Software Engineer Sommerville 9th Edition Solution Manual

Right here, we have countless book **Software Engineer Sommerville 9th Edition Solution Manual** and collections to check out. We additionally provide variant types and moreover type of the books to browse. The usual book, fiction, history, novel, scientific research, as competently as various further sorts of books are readily easy to get to here.

As this Software Engineer Sommerville 9th Edition Solution Manual, it ends going on physical one of the favored ebook Software Engineer Sommerville 9th Edition Solution Manual collections that we have. This is why you remain in the best website to look the unbelievable ebook to have.

Erfolgreich selbstgesteuert lernen Klaus Konrad 2008

Best Practice Software-Engineering Alexander Schatten 2010-01-28

Software-Komponenten tragen durch einen hohen Grad an Wiederverwendbarkeit, bessere

Testbarkeit und Wartbarkeit zur effizienten Herstellung komplexer Software-Anwendungen bei. Diese Vorteile bedingen jedoch oft eine aufwendigere Einarbeitung beim Einstieg in diese Materie durch die Vielzahl an komplexen Komponenten-Frameworks, Werkzeugen und Entwurfsansätzen. Das vorliegende Buch „Best-Practice Software Engineering" bietet Neu- und Wiedereinsteigern in die komponentenorientierte Software-Entwicklung eine Einführung in die Materie durch eine abgestimmte Zusammenstellung von praxiserprobten Konzepten, Techniken und Werkzeugen für alle Aspekte eines erfolgreichen Projekts. Für moderne Software-Entwicklung sind eine Vielzahl von unterschiedlichen Fähigkeiten erforderlich, die nur im richtiger Kombination zu einem erfolgreichen Ergebnis führen. Daher wird in diesem Buch besonderer Wert darauf gelegt, nicht einzelne Techniken des Software Engineerings isoliert zu betrachten, sondern das effiziente Zusammenspiel verschiedener Aspekte darzustellen. Schwerpunkte liegen auf Vorgehensstrategien im Software-Lebenszyklus, Projektmanagement, Qualitätssicherung, UML-Modellierung, Entwurfsmustern und Architekturen, komponentenorientierter Software-Entwicklung sowie ausgewählten Techniken und Werkzeugen. Zu den Beispielen im Buch finden Sie den vollständigen Source Code sowie umfangreiche Fallbeispiele zu Artefakten aus dem Projektverlauf auf der Webseite zum Buch.

Einführung in die Programmierung mit Java Robert Sedgewick 2011

Agile Transformation Bas van Lieshout 2020-12-04 Transformation ist eine Reise! Agile Transformation bedeutet eine völlige Kehrtwendung und damit eine völlige Umgestaltung der gesamten Organisation. Wie wird nun eine solche unternehmensweite Umstellung auf Agilität umgesetzt? Was sind die Voraussetzungen für den Erfolg einer solchen Transformation? Das Buch bietet Transformationsteams eine praktische Anleitung zu allen Aspekten eines Transformationsprozesses. Die Autoren, ausgewiesene Experten auf diesem Gebiet, beschreiben anhand vieler Fallbeispiele einen Schritt-für-Schritt-Plan zur Vorbereitung und Durchführung der agilen Transformation, wohlwissend, dass dieser Plan kontinuierlich angepasst werden muss, basierend auf den Erkenntnissen und Ergebnissen der vorherigen Lernerfahrungen. Auch Themen, die während der Transformation zusätzliche Aufmerksamkeit erfordern und gerne vergessen werden, wie u.a. Personalentwicklung, Finanzen und Compliance, wird ein eigenes Kapitel gewidmet. Das Buch ist Pflichtlektüre für alle, die sich mit agilen Transformationen und Veränderung im Allgemeinen befassen und ihr Unternehmen in eine nachhaltige und bewegliche Zukunft führen möchten.
Geschichte der Psychiatrie Edward Shorter 2003
Selbstgesteuertes Lernen Klaus Konrad 2009-01
Das UML-Benutzerhandbuch Grady Booch 2006 In dieser - lang erwarteten - Überarbeitung zur Version 2.0 der umfassenden Einführung in UML bieten

