Kindle File Format Reusable Software Components Object Oriented Embedded Systems Programming In C

Recognizing the artifice ways to get this books **Reusable Software Components Object Oriented Embedded Systems Programming In C** is additionally useful. You have remained in right site to start getting this info. get the Reusable Software Components Object Oriented Embedded Systems Programming In C member that we offer here and check out the link.

You could buy guide Reusable Software Components Object Oriented Embedded Systems Programming In C or get it as soon as feasible. You could speedily download this Reusable Software Components Object Oriented Embedded Systems Programming In C after getting deal. So, taking into consideration you require the ebook swiftly, you can straight acquire it. Its so agreed simple and correspondingly fats, isnt it? You have to favor to in this broadcast

Reusable Software-Bertrand Meyer 1994

Techniques and principles; Presentation of the libraries; Class reference.

Software Engineering with Reusable Components-

Johannes Sametinger 1997-06-19

The book provides a clear understanding of what software reuse is, where the problems are, what benefits to expect, the activities, and its different forms. The reader is also

given an overview of what sofware components are, different kinds of components and compositions, a taxonomy thereof, and examples of successful component reuse. An introduction to software engineering and software process models is also provided.

Design Patterns: Elements of Reusable Object-Oriented Software-Erich Gamma 1995

Capturing a wealth of experience about the design of object-oriented software, four top-notch designers present a catalog of simple and succinct solutions to commonly occurring design problems. Previously undocumented, these 23 patterns allow designers to create more flexible, elegant, and ultimately reusable designs without having to rediscover the design solutions themselves.

Component-Oriented Programming-Andy Ju An Wang 2005-04-29

Component Oriented Programming offers a unique programming-centered approach to component-based software development that delivers the well-developed training and practices you need to successfully apply this cost-effective method. Following an overview of basic theories and methodologies, the authors provide a unified component infrastructure for building component software using JavaBeans, EJB, OSGi, CORBA, CCM, .NET, and Web services. You'll learn how to develop reusable software components; build a software system of pre-built software components; design and implement a component-based software system using various component-based approaches. Clear organization and self-testing features make Component Oriented

Programming an ideal textbook for graduate and undergraduate courses in computer science, software engineering, or information technology as well as a valuable reference for industry professionals.

Programming .NET Components-Juval Lowy 2005-07-27

'Programming .NET Components', second edition, updated to cover .NET 2.0., introduces the Microsoft .NET Framework for building components on Windows platforms. From its many lessons, tips, and guidelines, readers will learn how to use the .NET Framework to program reusable, maintainable, and robust components.

Reusable Software Components-Ted Van Sickle 1997

Helps real-time embedded systems designers combine the development benefits of the widely-used C language and object-oriented techniques not normally associated with C.Introduces object-oriented programming to microcontroller programmers familiar with C. Shows how objects can be written in C, and developed into classes. Presents useful objects and classes for microcontroller programs, including a class that creates instances of an asynchronous serial port. Shows how to implement components to handle timer functions and input capture. Compiles data sheets for all components derived in the book.Programmers working with real-time embedded systems.

Design Patterns CD-Erich Gamma 1998

The 23 patterns contained in the book, Design Patterns: Elements of Reusable Object-Oriented Software have become an essential resource for anyone developing

reusable software designs. Now these design patterns, along with the entire text of the book, are being made available on CD. This electronic version will enable programmers to install the patterns directly onto a computer or network and create an architecture for using and building reusable components. Produced in HTML format, the CD is heavily cross-referenced with numerous links to the online text.

