're new to the field or an experienced practitioner, you'll come away with fresh insights each and every time. Dave Thomas and Andy Hunt wrote the first edition of this influential book in 1999 to help their clients create better software and rediscover the joy of coding. These lessons have helped a generation of programmers examine the very essence of software development, independent of any particular language, framework, or methodology, and the Pragmatic philosophy has spawned hundreds of books, screencasts, and audio books, as well as thousands of careers and success stories. Now, twenty years later, this new edition re-examines what it means to be a modern programmer. Topics range from personal responsibility and career development to architectural techniques for keeping your code flexible and easy to adapt and reuse. Read this book, and you'll learn how to: Fight software rot Learn continuously Avoid the trap of duplicating knowledge Write flexible, dynamic, and adaptable code Harness the power of basic tools Avoid programming by coincidence Learn real requirements Solve the underlying problems of concurrent code Guard against security

vulnerabilities Build teams of Pragmatic Programmers Take responsibility for your work and career Test ruthlessly and effectively, including property-based testing Implement the Pragmatic Starter Kit Delight your users Written as a series of self-contained sections and filled with classic and fresh anecdotes, thoughtful examples, and interesting analogies, The Pragmatic Programmer illustrates the best approaches and major pitfalls of many different aspects of software development. Whether you're a new coder, an experienced

programmer, or a manager responsible for software projects, use these lessons daily, and you'll quickly see improvements in personal productivity, accuracy, and job satisfaction. You'll learn skills and develop habits and attitudes that form the foundation for long-term success in your career. You'll become a Pragmatic Programmer. Register your book for convenient access to downloads, updates, and/or corrections as they become available. See inside book for details.