die Entwickler der Sprache - Grady Brooch, James Rumbaugh, Ivar Jacobsen - eine Einführung, die sich mit den Kernpunkten befasst. Ausgehend von einer Übersicht über UML wird die Sprache anhand der Vorstellung bestimmter Konzepte und Schreibweisen in jedem Kapitel Schritt für Schritt erläutert. Das Buch sorgt einerseits für einen umfassenden Überblick über alle Diagrammtypen sowie Elemente von UML in der zweiten Version und stellt andererseits den nötigen Praxisbezug her, um UML 2.0 effektiv für eigene Projekte einzusetzen. Die tief greifenden Erläuterungen und die an Beispielen orientierte Herangehensweise der Autoren, sorgen für ein schnelles Verständnis des komplexen Themas.

Software-Engineering - kompakt Anja Metzner 2020-03-09 Im Software-Engineering geht es um die Modellierung und Entwicklung komplexer, qualitativ hochwertiger Software und die für einen erfolgreich durchgeführten Realisierungsprozess geeigneten Methoden, Werkzeuge und Standards. In diesem kompakten Lehrbuch werden die wichtigsten Themen rund um Software-Engineering erklärt, zusammengefasst und mit kleinen Praxisbeispielen vertieft. Von zentraler Bedeutung für das Software-Engineering ist der Software-Lebenszyklus. Gemeint ist damit der gesamte Prozess, der zur Erstellung und Erhaltung eines Softwaresystems führt. Sowohl in traditionellen als auch in agilen Softwareerstellungsprozessen läuft dieser Lebenszyklus ab. Bewährt hat sich in der Praxis die Einteilung in sogenannte Phasen, denen die

Gliederung folgt. Nach einer kurzen Einführung werden in Kapitel 2 vorab phasenübergreifende Verfahren wie divergierende Vorgehensmodelle und Projektmanagement besprochen. Kapitel 3 behandelt die Planungsphase; Kapitel 4 ist dem Requirements-Engineering gewidmet, bei dem die Software-Anforderungen kreativ konstruiert, analysiert und – traditionell oder agil – dokumentiert werden. In Kapitel 5 folgt die Besprechung der Verfahren für die Designphase der Software. Hier wird hinterfragt, wie gute Software-Architekturen Erfolg versprechend erdacht, mit der UML-Notation geeignet modelliert und in späteren Projekten wiederverwendet werden können. Kapitel 6 widmet sich der Test- und Abnahmephase und damit den wichtigen Qualitätssicherungsfragen. Abschließend wird in Kapitel 7 die Wartung – zur wirksamen Erhaltung von Softwaresystemen – erklärt. Anfänger erhalten eine schnelle Orientierung und kompaktes, fundiertes Grundwissen. Fortgeschrittene Leser finden hier ein aktuelles, gut strukturiertes Nachschlagewerk. Unter https://www.hanser-fachbuch.de/buch/Software+Engineering+kompakt/9783446459496 finden interessierte Leser weitere Übungsaufgaben zum Thema Software-Engineering.

Angewandte abstrakte Algebra Rudolf Lidl 1982

Usability Evaluation and Interface Design Michael J. Smith 2001-08-01 This three volume set provides the complete proceedings of the Ninth International Conference on Human-Computer Interaction held August, 2001 in New Orleans. A total of 2,738

individuals from industry, academia, research institutes, and governmental agencies from 37 countries submitted their work for presentation at the conference. The papers address the latest research and application in the human aspects of design and use of computing systems. Those accepted for presentation thoroughly cover the entire field of human-computer interaction, including the cognitive, social, ergonomic, and health aspects of work with computers. The papers also address major advances in knowledge and effective use of computers in a variety of diversified application areas, including offices, financial institutions, manufacturing, electronic publishing, construction, and health care.

Effektiv Java programmieren Joshua Bloch 2002-01

Der rational unified process Philippe Kruchten 1999

Digitale Bildverarbeitung Bernd Jähne 2012-12-02 Selbst komplexe Aufgaben der Bildverarbeitung sind heute auf gängigen PCs lösbar. Das Know-how dafür vermittelt dieses Standardwerk – von den Grundlagen der digitalen Bildverarbeitung bis hin zu modernen Konzepten. Es gliedert sich in die drei Teile Bildgewinnung, Bildverarbeitung und Bildanalyse. Die Übungsaufgaben sind größtenteils interaktiv und können mit der auf CD-ROM beiliegenden Demosoftware bearbeitet werden. Die 7. Auflage ist an die Erfordernisse der Master-Studiengänge angepasst worden und enthält ein neues Kapitel über Bildsensoren.