Principles of Package Design-Matthias Noback 2018-11-13

Apply design principles to your classes, preparing them for reuse. You will use package design principles to create packages that are just right in terms of cohesion and coupling, and are user- and maintainer-friendly at the same time. The first part of this book walks you through the five SOLID principles that will help you improve the design of your classes. The second part introduces you to the best practices of package design, and covers both package cohesion principles and package coupling principles. Cohesion principles show you which classes should be put together in a package, when to split packages, and if a combination of classes may be considered a "package" in the first place. Package coupling principles help you choose the right dependencies and prevent wrong directions in the dependency graph of your packages. What You'll LearnApply the SOLID principles of class designDetermine if classes belong in the same packageKnow whether it is safe for packages to depend on each other Who This Book Is For Software developers with a broad range of experience in the field, who are looking for ways to reuse,share, and

distribute their code

Software Reuse-James W. Hooper 1991

1. Background and Introduction.- 1.1 The Problem.- 1.2 Concepts and Definitions.- 1.3 Research Activities.- 1.4 Status of Reuse Practice.- 1.5 Scope and Organization of this Book.- 1.6 References.- 2. Managerial Guidelines.- 2.1 Managerial Issues and Approaches.- 2.1.1 Organizational Management and Structure.- 2.1.2 Organizational Behavior.- 2.1.3 Contractual and Legal Considerations.- 2.1.4 Financial Considerations.- 2.1.5 Case Study: Reuse Program at Hartford Insurance Group.- 2.2 Software Development and Maintenance Incorporating Reuse.- 2.2.1 The Software Process.- 2.2.2 Life-Cycle Models.- 2.2.3 A Generic Reuse/Reusability Model.- 2.2.4 Establishing a Process.- 2.2.5 Case Study: JIAWG Reuse-Based Process Plan.- 2.3 References.- 3. Technical Guidelines.- 3.1 Domain Analysis.- 3.1.1 Overview.- 3.1.2 Case Study: The Domain Analysis Project at Software Engineering Institute (SEI).- 3.2 Creating Reusable Components.- 3.2.1 Spanning the Life Cycle.- 3.2.2 Requirements and Designs.- 3.2.2.1 Overview.- 3.2.2.2 Object-Oriented Approaches.- 3.2.3 Code Components.- 3.2.3.1 Code Component Structures.- 3.2.3.2 Programming Style.- 3.2.4 Component Quality.- 3.2.5 Classifying and Storing Components.- 3.2.6 Case Study: A Design Study of Telephony Software at Ericsson Telecom.- 3.3 Reusing Components.- 3.3.1 Cognitive Aspects.- 3.3.2 Searching and Retrieving.- 3.3.3 Understanding and Assessing Components.- 3.3.4 Adapting Components.- 3.3.5

Composition of Code Components.- 3.3.6 Case Study: A Quantitative Study of Spacecraft Control Software Reuse at GSFC.- 3.3.7 Case Study: The Reusable Software Library (RSL) at Intermetrics, Inc..- 3.4 Tools and Environments.- 3.5 References.- 4. Getting Started.- 4.1 Discussion.- 4.2 A Phased Approach.- 4.3 References.- Appendix A: Collected Guidelines.- Appendix B: Guidelines for Reusable Ada Code.

UML Components-John Cheesman 2001

The UML was conceived and first implemented as a language for describing the design of object-oriented programs. Its widespread adoption and inherent flexibility has, inevitably, led to its use in other areas, including the design of component-based systems, While it is not a perfect fit for component-based development, this book describes how best to use UML 1.3 in the specification and design of medium to large systems that utilize server-side component technologies.

Testing and Quality Assurance for Component-based Software-Jerry Gao 2003

From the basics to the most advanced quality of service (QoS) concepts, this all encompassing, first-of-its-kind book offers an in-depth understanding of the latest technical issues raised by the emergence of new types, classes and qualities of Internet services. The book provides end-to-end QoS guidance for real time multimedia communications over the Internet. It offers you a multiplicity of hands-on examples and simulation script support, and shows you where and when it is preferable to use these techniques for QoS support in networks and Internet traffic with widely varying characteristics and demand

profiles. This practical resource discusses key standards and protocols, including real-time transport, resource reservation, and integrated and differentiated service models, policy based management, and mobile/wireless QoS. The book features numerous examples, simulation results and graphs that illustrate important concepts, and pseudo codes are used to explain algorithms. Case studies, based on freely available Linux/FreeBSD systems, are presented to show you how to build networks supporting Quality of Service. Online support material including presentation foils, lab exercises and additional exercises are available to text adopters.

