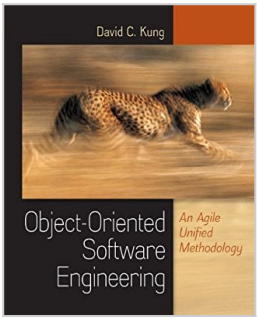

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science)

By David Kung

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung

Object-Oriented Software Engineering: An Agile Unified Methodology, presents a step-by-step methodology - that integrates Modeling and Design, UML, Patterns, Test-Driven Development, Quality Assurance, Configuration Management, and Agile Principles throughout the life cycle. The overall approach is casual and easy to follow, with many practical examples that show the theory at work. The author uses his experiences as well as real-world stories to help the reader understand software design principles, patterns, and other software engineering concepts. The book also provides stimulating exercises that go far beyond the type of question that can be answered by simply copying portions of the text.

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science)

By David Kung

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung

Object-Oriented Software Engineering: An Agile Unified Methodology, presents a step-by-step methodology - that integrates Modeling and Design, UML, Patterns, Test-Driven Development, Quality Assurance, Configuration Management, and Agile Principles throughout the life cycle. The overall approach is casual and easy to follow, with many practical examples that show the theory at work. The author uses his experiences as well as real-world stories to help the reader understand software design principles, patterns, and other software engineering concepts. The book also provides stimulating exercises that go far beyond the type of question that can be answered by simply copying portions of the text.

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung Bibliography

- Sales Rank: #706593 in Books
- Brand: Brand: McGraw-Hill Science/Engineering/Math
- Published on: 2013-01-22
- Original language: English
- Number of items: 1
- Dimensions: 9.40" h x 1.20" w x 7.60" l, 2.70 pounds
- Binding: Hardcover
- 720 pages

Editorial Review

Review
"It has been two years since my graduation. I really enjoyed OOSE and Design Patterns classes. I still remember the first day how I learned to create Use Cases from requirements. As a software developer, I follow your teaching in every project. The methodology that I leaned has helped me design/develop software in a RIGHT WAY. In every project I had worked on, I use STEP BY STEP DESIGN methodology that I leaned in your classes. My project lead is so glad to see professional documents ... THANK YOU for the knowledge that you shared with us." - from an undergraduate alumnus

"I would like to say 'thank you' for the way you designed and taught our classes. The two classes I took with you - OO concepts and Design Patterns are helping me earning 'bread and butter' every day. I worked as a consultant for past few years in different companies and I realized the process you taught is far more advanced than they follow here in real world - which is very good and helpful. I got chance to suggest even more experienced people out here in the field. I am very grateful to you." - from an undergraduate alumnus

"We are using your methodology on our project. It's gone very well so far. We are currently in the implementation phase. They have a lot of good design documentation. I'm thinking of setting a goal this year for each group to do at least one project using your methodology." - from a large multinational company

"Dr. Kung taught an eight week (32 hours) course to our team of 40 software engineers last summer. The methodology and patterns are currently being used in several projects of our embedded and simulation lines of products. We successfully completed a major project using this methodology and see significant productivity and quality improvements with very few integration and verification defects in comparison to similar projects we have done before. I highly recommend this to any one developing and managing software." - from the project manager of a large multinational manufacturing company

From the Author

The writing of the book has been motivated by years of unsuccessful search for an OO software engineering textbook that
(1) teaches students practical, up-to-date problem-solving skills and solid theoretical foundations,
(2) is interesting and easy to learn, and
(3) contributes to the student's long term career growth.
As the instructor, and director of the ABET accredited software engineering program, I feel obligated to develop the needed teaching material that fulfills these goals. The material presented in the book is the result of years of effort and continual improvements, based on my observation of students' performance, and the feedback received from students.
The book is also written for instructors who want to switch to an agile software engineering approach. Software engineers and students who are puzzled by the problems faced in design, implementation and testing and who want to improve their OO development capabilities will find the book helpful. Finally, the book also devotes separate chapters for system engineering, software quality assurance, testing object-oriented and web applications, software maintenance, software configuration management, software project management, and software security.

From the Inside Flap

PREFACE

Background

Computers are widely used in all sectors of our society, performing a variety of functions with the application software running on them. As a result, the market for software engineers is booming. The March 2006 issue of the Money Magazine ranked software engineer as the number 1 of the 50 best jobs in U.S. According to Bureau of Labor Statistics (BLS) 2010-2020 projections, the total number of jobs in application development software engineers and system analyst positions is expected to increase from 520,800 to 664,500 (27.6%) and from 544,400 to 664,800 (22.10%), respectively. To be able to perform the work required of an application development software engineer or systems analyst, an education in software engineering is highly desired. However, according to the data released by BLS ("Earned Awards and Degrees, by Field of Study, 2005-2006"), only 160 bachelor and 600 master's degrees in software engineering, and 10,289 bachelor and 4,512 master's degrees in computer science were awarded in 2006. Thus, there is a significant gap between the demand and supply, especially for graduates with a software engineering degree.

Many people do not know the scope and usefulness of software engineering as a practice, and the discipline is often misunderstood. Many media outlets seem to define software engineering as writing Java programs. Some students think that software engineering includes everything related to software. Others think that software engineering is drawing UML diagrams, as the following story illustrates. Several years ago, after the first class of an object-oriented software engineering (OOSE) course, a student said to me, "Professor, you know that this will be an easy course for me because we've drawn lots of UML diagrams before." At the end of the semester, the student came to me again and said, "Professor, I want to tell you that we had worked very hard but we learned a lot about OO design. It is not just drawing UML diagrams as I thought." So what is software engineering? As a discipline, it encompasses research, education and application of engineering processes, methodologies, quality assurance, and project management to significantly increase software productivity and software quality while reducing software cost and time to market.