Concurrent programming in Java Doug Lea 1997

Verteilte Systeme Andrew S. Tanenbaum 2008

Objektorientierte Analyse & Design von Kopf bis Fuss Brett D. McLaughlin 2007-05-15 Kluge Bücher über Objektorientierte Analyse & Design gibt es viele. Leider versteht man die meisten erst, wenn man selbst schon Profi-Entwickler ist... Und was machen all die Normalsterblichen, die natürlich davon gehört haben, dass OOA&D dazu beiträgt, kontinuierlich tolle Software zu schreiben, Software, die Chef und Kunden glücklich macht - wenn sie aber nicht wissen, wie sie anfangen sollen? Sie könnten damit beginnen, dieses Buch zu lesen! Denn Objektorientierte Analyse & Design von Kopf bis Fuß zeigt Ihnen Schritt für Schritt, wie Sie richtige OO-Software analysieren, entwerfen und entwickeln. Software, die sich leicht wiederverwenden, warten und erweitern lässt. Software, die keine Kopfschmerzen bereitet. Software, der Sie neue Features spendieren können, ohne die existierende Funktionalität zu gefährden. Sie lernen, Ihre Anwendungen flexibel zu halten, indem Sie OO-Prinzipien wie Kapselung und Delegation anwenden. Sie lernen, die Wiederverwendung Ihrer Software dadurch zu begünstigen, dass Sie das OCP (das Open-Closed-Prinzip) und das SRP (das Single-Responsibility-Prinzip) befolgen. Sie lernen, wie sich verschiedene Entwurfsmuster, Entwicklungsansätze und Prinzipien zu einem echten OOA&D-Projektlebenszyklus ergänzen, UML, Anwendungsfälle und -diagramme zu verwenden, damit auch alle Beteiligten klar miteinander

kommunizieren können, und Sie die Software abliefern, die gewünscht wird. Diesem Buch wurden die neuesten Erkenntnisse aus der Lerntheorie und der Kognitionswissenschaft zugrunde gelegt - Sie können davon ausgehen, dass Sie nicht nur schnell vorankommen, sondern dabei auch noch eine Menge Spaß haben!

Softwareentwicklung von Kopf bis Fuss Dan Pilone 2008-07-15 Was lernen Sie mit diesem Buch? Haben Sie sich schon einmal gefragt, was es mit testgetriebener Entwicklung auf sich hat? Oder auf welcher Basis es die richtig guten Consultants schaffen, gewaltige Stundensätze zu kassieren? Vielleicht sind Sie auch gerade an dem Punkt, an dem Sie Ihre Builds automatisieren wollen, Ihren Code in eine Versionskontrolle füttern, einem Refactoring unterziehen oder mit ein paar Entwurfsmustern anreichern wollen. Egal: Wenn Sie mit diesem Buch fertig sind, werden Sie ganz selbstverständlich Ihre Burndown-Rate verfolgen, den Durchsatz Ihres Teams berücksichtigen und sich erfolgreich Ihren Weg durch Anforderungen, Entwurf, Entwicklung und Auslieferung iterieren. Wieso sieht dieses Buch so anders aus? Wir gehen davon aus, dass Ihre Zeit zu kostbar ist, um mit neuem Stoff zu kämpfen. Statt Sie mit Bleiwüstentexten langsam in den Schlaf zu wiegen, verwenden wir für Softwareentwicklung von Kopf bis Fuß ein visuell und inhaltlich abwechslungsreiches Format, das auf Grundlage neuster Forschungsergebnisse im Bereich der Kognitionswissenschaft und der Lerntheorie entwickelt wurde. Wir wissen nämlich, wie Ihr Gehirn

arbeitet.