Modular Specification and Verification of Object-Oriented Programs-Peter Müller 2003-07-31

Software systems play an increasingly important role in modern societies. Smart cards for personal identi?cation, e-banking, software-controlled me- cal tools, airbags in cars, and autopilots for aircraft control are only some examples that illustrate how everyday life depends on the good behavior of software. Consequently, techniques and methods for the development of hi-quality, dependable software systems are a central research topic in computer science. A fundamental approach to this area is to use formal speci?cation and veri?cation. Speci?cation languages allow one to describe the crucial p-perties of software systems in an abstract, mathematically precise, and implementation-independent way. By formal veri?cation, one can then prove that an implementation really has the desired, speci?ed properties. Although this

formal methods approach has been a research topic for more than 30 years, its practical success is still restricted to domains in which devel- ment costs are of minor importance. Two aspects are crucial to widen the application area of formal methods: – Formal speci?cation techniques have to be smoothly integrated into the software and program development process. – The techniques have to be applicable to reusable software components. This way, the quality gain can be exploited for more than one system, thereby justifying the higher development costs. Starting from these considerations, Peter Muller¨ has developed new te- niques for the formal speci?cation and veri?cation of object-oriented so- ware. The speci?cation techniques are declarative and implementati- independent. They can be used for object-oriented design and programming.

Software Reuse: Advances in Software Reusability-William B. Frakes 2004-02-02

This book constitutes the refereed proceedings of the 6th International Conference on Software Reuse, ICSR-6, held in Vienna, Austria, in June 2000. The 26 revised full papers presented were carefully reviewed and selected from numerous submissions. The book is divided into topical sections on generative reuse and formal description languages, object-oriented methods, product line architectures, requirements reuse and business modeling, components and libraries, and design patterns.

Tutorial, Software Reuse-Will Tracz 1988

An overview of the basic issues concerning software reuse with focus on mental

and supplemental tools that support the concept. Describes the processes including: components, software libraries, methodologies, Ada resuse experiences, and object-oriented computing. Acidic paper; no index. Annotation

Software Engineering with Reusable Components-Johannes Sametinger 2013-04-17

The book provides a clear understanding of what software reuse is, where the problems are, what benefits to expect, the activities, and its different forms. The reader is also given an overview of what sofware components are, different kinds of components and compositions, a taxonomy thereof, and examples of successful component reuse. An introduction to software engineering and software process models is also provided.

Component-based Software Development-Kung-Kiu Lau 2004

- First book of its kind (case studies in CBD) - Covers different kinds of components - Covers different component models/technologies - Includes a wide scope of CBD topics - Covers both theoretical and practical work - Includes both formal and informal approaches - Provides a snapshot of current concerns and pointers to future trends

Software Reuse-Bernard Coulange 2012-12-06

Software Reuse is a state of the art book concerning all aspects of software reuse. It does away with the hype and shows the reality. Different techniques are presented which enable software reuse and the author demonstrates why object-oriented methods are better for reuse than other approaches. The book details the different factors

to take into account when managing reusable components: characterisation, identification, building, verification, storage, search, adaptation, maintenance and evolution. Comparisons and description of various types of companies that could benefit from applying reuse techniques are included outlining, amongst other things, increased profitability and likely problems that might arise from the purchase and selling of reuse tools and components. Based on a real experience of software reuse in a company with a bibliography of more than 200 references provided, this book is a 'must have' for all those working in the software reuse field.

