

Software Project Survival Developer Best Practices

Publisher Fact Sheet A concise, hands-on approach to managing & improving the critical requirements process in software development.

An ethologist shows man to be a gene machine whose world is one of savage competition and deceit

Learn how to build dynamic web applications with Express, a key component of the Node/JavaScript development stack. In this hands-on guide, author Ethan Brown teaches you the fundamentals through the development of a fictional application that exposes a public website and a RESTful API. You'll also learn web architecture best practices to help you build single-page, multi-page, and hybrid web apps with Express. Express strikes a balance between a robust framework and no framework at all, allowing you a free hand in your architecture choices. With this book, frontend and backend engineers familiar with JavaScript will discover new ways of looking at web development. Create webpage templating system for rendering dynamic data Dive into request and response objects, middleware, and URL routing Simulate a production environment for testing and development Focus on persistence with document databases, particularly MongoDB Make your resources available to other programs with RESTful APIs Build secure apps with authentication, authorization, and HTTPS Integrate with social media, geolocation, and other third-party services Implement a plan for launching and maintaining your app Learn critical debugging skills This book covers Express 4.0.

The orderly Sweet-Williams are dismayed at their son's fondness for the messy pastime of gardening.

Dynamics of Software Development

Docs for Developers

The Selfish Gene

The Future of Leadership Development

The Pragmatic Programmer

Navigate Your Workplace, Cube Farm, or Startup

Clinical Research Computing

In a book that will intrigue anyone who is curious about Silicon Valley, computer programming, or the world of high technology, respected software pioneer and computer scientist Richard Gabriel offers an informative insider's look at the world of software design and computer programming and the business that surrounds them. 10 illustrations. A new edition of the most popular book of project management case studies, expanded to include more than 100 cases plus a "super case" on the Iridium Project Case studies are an important part of project management education and training. This Fourth Edition of Harold Kerzner's Project Management Case Studies features a number of new cases covering value measurement in project management. Also included is the well-received "super case," which covers all aspects of project management and may be used as a capstone for a

course. This new edition: Contains 100-plus case studies drawn from real companies to illustrate both successful and poor implementation of project management Represents a wide range of industries, including medical and pharmaceutical, aerospace, manufacturing, automotive, finance and banking, and telecommunications Covers cutting-edge areas of construction and international project management plus a "super case" on the Iridium Project, covering all aspects of project management Follows and supports preparation for the Project Management Professional (PMP®) Certification Exam Project Management Case Studies, Fourth Edition is a valuable resource for students, as well as practicing engineers and managers, and can be used on its own or with the new Eleventh Edition of Harold Kerzner's landmark reference, Project Management: A Systems Approach to Planning, Scheduling, and Controlling. (PMP and Project Management Professional are registered marks of the Project Management Institute, Inc.)

An exploration of how design might be led by marginalized communities, dismantle structural inequality, and advance collective liberation and ecological survival. What is the relationship between design, power, and social justice? "Design justice" is an approach to design that is led by marginalized communities and that aims explicitly to challenge, rather than reproduce, structural inequalities. It has emerged from a growing community of designers in various fields who work closely with social movements and community-based organizations around the world. This book explores the theory and practice of design justice, demonstrates how universalist design principles and practices erase certain groups of people—specifically, those who are intersectionally disadvantaged or multiply burdened under the matrix of domination (white supremacist heteropatriarchy, ableism, capitalism, and settler colonialism)—and invites readers to "build a better world, a world where many worlds fit; linked worlds of collective liberation and ecological sustainability." Along the way, the book documents a multitude of real-world community-led design practices, each grounded in a particular social movement. Design Justice goes beyond recent calls for design for good, user-centered design, and employment diversity in the technology and design professions; it connects design to larger struggles for collective liberation and ecological survival.

Once upon a time Linus Torvalds was a skinny unknown, just another nerdy Helsinki techie who had been fooling around with computers since childhood. Then he wrote a groundbreaking operating system and distributed it via the Internet -- for free. Today Torvalds is an international folk hero. And his creation LINUX is used by over 12 million people as well as by companies such as IBM. Now, in a narrative that zips along with the speed of e-mail, Torvalds gives a history of his renegade software while candidly revealing the quirky mind of a genius. The result is an engrossing portrayal of a man with a revolutionary vision, who challenges our values and may change our world.