Struktur und Interpretation von Computerprogrammen Harold Abelson 1991 Diese moderne Einf hrung in die Informatik ist am renommierten Massachusetts Institute of Technology entstanden und repr sentiert den dortigen Ausbildungsstandard f r Studenten der Informatik und der Elektrotechnik. Das ganzheitliche Verst ndnis der Informatik unter Einbeziehung der K nstlichen Intelligenz, das in diesem Buch vermittelt wird, hat es weltweit, und insbesondere auch im deutschsprachigen Bereich, bereits in der englischen Originalfassung zu einer beliebten Grundlage f r die Einf hrungsvorlesung gemacht. Zur Notation der Programme wird Scheme verwendet, ein Dialekt der Programmiersprache Lisp, der die Leistungsf higkeit und die Eleganz von Lisp und Algol verbindet. Die Besonderheit dieser einf hrenden Vorlesung beruht auf zwei Grund berzeugungen: 1. Eine Computersprache ist nicht einfach ein Weg, einen Computer zur Ausf hrung von Operationen zu bewegen, sondern vielmehr ein neuartiges Medium, um Vorstellungen ber Verfahrensweisen auszudr cken. So m ssen Programme geschrieben werden, damit Menschen sie lesen und modifizieren, und nur nebenbei, damit Maschinen sie ausf hren k nnen. 2. Das Wesentliche bei einer Vorlesung auf diesem Niveau ist weder die Syntax von speziellen Sprachkonstruktionen, noch sind es raffinierte Algorithmen zur effizienten Berechnung bestimmter Funktionen oder gar die mathematische Analyse von Algorithmen oder die

Grundlagen der Informatik, sondern vielmehr die Techniken, mit denen die geistige Komplexit t gro er Softwaresysteme unter Kontrolle gehalten werden kann.

Software requirements Karl E. Wiegers 2005 Nicht wenige Software-Projekte erreichen ihre gesteckten Ziele nicht, da bereits in ihrer Anfangsphase Anforderungen an die Software nicht gründlich genug analysiert und dokumentiert wurden. Oft wird auch vernachlässigt, dass Softwareentwicklung genauso viel mit Kommunikation, wie mit eigentlicher Entwicklungsarbeit zu tun hat. An diesem Punkt setzt dieser Klassiker der Softwareentwicklungsliteratur an, in dem überzeugend präsentiert wird, warum die Erhebung, Zusammenstellung und das Managen von Software Requirements essentiell für erfolgreiche Projekte ist und mit welchen erprobten Mitteln diese Aufgaben am besten zu meistern sind. Karl Wiegers zeigt damit, wie Requirements-Analysten, Projektleiter, aber auch alle Programmierer und Designer, die Anforderungen der Kunden umsetzen müssen, Produktivität, Termintreue, Kundenzufriedenheit und Wartungs- und Supportkosten mit dem im Buch beschriebenen Praktiken drastisch verbessern können. - Realistische Erwartungen für Funktionalität und Qualität setzen - Geschäftsregeln in die Anwendungsentwicklung integrieren - Anwendungsfälle zur Definition von Benutzeranforderungen verwenden - Unausgesprochene und wechselnde Requirements identifizieren und managen - Revisionen einschränken und damit Kosten sparen - Besser Software

produzieren
Implementation Patterns - Studentenausgabe Kent Beck 2010
Datenbanksysteme Thomas Connolly 2002
Software Engineering Ian Sommerville 2018-09-21
Rechnerarchitektur : Von der digitalen Logik zum Parallelrechner Andrew S. Tanenbaum 2014
Software Engineering: A Practitioner's Approach Roger Pressman 2014-01-23 For almost three decades, Roger Pressman's Software Engineering: A Practitioner's Approach has been the world's leading textbook in software engineering. The new eighth edition represents a major restructuring and update of previous editions, solidifying the book's position as the most comprehensive guide to this important subject. The eighth edition of Software Engineering: A Practitioner's Approach has been designed to consolidate and restructure the content introduced over the past two editions of the book. The chapter structure will return to a more linear presentation of software engineering topics with a direct emphasis on the major activities that are part of a generic software process. Content will focus on widely used software engineering methods and will de-emphasize or completely eliminate discussion of secondary methods, tools and techniques. The intent is to provide a more targeted, prescriptive, and focused approach, while attempting to maintain SEPA's reputation as a comprehensive guide to software engineering. The 39 chapters of the eighth edition are organized into five parts - Process,