Computational Intelligence Techniques and Their Applications to Software Engineering Problems- Ankita Bansal 2020-09-28

Computational Intelligence Techniques and Their Applications to Software Engineering Problems focuses on computational intelligence approaches as applicable in varied areas of software engineering such as software requirement prioritization, cost estimation, reliability assessment, defect prediction, maintainability and quality prediction, size estimation, vulnerability prediction, test case selection and prioritization, and much more. The concepts of expert systems, case-based reasoning, fuzzy logic, genetic algorithms, swarm computing, and rough sets are introduced with their applications in software engineering. The field of knowledge discovery is explored using neural networks and data mining techniques by determining the underlying and hidden patterns in software data sets. Aimed at graduate students and researchers in computer science

engineering, software engineering, information technology, this book: Covers various aspects of in-depth solutions of software engineering problems using computational intelligence techniques Discusses the latest evolutionary approaches to preliminary theory of different solve optimization problems under software engineering domain Covers heuristic as well as meta-heuristic algorithms designed to provide better and optimized solutions Illustrates applications including software requirement prioritization, software cost estimation, reliability assessment, software defect prediction, and more Highlights swarm intelligence-based optimization solutions for software testing and reliability problems

Component Software: Beyond Object-Oriented Programming, 2/E-Szyperski 2003-09

Developing Object-oriented Multimedia Software-Philipp Ackermann 1996

This book on the MET++ multimedia application framework provides an in-depth look at the concepts and techniques applied in an object-oriented class library to support multimedia application development. It is a reference for software designers and programmers who want to build multimedia applications by reusing components of the MET++ framework.

Software Reuse: Methods, Techniques, and Tools-Jan Bosch 2004-06-14

This book constitutes the refereed proceedings of the 8th International Conference on Software Reuse, ICSR-8, held in Madrid, Spain in July 2004. The 28 revised full papers presented were carefully reviewed and selected from numerous submissions. The

papers are organized in topical sections on software variability: requirements; testing reusable software; feature modeling; aspect-oriented software development; component and service development; code level reuse; libraries, classification, and retrieval; model-based approaches; transformation and generation; and requirements.

Cost-justifying Usability- Randolph G. Bias 1994

Today's increasingly competitive and fiscally constrained business environment is fostering the need to cut costs and justify expenditures. Usability engineering is not yet universally accepted, nor is it yet an integrated aspect of software engineering, and would-be usability champions need more help than ever to win the funding necessary to introduce and promote usability engineering techniques. Cost-Justifying Usability is the first book to address pragmatically and in detail the question of how usability engineering professionals and their managers can cost-justify their proposals and efforts. The book offers specific techniques for quantifying costs and benefits, making a convincing and successful business case for investment in usability engineering.This book comprises a thorough and well-integrated collection of chapters written by experienced and prominent usability experts. Taken together, these chapters provide readers with:An overall framework for cost-justifying usability engineering programs that can be applied to any contextAn examination of the unique factors and issues in cost-justifying usability efforts for three very different types of organizations: vendor companies, international development organizations,

and contractor companiesCase studies of successful cost-justification effortsA look at some special issues regarding cost-justification of usability, including"discount"usability engineering techniques, success factors for introducing usability engineering into development organizations, specialized tools for usability cost-justification, and a look to the future of usability engineeringPractical and effective insight for human factors professionals, interface designers, software development managers, and

eBook: Object-Oriented Systems Analysis 4e- BENNETT 2021-03-26

eBook: Object-Oriented Systems Analysis 4e

Object-Oriented Software: Design and Maintenance- Luiz Fernando Capretz 1996-09-09

This is a textbook for a course in object-oriented software engineering at advanced undergraduate and graduate levels, as well as for software engineers. It contains more than 120 exercises of diverse complexity. The book discusses fundamental concepts and terminology on object-oriented software development, assuming little background on software engineering, and emphasizes design and maintenance rather than programming. It also presents up-to-date and easily understood methodologies and puts forward a software life cycle model which explicitly encourages reusability during software development and maintenance.