Community-Led Practices to Build the Worlds We Need

Just for Fun

Shorter Schedules, Higher Quality Products, More Successful Projects, Enhanced Careers

The Mythical Man-month

An Engineer's Field Guide to Technical Writing

Building Evolutionary Architectures

Software Estimation

This practical handbook on software project success and survival explains how to confront five important issues involved in all software projects--people, politics, process, project management, and tools.

The book is about Software Quality Engineering with basic concepts, self-review, interviews preparation for java based projects test automation in a practical sense with questions and answers mode. There are about 500+ questions and answers to ease on understanding the concepts and review purpose. There are 15 core skills covered in this book as listed below. 1. Software Development Life Cycle (SDLC), 2. Software Quality Concepts, 3. OOPS, 4. XML, 5. XPath, 6. SCM/SCCS (SVN/GIT), 7. Unix/Linux, 8. Java & JDBC, 9. ANT, 10. Maven, 11. JUnit, 12. TestNG, 13. Jenkins/Hudson (CI), 14. Web Applications Testing - Selenium, 15. Web Services - SOAP/REST API. This book is aimed at beginners to the software quality and also useful for experienced quality engineers to assess and be on top of relevant skills. Here the author is considering "Quality Assurance" and "Quality Engineering" as same to carry out the similar effort except that to stress the importance of applying the Engineering principles rather than simply repeating the assurance test actions. This book should help in making sure that you get the

basic core concepts, working knowledge and in summary as a survival guide for programming and automation with all required skills. The goal is not to aim at making you an expert at one skill or entirely on these skills. For the Manual QA engineer, this book helps in understanding quality concepts, SDLC (Software Development Life Cycle), technical terminology, etc. Also, this helps in moving from manual to automation engineer. It is also useful for Developers working on Java projects because Java programming, unit testing and most of the other skills are in common with QA automation. Also, it gives understanding some of the test frameworks and terminologies in the test development. Finally, this book is an attempt to share and build confidence in core skills for Software quality engineering.

How to rewire your brain to improve virtually every aspect of your life—based on the latest research in neuroscience and psychology on neuroplasticity and evidence-based practices Not long ago, it was thought that the brain you were born with was the brain you would die with, and that the brain cells you had at birth were the most you would ever possess. Your brain was thought to be “hardwired” to function in predetermined ways. It turns out that's not true. Your brain is not hardwired, it's "softwired" by experience. This book shows you how you can rewire parts of the brain to feel more positive about your life, remain calm during stressful times, and improve your social relationships. Written by a leader in the field of Brain-Based Therapy, it teaches you how to activate the parts of your brain that have been underactivated and calm down those areas that have been hyperactivated so that you feel positive about your life and remain calm during stressful times. You will also learn to improve your memory, boost your mood, have better relationships, and get a good night sleep. Reveals how cutting-edge developments in neuroscience, and evidence-based practices can be used to improve your everyday life Other titles by Dr. Arden include: Brain-Based Therapy-Adult, Brain-Based Therapy-Child, Improving Your Memory For Dummies and Heal Your Anxiety Workbook Dr. Arden is a leader in integrating the new developments in neuroscience with psychotherapy and Director of Training in Mental Health for Kaiser Permanente for the Northern California Region Explaining exciting new developments in neuroscience and their applications to daily living, Rewire Your Brain will guide you through the process of changing your brain so you can change your life and be free of self-imposed limitations.

The author explains how he organized and supervised effective software development teams at the Microsoft company to come up with timely and high-quality commercial applications, offering a

candid look at the group dynamics of software development. Original. (Advanced).