Modeling, Quality Management, Managing Software Projects, and Advanced Topics. The book has been revised and restructured to improve pedagogical flow and emphasize new and important software engineering processes and practices.
Efficiently Conducting Quality-of-Service Analyses by Templating Architectural Knowledge Lehrig, Sebastian Michael 2018-04-06
Clean Code - Refactoring, Patterns, Testen und Techniken für sauberen Code Robert C. Martin 2013-12-18 h2> Kommentare, Formatierung, Strukturierung Fehler-Handling und Unit-Tests Zahlreiche Fallstudien, Best Practices, Heuristiken und Code Smells Clean Code - Refactoring, Patterns, Testen und Techniken für sauberen Code Aus dem Inhalt: Lernen Sie, guten Code von schlechtem zu unterscheiden Sauberen Code schreiben und schlechten Code in guten umwandeln Aussagekräftige Namen sowie gute Funktionen, Objekte und Klassen erstellen Code so formatieren, strukturieren und kommentieren, dass er bestmöglich lesbar ist Ein vollständiges Fehler-Handling implementieren, ohne die Logik des Codes zu verschleiern Unit-Tests schreiben und Ihren Code testgesteuert entwickeln Selbst schlechter Code kann funktionieren. Aber wenn der Code nicht sauber ist, kann er ein Entwicklungsunternehmen in die Knie zwingen. Jedes Jahr gehen unzählige Stunden und beträchtliche Ressourcen verloren, weil Code schlecht geschrieben ist. Aber das muss nicht sein. Mit Clean Code präsentiert Ihnen der bekannte Software-Experte Robert C. Martin ein

revolutionäres Paradigma, mit dem er Ihnen aufzeigt, wie Sie guten Code schreiben und schlechten Code überarbeiten. Zusammen mit seinen Kollegen von Object Mentor destilliert er die besten Praktiken der agilen Entwicklung von sauberem Code zu einem einzigartigen Buch. So können Sie sich die Erfahrungswerte der Meister der Software-Entwicklung aneignen, die aus Ihnen einen besseren Programmierer machen werden – anhand konkreter Fallstudien, die im Buch detailliert durchgearbeitet werden. Sie werden in diesem Buch sehr viel Code lesen. Und Sie werden aufgefordert, darüber nachzudenken, was an diesem Code richtig und falsch ist. Noch wichtiger: Sie werden herausgefordert, Ihre professionellen Werte und Ihre Einstellung zu Ihrem Beruf zu überprüfen. Clean Code besteht aus drei Teilen:Der erste Teil beschreibt die Prinzipien, Patterns und Techniken, die zum Schreiben von sauberem Code benötigt werden. Der zweite Teil besteht aus mehreren, zunehmend komplexeren Fallstudien. An jeder Fallstudie wird aufgezeigt, wie Code gesäubert wird – wie eine mit Problemen behaftete Code-Basis in eine solide und effiziente Form umgewandelt wird. Der dritte Teil enthält den Ertrag und den Lohn der praktischen Arbeit: ein umfangreiches Kapitel mit Best Practices, Heuristiken und Code Smells, die bei der Erstellung der Fallstudien zusammengetragen wurden. Das Ergebnis ist eine Wissensbasis, die beschreibt, wie wir denken, wenn wir Code schreiben, lesen und säubern. Dieses Buch ist ein Muss für alle Entwickler, Software-Ingenieure,

Projektmanager, Team-Leiter oder Systemanalytiker, die daran interessiert sind, besseren Code zu produzieren. Über den Autor: Robert C. »Uncle Bob« Martin entwickelt seit 1970 professionell Software. Seit 1990 arbeitet er international als Software-Berater. Er ist Gründer und Vorsitzender von Object Mentor, Inc., einem Team erfahrener Berater, die Kunden auf der ganzen Welt bei der Programmierung in und mit C++, Java, C#, Ruby, OO, Design Patterns, UML sowie Agilen Methoden und eXtreme Programming helfen.

Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie John E. Hopcroft 2003

Concurrent Engineering in the 21st Century Josip Stjepandić 2015-01-30 Presenting the gradual evolution of the concept of Concurrent Engineering (CE), and the technical, social methods and tools that have been developed, including the many theoretical and practical challenges that still exist, this book serves to summarize the achievements and current challenges of CE and will give readers a comprehensive picture of CE as researched and practiced in different regions of the world. Featuring in-depth analysis of complex real-life applications and experiences, this book demonstrates that Concurrent Engineering is used widely in many industries and that the same basic engineering principles can also be applied to new, emerging fields like sustainable mobility. Designed to serve as a valuable reference to industry experts, managers, students, researchers, and

software developers, this book is intended to serve as both an introduction to development and as an analysis of the novel approaches and techniques of CE, as well as being a compact reference for more experienced readers.

Maschinelles Lernen Ethem Alpaydin 2022-01-19 Maschinelles Lernen ist die künstliche Generierung von Wissen aus Erfahrung. Dieses Buch diskutiert Methoden aus den Bereichen Statistik, Mustererkennung und kombiniert die unterschiedlichen Ansätze, um effiziente Lösungen zu finden. Diese Auflage bietet ein neues Kapitel über Deep Learning und erweitert die Inhalte über mehrlagige Perzeptrone und bestärkendes Lernen. Eine neue Sektion über erzeugende gegnerische Netzwerke ist ebenfalls dabei.

Das DevOps-Handbuch Gene Kim 2017-08-09 Mehr denn je ist das effektive Management der IT entscheidend für die Wettbewerbsfähigkeit von Organisationen. Viele Manager in softwarebasierten Unternehmen ringen damit, eine Balance zwischen Agilität, Zuverlässigkeit und Sicherheit ihrer Systeme herzustellen. Auf der anderen Seite schaffen es High-Performer wie Google, Amazon, Facebook oder Netflix, routinemäßig und zuverlässig hundertoder gar tausendmal pro Tag Code auszuliefern. Diese Unternehmen verbindet eins: Sie arbeiten nach DevOps-Prinzipien. Die Autoren dieses Handbuchs folgen den Spuren des Romans Projekt Phoenix und zeigen, wie die DevOps-Philosophie praktisch implementiert wird und Unternehmen dadurch umgestaltet werden können.

Sie beschreiben konkrete Tools und Techniken, die Ihnen helfen, Software schneller und sicherer zu produzieren. Zudem stellen sie Ihnen Maßnahmen vor, die die Zusammenarbeit aller Abteilungen optimieren, die Arbeitskultur verbessern und die Profitabilität Ihres Unternehmens steigern können. Themen des Buchs sind: Die Drei Wege: Die obersten Prinzipien, von denen alle DevOps-Maßnahmen abgeleitet werden. Einen Ausgangspunkt finden: Eine Strategie für die DevOps-Transformation entwickeln, Wertketten und Veränderungsmuster kennenlernen, Teams schützen und fördern. Flow beschleunigen: Den schnellen Fluss der Arbeit von Dev hin zu Ops ermöglichen durch eine optimale Deployment-Pipeline, automatisierte Tests, Continuous Integration und Continuous Delivery. Feedback verstärken: Feedback-Schleifen verkürzen und vertiefen, Telemetriedaten erzeugen und Informationen unternehmensweit sichtbar machen. Kontinuierliches Lernen ermöglichen: Eine Just Culture aufbauen und ausreichend Zeit reservieren, um das firmenweite Lernen zu fördern.

Grundlagen der Kommunikationstechnik John G. Proakis 2003 Proakis und Salehi haben mit diesem Lehrbuch einen Klassiker auf dem Gebiet der modernen Kommunikationstechnik geschaffen. Der Schwerpunkt liegt dabei auf den digitalen Kommunikationssystemen mit Themen wie Quellen- und Kanalcodierung sowie drahtlose Kommunikation u.a. Es gelingt den Autoren dabei der Brückenschlag von der Theorie zur

Praxis. Außerdem werden mathematische Grundlagen wie Fourier-Analyse, Stochastik und Statistik gleich mitgeliefert. Zielgruppe:Studierende der Elektro- und Informationstechnik und verwandter technischer Studienrichtungen wie Kommunikationstechnik, Technische Infor.