OOSE is a branch of software engineering that is characterized by its view of the world as consisting of objects relating to and interacting with each other. The advent of the C++ programming language in the 1980s marked the beginning of the OOSE era. Since then, software production began its unprecedented world-wide growth and was further accelerated by the creation and world-wide adoption of the unified modeling language (UML) and the unified process (UP). Strictly speaking, a software process describes the phases and what should be done in each phase. It does not define (in detail) how to perform the activities in each phase. A modeling language, such as UML, defines the notations, syntax and semantics for communicating and documenting analysis and design ideas. UML and UP are good and necessary but not sufficient. This is because how to produce the analysis and design ideas required to draw meaningful UML diagrams is missing.

Motivation

To fill the gaps discussed in the last paragraph, we need a methodology or a "cookbook." Unlike a process, a methodology is a detailed description of the steps and procedures or how to carry out the activities to the extent that a beginner can follow to produce and deploy the desired software system. Without a methodology, a beginning software engineer would have to spend a few years of on-job training to learn OO design, implementation and testing skills.

The writing of the book is also motivated by emerging interests in agile processes, design patterns and test driven development (TDD). Agile processes emphasize teamwork, design for change, rapid deployment of small increments of the software system, and joint development with the customer and users. Design patterns are effective design solutions to common design problems. Design patterns promote software reuse and improve team communication. TDD advocates testable software, requires test scripts to be produced before the implementation so that the latter can be tested immediately and frequently.

As an analogy, consider the development of an amusement park. The overall process includes the following

phases: planning, public approval, analysis and design, financing, construction drawings, construction, procurement of equipment, installation of equipment, pre-opening, and grand opening. However, knowing the overall process is not enough. The development team must know how to perform the activities of the phases. For example, the planning activities include development of initial concept, feasibility study, and master plan generation. The theme park team must know how to perform these activities. The analysis and design activities include "requirements acquisition" from stakeholders, site investigation, design of park layout, design of theming for different areas of the park, creating models to study the layout design and theming, and producing the master design. Again, the theme park team must know how to perform these activities to produce the master design. Unlike a process that describes the phases of activities, a methodology details the steps and procedures or how to perform the activities.
The development of an amusement park is a multi-year project and costs billions of dollars. The investor wants the park to generate revenue as early as possible; but with the above process, the investor has to wait until the entire park is completed. Once the master design is finalized, it cannot be modified easily due to the restrictions imposed by the conventional process. If the park does not meet the expectations of the stakeholders, then changes are costly once the park is completed.
Agile processes are aimed to solve these problems. With an agile process, a list of preliminary theme park requirements is acquired quickly and allowed to evolve during the development process. The amusement and entertainment facilities are then derived from the requirements and carefully grouped into clusters of facilities. A plan to develop and deploy the clusters in relatively short periods of time is produced --- that is, rapid deployment of small increments. Thus, instead of a finalized master design, the development process designs and deploys one cluster at a time. As the clusters of facilities are deployed and operational, feedback is sought and changes to the requirements, the development plan, budget and schedule are worked out with the stakeholders --- that is, joint development. In addition, the application of architectural design patterns improves quality and ability of the park to adapt to changing needs --- that is, design for change. Teamwork is emphasized because effective collaboration and coordination between the teams and team members ensure that the facilities will be developed and deployed timely and seamlessly. The agile process has a number of merits. The investor can reap the benefits much earlier because the facilities are operational as early as desired and feasible. Since a small number of the facilities is developed and deployed at a time, errors can be corrected and changes can be made more easily.
In summary, ...
Audiences, Organization, and Acknowledgment are omitted due to limit on space.

Users Review

From reader reviews:

Brandon Li:

Why don't make it to become your habit? Right now, try to ready your time to do the important work, like looking for your favorite reserve and reading a reserve. Beside you can solve your short lived problem; you can add your knowledge by the reserve entitled Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science). Try to make the book Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) as your friend. It means that it can to be your friend when you feel alone and beside that course make you smarter than previously. Yeah, it is very fortuned for you. The book makes you much more confidence because you can know anything by the book. So , we need to make new experience and also knowledge with this book.

Esmeralda Rossman:

The actual book Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) will bring that you the new experience of reading a book. The author style to elucidate the idea is very unique. Should you try to find new book you just read, this book very ideal to you. The book Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) is much recommended to you you just read. You can also get the e-book through the official web site, so you can quicker to read the book.

Oliver Crites:

Reading a book to become new life style in this yr; every people loves to examine a book. When you learn a book you can get a lots of benefit. When you read publications, you can improve your knowledge, simply because book has a lot of information in it. The information that you will get depend on what sorts of book that you have read. If you need to get information about your analysis, you can read education books, but if you act like you want to entertain yourself read a fiction books, these us novel, comics, in addition to soon. The Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) will give you a new experience in reading through a book.

Jodi Dunn:

That e-book can make you to feel relax. This kind of book Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) was multi-colored and of course has pictures on there. As we know that book Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) has many kinds or genre. Start from kids until young adults. For example Naruto or Investigation company Conan you can read and think you are the character on there. Therefore not at all of book tend to be make you bored, any it makes you feel happy, fun and loosen up. Try to choose the best book for you and try to like reading that will.

Read Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung for online ebook

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung Free PDF d0wnl0ad, audio books, books to read, good books to read, cheap books, good books, online books, books online, book reviews epub, read books online, books to read online, online library, greatbooks to read, PDF best books to read, top books to read Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung books to read online.

Online Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung ebook PDF download

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung Doc

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung Mobipocket

Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung EPub

YS2GDPXUCOA: Object-Oriented Software Engineering: An Agile Unified Methodology (Irwin Computer Science) By David Kung