The Business Value of Software

Leveraging the JavaScript Stack

Patterns of Software

Concepts, Principles, and Practices

New Programmer's Survival Manual

The Complete Software Developer's Career Guide

Rapid Development

The software development ecosystem is constantly changing, providing a constant stream of new tools, frameworks, techniques, and paradigms. Over the past few years, incremental developments in core engineering practices for software development have created the foundations for rethinking how architecture changes over time, along with ways to protect important architectural characteristics as it evolves. This practical guide ties those parts together with a new way to think about architecture and time.

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. Now in the 5th edition, **Cracking the Coding Interview** gives you the interview preparation you need to get the top software developer jobs. This book provides: 150 Programming Interview Questions and Solutions: From binary trees to binary search, this list of 150 questions includes the most common and most useful questions in data structures, algorithms, and knowledge based questions. 5 Algorithm Approaches: Stop being blind-sided by tough algorithm questions, and learn these five approaches to tackle the trickiest problems. Behind the Scenes of the interview processes at Google, Amazon, Microsoft, Facebook, Yahoo, and Apple: Learn what really goes on during your interview day and how decisions get made. Ten Mistakes Candidates Make -- And How to Avoid Them: Don't lose your dream job by making these common mistakes. Learn what many candidates do wrong, and how to avoid these issues. Steps to Prepare for Behavioral and Technical Questions: Stop meandering through an endless set of questions, while missing some of the most important preparation techniques. Follow these steps to more thoroughly prepare in less time.

In times of constant change, adaptive leadership is critical. This Harvard Business Review collection brings together the seminal ideas on how to adapt and thrive in challenging environments, from leading thinkers on the topic—most notably Ronald A. Heifetz of the Harvard Kennedy

School and Cambridge Leadership Associates. The Heifetz Collection includes two classic books: *Leadership on the Line*, by Ron Heifetz and Marty Linsky, and *The Practice of Adaptive Leadership*, by Heifetz, Linsky, and Alexander Grashow. Also included is the popular Harvard Business Review article, "Leadership in a (Permanent) Crisis," written by all three authors. Available together for the first time, this collection includes full digital editions of each work. Adaptive leadership is a practical framework for dealing with today's mix of urgency, high stakes, and uncertainty. It has been used by individuals, organizations, businesses, and governments worldwide. In a world of challenging environments, adaptive leadership serves as a guide to distinguishing the essential from the expendable, beginning the meaningful process of adaptation, and changing the status quo. Ronald A. Heifetz is a cofounder of the international leadership and consulting practice Cambridge Leadership Associates (CLA) and the founding director of the Center for Public Leadership at the Harvard Kennedy School. He is renowned worldwide for his innovative work on the practice and teaching of leadership. Marty Linsky is a cofounder of CLA and has taught at the Kennedy School for more than twenty-five years. Alexander Grashow is a Senior Advisor to CLA, having previously held the position of CEO.

Adaptive Leadership: The Heifetz Collection (3 Items)

Why Some Companies Make the Leap...And Others Don't

Demystifying the Black Art

Tales from the Software Community

A Hands-On Guide

Essays on Software Engineering

Good to Great

Equip yourself with SOFTWARE PROJECT SURVIVAL GUIDE. It's for everyone with a stake in the outcome of a development project--and especially for those without formal software project management training. That includes top managers, executives, clients, investors, end-user representatives, project managers, and technical leads. Here you'll find guidance from the acclaimed author of the classics CODE COMPLETE and RAPID DEVELOPMENT. Steve McConnell draws on solid research and a career's worth of hard-won experience to map the surest path to your goal--what he calls "one specific approach to software development that works pretty well most of the time for most projects." Nineteen chapters in four sections cover the concepts and strategies you need for mastering the development process, including planning, design, management, quality assurance, testing, and archiving. For newcomers and seasoned project managers alike, SOFTWARE PROJECT SURVIVAL GUIDE draws on a vast store of techniques to create an elegantly simplified and reliable framework for project management success. So don't worry about wandering among complex sets of project management techniques that require years to sort out and master. SOFTWARE PROJECT SURVIVAL GUIDE goes straight to the heart of the matter to help your projects succeed. And that makes it a required addition to every professional's bookshelf.