Computernetze James F. Kurose 2004

Books in Print 1987

Die Fünfte Computer-Generation FEIGENBAUM 2013-11-11

Produkt-Service Systeme Jan C. Aurich 2010-01-25 Bisher haben sich produzierende Unternehmen auf Entwicklung, Produktion und Vertrieb von Sachprodukten konzentriert, inzwischen fragen Kunden jedoch zunehmend komplette Problemlösungen nach. Unternehmen stehen vor der Herausforderung, den Wandel zum Full-Service Provider zu vollziehen. In dem Band wird ein Managementsystem zur Unterstützung von Planung, Entwicklung, Konfiguration und Realisierung von Produkt-Service Systemen (PSS) im erweiterten Wertschöpfungsnetzwerk vorgestellt und Methoden zur Organisation sowie zum Kompetenzerwerb aufgezeigt.

Vom Mythos des Mann-Monats Frederick P. Brooks 2019-05-08 Nur wenige Bücher über das Projektmanagement bei Software haben sich als so einflussreich und zeitlos gültig erwiesen wie "Vom Mythos des Mann-Monats": Fred Brooks bietet hier mit einem Mix aus harten Fakten und provokanten Ideen jedem tiefe Einsichten, der komplexe Projekte zu managen hat. Die Essays in diesem Buch stellen die Quintessenz seiner Erfahrungen als Projektmanager erst

für die Hardware der IBM/360-Computerfamilie, dann als Leiter der Entwicklung des - wahrhaft gigantischen - Betriebssystems OS/360 dar. Die Besonderheit dieses Buches liegt aber auch darin, dass Brooks, 20 Jahre nach Erscheinen des Originals, seine ursprünglichen Vorstellungen und Visionen noch einmal überdacht und sie um neue Erkenntnisse und Ratschläge bereichert hat. Dieses Buch ist ein Muss sowohl für Kenner seiner Arbeiten als auch Leser, die Brooks nun zum ersten Mal entdecken.

CSCW 1994

Requirements Engineering Klaus Pohl 2008

Die 7 Wege zur Effektivität für Jugendliche Sean Covey 2019-09-10

Stell dir vor, du hättest eine Roadmap, eine Schritt-für-Schritt-Anleitung, die dir hilft, von dort, wo du jetzt stehst, dorthin zu gelangen, wo du in der Zukunft sein möchtest. Deine Ziele, deine Träume, deine Pläne, sie alle sind für dich zum Greifen nah. Du brauchst nur die richtigen Werkzeuge, um sie zu erreichen, zu verwirklichen, umzusetzen. Genau das ist Sean Coveys wegweisendes Buch" Die 7 Wege zur Effektivität für Jugendliche" für Millionen von Teenagern: ein Leitfaden zu mehr Selbstbewusstsein und Erfolg. Die neu übersetzte und gekürzte Neuauflage des Bestsellers, der in 20 Sprachen übersetzt und weltweit mehr als fünf Millionen Mal verkauft wurde, wendet die zeitlosen Prinzipien der 7 Wege auf die besonderen Probleme und lebensverändernden Entscheidungen von Jugendlichen an. Covey bietet mit

diesem Buch einen einfachen Ansatz, um dir zu helfen, dein Selbstbild zu verbessern, Freundschaften aufzubauen, Gruppenzwängen zu widerstehen, deine Ziele zu erreichen, mit deinen Eltern gut auszukommen sowie die neuen Herausforderungen unserer Zeit wie Cybermobbing und Social Media zu meistern. Cartoons, clevere Ideen, tolle Zitate und wahre Geschichten von Jugendlichen aus der ganzen Welt machen das Buch zu einem kurzweiligen Lesevergnügen – lebendig erzählt und von Timo Wuerz ansprechend illustriert. "Die 7 Wege zur Effektivität für Jugendliche" sind das beste Beispiel dafür, wie du als Teenager und junger Erwachsener sinnerfüllt leben und erfolgreich sein kannst: ein unverzichtbares Buch für Jugendliche, aber auch für Eltern, Lehrer, Berater und alle, die mit Jugendlichen arbeiten. "Die 7 Wege zur Effektivität für Jugendliche" ist der ultimative Erfolgsratgeber für Teenager – jetzt aktualisiert und überarbeitet für das digitale Zeitalter. „Wenn die 7 Wege zur Effektivität für Jugendliche dir nicht helfen, dann musst du schon jetzt ein perfektes Leben haben." Jordan McLaughlin, 17 Jahre alt

Compiler 2008