Computational Intelligence Techniques and Their Applications to Software Engineering Problems- Ankita Bansal 2020-09-27

Computational Intelligence

Techniques and Their Applications to Software Engineering Problems focuses on computational intelligence approaches as applicable in varied areas of software engineering such as software requirement prioritization, cost estimation, reliability assessment, defect prediction, maintainability and quality prediction, size estimation, vulnerability prediction, test case selection and prioritization, and much more. The concepts of expert systems, case-based reasoning, fuzzy logic, genetic algorithms, swarm computing, and rough sets are introduced with their applications in software engineering. The field of knowledge discovery is explored using neural networks and data mining techniques by determining the underlying and hidden patterns in software data sets. Aimed at graduate students and researchers in computer science engineering, software engineering, information technology, this book: Covers various aspects of in-depth solutions of software engineering problems using computational intelligence techniques Discusses the latest evolutionary approaches to preliminary theory of different solve optimization problems under software engineering domain Covers heuristic as well as meta-heuristic algorithms designed to provide better and optimized solutions Illustrates applications including software requirement prioritization, software cost estimation, reliability assessment, software defect prediction, and more Highlights swarm intelligence-based optimization solutions for software testing and reliability problems

Object-Oriented Software-Luiz Fernando Capretz 1996-09-01

This is a textbook for a

course in object-oriented software engineering at advanced undergraduate and graduate levels, as well as for software engineers. It contains more than 120 exercises of diverse complexity.The book discusses fundamental concepts and terminology on object-oriented software development, assuming little background on software engineering, and emphasizes design and maintenance rather than programming.It also presents up-to-date and easily understood methodologies and puts forward a software life cycle model which explicitly encourages reusability during software development and maintenance.

Testing Object-Oriented Software-Imran Bashir 2012-12-06

Addressing various aspects of object-oriented software techniques with respect to their impact on testing, this text argues that the testing of object-oriented software is not restricted to a single phase of software development. The book concentrates heavily on the testing of classes and of components or sub-systems, and a major part is devoted to this subject. C++ is used throughout this book that is intended for software practitioners, managers, researchers, students, or anyone interested in object-oriented technology and its impacts throughout the software engineering life-cycle.

Object-oriented Technology For Database And Software Systems-V S Alagar 1995-09-28

Object orientation has become a “must know” subject for managers, researchers, and software practitioners interested in the design, evolution, reuse and management of efficient software components.The book

contains technical papers reflecting both theoretical and practical contributions from researchers in the field of object-oriented (OO) databases and software engineering systems. The book identifies actual and potential areas of integration of OO and database technologies, current and future research directions in software methodologies, and reflections about the OO paradigm.In providing current research and relevant information about this promising and rapidly growing field of object-oriented databases and software engineering systems, this book is invaluable to research scientists, practitioners, and graduate students working in the areas of databases and software engineering.

Business Component-Based Software Engineering-Franck Barbier 2012-12-06

Business Component-Based Software Engineering, an edited volume, aims to complement some other reputable books on CBSE, by stressing how components are built for large-scale applications, within dedicated development processes and for easy and direct combination. This book will emphasize these three facets and will offer a complete overview of some recent progresses. Projects and works explained herein will prompt graduate students, academics, software engineers, project managers and developers to adopt and to apply new component development methods gained from and validated by the authors. The authors of Business Component-Based Software Engineering are academic and professionals, experts in the field, who will introduce the state of the art on CBSE from their shared experience by working on the same

projects. Business Component-Based Software Engineering is designed to meet the needs of practitioners and researchers in industry, and graduate-level students in Computer Science and Engineering.

SOFTWARE ENGINEERING- K. L. JAMES 2008-11-17

Software Engineering discusses the major issues associated with different phases of software development life cycle. Starting from the basics, the book discusses several advanced topics. Topics like software project management, software process models, developing methodologies, software specification, software testing and quality, software implementation, software security, software maintenance and software reuse are discussed. This book also gives an introduction to the new emerging technologies, trends and practices in software engineering field. New topics such as MIMO technology, AJAX, etc. are included in the book. The topics like .NET framework, J2EE, etc. are also dealt with. Case Studies, discussions on real-life situations of dealing with IT related problems and finding their solutions in an easy manner, are given in each chapter. Elegant and simple style of presentation makes the reading of this book a pleasant experience. Students of Computer Science and Engineering, Information Technology and Computer Applications should find this book highly useful. It would also be useful for IT technology professionals who are interested to get acquainted with the latest and the newest technologies.