The Challenge Built to Last, the defining management study of the nineties, showed how great companies triumph over time and how long-term sustained performance can be engineered into the DNA of an enterprise from the very beginning. But what

about the company that is not born with great DNA? How can good companies, mediocre companies, even bad companies achieve enduring greatness? The Study For years, this question preyed on the mind of Jim Collins. Are there companies that defy gravity and convert long-term mediocrity or worse into long-term superiority? And if so, what are the universal distinguishing characteristics that cause a company to go from good to great? The Standards Using tough benchmarks, Collins and his research team identified a set of elite companies that made the leap to great results and sustained those results for at least fifteen years. How great? After the leap, the good-to-great companies generated cumulative stock returns that beat the general stock market by an average of seven times in fifteen years, better than twice the results delivered by a composite index of the world's greatest companies, including Coca-Cola, Intel, General Electric, and Merck. The Comparisons The research team contrasted the good-to-great companies with a carefully selected set of comparison companies that failed to make the leap from good to great. What was different? Why did one set of companies become truly great performers while the other set remained only good? Over five years, the team analyzed the histories of all twenty-eight companies in the study. After sifting through mountains of data and thousands of pages of interviews, Collins and his crew discovered the key determinants of greatness -- why some companies make the leap and others don't. The Findings The findings of the Good to Great study will surprise many readers and shed light on virtually every area of management strategy and practice. The findings include: Level 5 Leaders: The research team was shocked to discover the type of leadership required to achieve greatness. The Hedgehog Concept (Simplicity within the Three Circles): To go from good to great requires transcending the curse of competence. A Culture of Discipline: When you combine a culture of discipline with an ethic of entrepreneurship, you get the magical alchemy of great results. Technology Accelerators: Good-to-great companies think differently about the role of technology. The Flywheel and the Doom Loop: Those who launch radical change programs and wrenching restructurings will almost certainly fail to make the leap. "Some of the key concepts discerned in the study," comments Jim Collins, "fly in the face of our modern business culture and will, quite frankly, upset some people." Perhaps, but who can afford to ignore these findings?

In business, driving value is a key strategy and typically starts at the top of an organization. In today's digital age, driving software value is also an important, and often overlooked, key strategy. Executives, and the corporate board, need to expect the highest level of business value from the software the organization is developing, buying, and selling. In today's digital transformation marketplace, it is imperative that organizations start driving business value from software development initiatives. For many years, the cost of software development challenged organizations with questions such as: How do we allocate software development costs? Should these costs be considered an overhead expense? Are we getting the most value possible for our investment? A fundamental problem has been built into these questions - the focus on cost. In almost every other part of the organization, maximizing profit or, in the case of a not-for-profit, maximizing the funds available, provides a clear focus with metrics to determine success or failure. In theory, simply aligning software spending with the maximizing profit goals should be sufficient to avoid any questions about value for money. Unfortunately, this alignment hasn't turned out

to be so simple, and the questions persist, particularly at the strategic or application portfolio level. In this book, Michael D.S. Harris describes how a software business value culture—one where all stakeholders, including technology and business—have a clear understanding of the goals and expected business value from software development. The book shows readers how they can transform software development from a cost or profit center to a business value center. Only a culture of software as a value center enables an organization to constantly maximize business value flow through software development. If your organization is starting to ask how it can change software from a cost-center to a value-center, this book is for you.

It's your first day on the new job. You've got the programming chops, you're up on the latest tech, you're sitting at your workstation... now what? *New Programmer's Survival Manual* gives your career the jolt it needs to get going: essential industry skills to help you apply your raw programming talent and make a name for yourself. It's a no-holds-barred look at what really goes on in the office--and how to not only survive, but thrive in your first job and beyond. Programming at industry level requires new skills - you'll build programs that dwarf anything you've done on your own. This book introduces you to practices for working on large-scale, long-lived programs at a professional level of quality. You'll find out how to work efficiently with your current tools, and discover essential new tools. But the tools are only part of the story; you've got to get street-smart too. Succeeding in the corporate working environment requires its own savvy. You'll learn how to navigate the office, work with your teammates, and how to deal with other people outside of your department. You'll understand where you fit into the big picture and how you contribute to the company's success. You'll also get a candid look at the tougher aspects of the job: stress, conflict, and office politics. Finally, programming is a job you can do for the long haul. This book helps you look ahead to the years to come, and your future opportunities--either as a programmer or in another role you grow into. There's nothing quite like the satisfaction of shipping a product and knowing, "I built that." Whether you work on embedded systems or web-based applications, in trendy technologies or legacy systems, this book helps you get from raw skill to an accomplished professional.