Software Maintenance - A Management Perspective- Phaneendra Nath Vellanky 2007-10-23

Computer systems play an important role in our society. Software drives those systems. Massive investments of time and resources are made in developing and implementing these systems. Maintenance is inevitable. It is hard and costly. Considerable resources are required to keep the systems active and dependable. We cannot maintain software unless maintainability characters are built into the products and processes. There is an urgent need to reinforce software development practices based on quality and reliability principles. Though maintenance is a mini development lifecycle, it has its own problems. Maintenance issues need corresponding tools and techniques to address them. Software professionals are key players in maintenance. While development is an art and science, maintenance is a craft. We need to develop maintenance personnel to master this craft. Technology impact is very high in systems world today. We can no longer conduct business in the way we did before. That calls for reengineering systems and software. Even reengineered software needs maintenance, soon after its implementation. We have to take business knowledge, procedures, and data into the newly reengineered world. Software maintenance people can play an important role in this migration process. Software technology is moving into global and distributed networking environments. Client/server systems and object-orientation are on their way. Massively parallel processing systems and networking resources are changing database services into corporate data warehouses. Software engineering environments,

rapid application development tools are changing the way we used to develop and maintain software. Software maintenance is moving from code maintenance to design maintenance, even onto specification maintenance. Modifications today are made at specification level, regenating the software components, testing and integrating them with the system. Eventually software maintenance has to manage the evolution and evolutionary characteristics of software systems. Software professionals have to maintain not only the software, but the momentum of change in systems and software. In this study, we observe various issues, tools and techniques, and the emerging trends in software technology with particular reference to maintenance. We are not searching for specific solutions. We are identifying issues and finding ways to manage them, live with them, and control their negative impact.

Object-oriented Software Construction-Bertrand Meyer 1997

This volume aims to study how practicing software developers, in industrial as well as academic environments, can use object technology to improve the quality of the software they produce. It includes topics on concurrency and Internet programming.

Software Reuse-Bernard Coulange 1998

Software Reuse is a state of the art book concerning all aspects of software reuse. It does away with the hype and shows the reality. Different techniques are presented which enable software reuse and the author

demonstrates why object-oriented methods are better for reuse than other approaches. The book details the different factors to take into account when managing reusable components: characterisation, identification, building, verification, storage, search, adaptation, maintenance and evolution. Comparisons and description of various types of companies that could benefit from applying reuse techniques are included outlining, amongst other things, increased profitability and likely problems that might arise from the purchase and selling of reuse tools and components. Based on a real experience of software reuse in a company with a bibliography of more than 200 references provided, this book is a 'must have' for all those working in the software reuse field.

Object-oriented Programming in Eiffel-Pete Thomas 1995

A complete tutorial of the Eiffel programming language, this book emphasizes the role of abstract data types (ADTs) in software development. It shows how Eiffel's unique approach to "programming by contact" encourages the design of reusable software components and explores techniques for ensuring the correctness of programs.

Software Reuse for Dynamic Systems in the Cloud and Beyond-Ina Schaefer 2014-12-22

This book constitutes the refereed proceedings of the 14th International Conference on Software Reuse for Dynamic Systems in the Cloud and Beyond, ICSR 2015, held in Miami, FL, USA, in January 2015. The 21 revised full papers presented together with 3 revised short papers were carefully reviewed and selected from 60

submissions. The papers cover several software engineering areas where software reuse is important, such as software product lines, domain analysis, open source, components, cloud, quality.

Formal Aspects of Measurement-Tim Denvir 2012-12-06

This book contains the eight invited papers presented at the workshop on Formal Aspects of Measurement held at South Bank University on 5th May 1991, organised by the British Computer Society's Special Interest Group on Formal Aspects of Computer Science (FACS). In addition, there are five papers which have been included because of their relevance to the subject of the workshop. The book represents something of a landmark in software engineering research. The British Computer Society's Special Interest Group on Formal Aspects of Computer Science (FACS) has an established reputa tion among researchers in formal methods of software specification, design and validation. These researchers have not in the past paid much attention to software measurement. Perhaps software measurement re search was felt to have emphasised its management potential at the expense of proper scientific foundations? At any rate, for the FACS group to host a workshop in this field is recognition of the significant body of formal measurement theories and techniques which has now become available to software engineers.

Software Reuse and Reverse Engineering in Practice-Patrick A. V. Hall 1992

OOIS'98-Colette Rolland

2012-12-06

The Sorbonne University is very proud to host this year the oms Conference on Object Oriented Information Systems. There is a growing awareness of the importance of object oriented techniques, methods and tools to support information systems engineering. The term information systems implies that the computer based systems are designed to provide adequate and timely information to human users in organizations. The term engineering implies the application of a rigorous set of problem solving approaches analogous to those found in traditional engineering disciplines. The intent of this conference is to present a selected number of those approaches which favor an object oriented view of systems engineering. oms '98 is the fifth edition of a series of conferences. Starting in 1994 in London, this series evolved from a British audience to a truly European one. The goal is to build a world wide acknowledged forum dedicated to object oriented information systems engineering. This conference is organized with the aim to bring together researchers and practitioners in Information Systems, Databases and Software Engineering who have interests in object oriented information systems. The objective is to advance understanding about how the object technology can empower information systems in organizations, on techniques for designing effective and efficient information systems and methods and development tools for information systems engineering. The conference aims also at discussing the lessons learned from large scale projects using objects. The call for oms was given

international audience.

Software Architecture- Patrick Donohoe 2013-06-05

Software architecture is a primary factor in the creation and evolution of virtually all products involving software. It is a topic of major interest in the research community where pronusmg formalisms, processes, and technologies are under development. Architecture is also of major interest in industry because it is recognized as a significant leverage point for manipulating such basic development factors as cost, quality, and interval. Its importance is attested to by the fact that there are several international workshop series as well as major conference sessions devoted to it. The First Working IFIP Conference on Software Architecture (WICSAl) provided a focused and dedicated forum for the international software architecture community to unify and coordinate its effort to advance the state of practice and research. WICSA 1 was organized to facilitate information exchange between practising software architects and software architecture researchers. The conference was held in San Antonio, Texas, USA, from February 22nd to February 24th, 1999; it was the initiating event for the new IFIP TC-2 Working Group on Software Architecture. This proceedings document contains the papers accepted for the conference. The papers in this volume comprise both experience reports and technical papers. The proceedings reflect the structure of the conference and are divided into six sections corresponding to the working groups established for the conference.

Plasticity, Limit Analysis,

Stability And Structural Design: An Academic Life Journey From Theory To Practice-Wai-fah Chen 2021-01-22

This book is a personal anthology of the author's utmost academic works and accomplishments with his former students and colleagues intended as an enduring record for the engineering community for many years to come.The author's forty-year professional career and academic life journey is first briefly sketched in Chapter 1 and more details are elaborated in three chapters that follow: Chapter 2: The first ten years at Lehigh — beginning to show; Chapter 3: Twenty=three years at Purdue — the highly productive years; and Chapter 4: seven years at UH — the pursuit of excellence. The author's specific academic contributions are documented in the following three chapters: Chapter 5: 23 academic bulletins are selected to highlight his 10 major research areas; Chapter 6: 23 Academic masterpiece books are listed along with their respective peer review comments; and Chapter 7: academic publications include journal articles, conference proceedings and symposiums, and lectures and keynotes. The book ends with the listing of all the author's 55 doctoral students' dissertation titles in Chapter 8.In 1975 at Lehigh, the author published a milestone treatise on Limit Analysis and Soil Plasticity. In 1982 at Purdue, he published another pioneering work on Plasticity in Reinforced Concrete.In September 1999, the author was recruited by UH to take the Deanship of the College of Engineering to accomplish the noble mission: to build the College to become one of the top 50 engineering schools by strengthening

the faculty, improving the facilities, and increasing the enrollment. Over his seven years at UH, a lot of progress was made in all these three areas — the research program expanded, facilities improved, and enrollment increased.