Case Studies

your journey to mastery, 20th Anniversary Edition

Debugging the Development Process

The Story of an Accidental Revolutionary

Don't Hire a Software Developer Until You Read This Book

Code Complete, 2nd Edition

Design Justice

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. 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.

*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.*

First Published in 2003. Routledge is an imprint of Taylor & Francis, an informa company.

Applied Survival Analysis

150 Programming Interview Questions and Solutions

Software Project Survival Guide

Web Development with Node and Express

With C and GNU Development Tools

Rewire Your Brain

System Engineering Analysis, Design, and Development

A software survival guide for non-technical entrepreneurs entering the tech space who want to reduce the uncertainty associated to starting their business, and for seed startups who require support and ideas when dealing with the daily realities of managing the software development process and getting a quality software application built and launched.

Provides a variety of ideas, techniques, and strategies for effective software development.

bull; Renowned software expert Steve McConnell presents his latest thoughts on the condition of the software engineering profession bull; Helps software developers regain the sight of the big-picture reasons why their jobs matter bull; A thinking man's guide to the current state of software

THE MOST PRACTICAL, UP-TO-DATE GUIDE TO MODELLING AND ANALYZING TIME-TO-EVENT DATA—NOW IN A VALUABLE NEW EDITION Since publication of the first edition nearly a decade ago, analyses using time-to-event methods have increase considerably in all areas of scientific inquiry mainly as a result of model-building methods available in modern statistical software packages. However, there has been minimal coverage in the available literature to9 guide researchers, practitioners, and students who wish to apply these methods to health-related areas of study. **Applied Survival Analysis, Second Edition** provides a comprehensive and up-to-date introduction to regression modeling for time-to-event data in medical, epidemiological, biostatistical, and other health-related research. This book places a unique emphasis on the practical and contemporary applications of regression modeling rather than the mathematical theory. It offers a clear and accessible presentation of modern modeling techniques supplemented with real-world examples and case studies. Key topics covered include: variable selection, identification of the scale of continuous covariates, the role of interactions in the model, assessment of fit and model assumptions, regression diagnostics, recurrent event models, frailty models, additive models, competing risk models, and missing data. Features of the Second Edition include: Expanded coverage of interactions and the covariate-adjusted survival functions The use of the Worcester Heart Attack Study as the main modeling data set for illustrating discussed concepts and techniques New discussion of variable selection with multivariable fractional polynomials Further exploration of time-varying covariates, complex with examples Additional treatment of the exponential, Weibull, and log-logistic parametric regression models Increased emphasis on interpreting and using results as well as utilizing multiple imputation methods to analyze data with missing values New examples and exercises at the end of each chapter Analyses throughout the text are performed using Stata® Version 9, and an accompanying FTP site contains the data sets used in the book. **Applied Survival Analysis, Second Edition** is an ideal book for graduate-level

courses in biostatistics, statistics, and epidemiologic methods. It also serves as a valuable reference for practitioners and researchers in any health-related field or for professionals in insurance and government.

Cracking the Coding Interview

Project Management

Doing Meta-Analysis with R

Practical Strategies for Staying Focused, Hitting Ship Dates, and Building Solid Teams

Information Theory and Best Practices in the IT Industry

An Agile Adoption and Transformation Survival Guide

Support Constant Change

Praise for the first edition: "This excellent text will be useful to every system engineer (SE) regardless of the domain. It covers ALL relevant SE material and does so in a very clear, methodical fashion. The breadth and depth of the author's presentation of SE principles and practices is outstanding." -Philip Allen This textbook presents a comprehensive, step-by-step guide to System Engineering analysis, design, and development via an integrated set of concepts, principles, practices, and methodologies. The methods presented in this text apply to any type of human system -- small, medium, and large organizational systems and system development projects delivering engineered systems or services across multiple business sectors such as medical, transportation, financial, educational, governmental, aerospace and defense, utilities, political, and charity, among others. Provides a common focal point for "bridging the gap" between and unifying System Users, System Acquirers, multi-discipline System Engineering, and Project, Functional, and Executive Management education, knowledge, and decision-making for developing systems, products, or services Each chapter provides definitions of key terms, guiding principles, examples, author's notes, real-world examples, and exercises, which highlight and reinforce key SE&D concepts and practices Addresses concepts employed in Model-Based Systems Engineering (MBSE), Model-Driven Design (MDD), Unified Modeling Language (UMLTM) / Systems Modeling Language (SysMLTM), and Agile/Spiral/V-Model Development such as user needs, stories, and use cases analysis; specification development; system architecture development; User-Centric System Design (UCSD); interface definition & control; system integration & test; and Verification & Validation (V&V) Highlights/introduces a new 21st Century Systems Engineering & Development (SE&D) paradigm that is easy to understand and implement.

Provides practices that are critical stagingpoints for technical decision making such as Technical StrategyDevelopment; Life Cycle requirements; Phases, Modes, & States;SE Process; Requirements Derivation; System ArchitectureDevelopment, User-Centric System Design (UCSD); EngineeringStandards, Coordinate Systems, and Conventions; et al. Thoroughly illustrated, with end-of-chapter exercises andnumerous case studies and examples, Systems EngineeringAnalysis, Design, and Development, Second Edition is a primarytextbook for multi-discipline, engineering, system analysis, andproject management undergraduate/graduate level students and avaluable reference for professionals.

“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.

Clinical Research Computing: A Practitioner's Handbook deals with the nuts-and-bolts of providing informatics and computing support for clinical research. The subjects that the practitioner must be aware of are not only technological and scientific, but also organizational and managerial. Therefore, the author offers case studies based on real life experiences in order to prepare the readers for the challenges they may face during their experiences either supporting clinical research or supporting electronic record systems. Clinical research computing is the application of computational methods to the broad field of clinical research. With the advent of modern digital computing, and the powerful data collection, storage, and analysis that is possible with it, it becomes more relevant to understand the technical details in order to fully seize its opportunities. Offers case studies, based on real-life examples where possible, to engage the readers with more complex examples Provides studies backed by technical details, e.g., schema diagrams, code snippets or algorithms illustrating particular techniques, to give the readers confidence to employ the techniques described in their own settings Offers didactic content organization and an increasing complexity through the chapters

The importance of benchmarking in the service sector is well recognized as it helps in continuous improvement in products and work processes. Through benchmarking, companies have strived to implement best practices in order to remain competitive in the product- market in which they operate. However studies on benchmarking, particularly in the software development sector, have neglected using multiple variables and therefore have not been as comprehensive. Information Theory and Best Practices in the IT Industry fills this void by examining benchmarking in the business of software development and studying how it is affected by development process, application type, hardware platforms used, and many other variables. Information Theory and Best Practices in the IT Industry begins by examining practices of benchmarking productivity and critically appraises them. Next the book identifies different variables which affect productivity and variables that affect quality,

developing useful equations that explaining their relationships. Finally these equations and findings are applied to case studies. Utilizing this book, practitioners can decide about what emphasis they should attach to different variables in their own companies, while seeking to optimize productivity and defect density.

Professional Software Development

Regression Modeling of Time-to-Event Data

A Practitioner's Handbook

Programming Embedded Systems

Software Quality and Java Automation Engineer Survival Guide

Taming Wild Software Schedules

Breaking the Code

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

"Early in his software developer career, John Sonmez discovered that technical knowledge alone isn't enough to break through to the next income level - developers need "soft skills" like the ability to learn new technologies just in time, communicate clearly with management and consulting clients, negotiate a fair hourly rate, and unite teammates and coworkers in working toward a common goal. Today John helps more than 1.4 million programmers every year to increase their income by developing this unique blend of skills. Who Should Read This Book? Entry-Level Developers - This book will show you how to ensure you have the technical skills your future boss is looking for, create a resume that leaps off a hiring manager's desk, and escape the "no work experience" trap. Mid-Career Developers - You'll see how to find and fill in gaps in your technical knowledge, position yourself as the one team member your boss can't live without, and turn those dreaded annual reviews into chance to make an iron-clad case for your salary bump. Senior Developers - This book will show you how to become a specialist who can command above-market wages, how building a name for yourself can make opportunities come to you, and how to decide whether consulting or entrepreneurship are paths you should pursue. Brand New Developers - In this book you'll discover what it's like to be a professional software developer, how to go from "I know some code" to possessing the skills to work on a development team, how to speed along your learning by avoiding common beginner traps, and how to decide whether you should invest in a programming degree or 'bootcamp.'"

Learn to integrate programming with good documentation. This book teaches you the craft of documentation for each step in the software development lifecycle, from understanding your users' needs to publishing, measuring, and maintaining useful developer documentation. Well-documented projects save time for both developers on the project and users of the software. Projects without adequate documentation

suffer from poor developer productivity, project scalability, user adoption, and accessibility. In short: bad documentation kills projects. Docs for Developers demystifies the process of creating great developer documentation, following a team of software developers as they work to launch a new product. At each step along the way, you learn through examples, templates, and principles how to create, measure, and maintain documentation—tools you can adapt to the needs of your own organization. What You'll Learn Create friction logs and perform user research to understand your users' frustrations Research, draft, and write different kinds of documentation, including READMEs, API documentation, tutorials, conceptual content, and release notes Publish and maintain documentation alongside regular code releases Measure the success of the content you create through analytics and user feedback Organize larger sets of documentation to help users find the right information at the right time Who This Book Is For Ideal for software developers who need to create documentation alongside code, or for technical writers, developer advocates, product managers, and other technical roles that create and contribute to documentation for their products and services.

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

Death March

Software Requirements

Five Steps to a Life-Changing Software Development Job

Basic Concepts, Self Review and Interview Preparation

Code Complete

Think Your Way to a Better Life

Doing Meta-Analysis with R: A Hands-On Guide serves as an accessible introduction on how meta-analyses can be conducted in R. Essential steps for meta-analysis are covered, including calculation and pooling of outcome measures, forest plots, heterogeneity diagnostics, subgroup analyses, meta-regression, methods to control for

publication bias, risk of bias assessments and plotting tools. Advanced but highly relevant topics such as network meta-analysis, multi-three-level meta-analyses, Bayesian meta-analysis approaches and SEM meta-analysis are also covered. A companion R package, `dmetar`, is introduced at the beginning of the guide. It contains data sets and several helper functions for the `meta` and `metafor` package used in the guide. The programming and statistical background covered in the book are kept at a non-expert level, making the book widely accessible. Features

- Contains two introductory chapters on how to set up an R environment and do basic imports/manipulations of meta-analysis data, including exercises
- Describes statistical concepts clearly and concisely before applying them in R
- Includes step-by-step guidance through the coding required to perform meta-analyses, and a companion R package for the book

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.

If you want to land a job in tech, you need to know how to code. That much is obvious. A skill that is equally as important-but often overlooked-is knowing how to market yourself to potential employers, especially if you're new to the industry. Do you know how to land interviews and deliver a compelling case for why you should be hired? Bobby Davis Jr. has helped place hundreds of aspiring coders into high-paying tech jobs. In *Breaking the Code*, he shares the proven strategies he uses with his students and offers up insider tips that will make you stand out from the competition. You'll learn how to avoid the biggest stumbling block when it comes to landing a job and what you should build before an interview if you don't have a project to show. Bobby also teaches you the secret to accessing jobs not found on corporate job boards and exponentially increasing your chances of getting the job you want. The path to \$100,000 a year-and infinite possibilities beyond that--begins with *Breaking the Code*.