Object Oriented Systems Design An Integrated Approach | 55be78f11a987b720dc6e1ab0105c1c9

Object Oriented Systems Development

This text teaches students object-oriented systems analysis and design in a highly practical and accessible way.

Covering the breadth of a large topic, this book provides a thorough grounding in object-oriented concepts, the software development process, UML and multi-tier technologies. After covering some basic ground work underpinning OO software projects, the book follows the steps of a typical development project (Requirements Capture - Design - Specification & Test), showing how an abstract problem is taken through to a concrete solution. The book is programming language agnostic - so code is kept to a minimum to avoid detail and deviation into implementation minutiae. A single case study running through the text provides a realistic example showing development from an initial proposal through to a finished system. Key artifacts such as the requirements document and detailed designs are included. For each aspect of the study case, there is an exercise for the reader to produce similar documents for a different system.

Summary: “The main objective of this book is to teach both students and practitioners of information systems, software engineering, computer science and related areas to analyze and design information systems using the FOOM methodology. FOOM combines the object-oriented approach and the functional (process-oriented) approach”--Provided by publisher.

Text written in 6 parts: 1) Introduction; 2) Management issues; 3) Object oriented analysis; 4) Object oriented design; 5) Case for OO; 6) How to get started.

This book gathers together and synthesizes all that is best and correct in object-oriented technology - emphasizing such areas as CASE tools, reuse, project management, metrics, configuration.

This text teaches readers object-oriented systems analysis and design in a highly practical and accessible way.

This book provides an introduction to practical formal modelling techniques in the context of object-oriented system design. It is aimed at both practising software engineers with some prior experience of object-oriented design/programming and at intermediate or advanced students studying object-oriented design or modelling in a short course. The following features make this book particularly attractive to potential instructors: § The relationship with UML and object-oriented programming makes it easy to integrate with the mainstream computing curriculum. Although the book is about formal methods, it does not have to be treated as a specialist topic. § The use of tools and an accessible modelling language improves student motivation. § The industry-based examples and case studies add to the credibility of the approach. § The light touch approach means that the material appeals to students with a wider range of abilities than is the case in a conventional formal methods text. § Support materials as listed above.

Evolutionary in approach, this book explores information systems development–both analysis and design–using an object-oriented methodology combined with a relational database as part of the implementation.

Object-Oriented Analysis and Design for Information Systems clearly explains real object-oriented programming in practice. Expert author Raul Sidnei Wazlawick explains concepts such as object responsibility, visibility and the real need for delegation in detail. The object-oriented code generated by using these concepts in a systematic way is concise, organized and reusable. The patterns and solutions presented in this book are based in research and industrial applications. You will come away with clarity regarding processes and use cases and a clear understanding of how to expand a use case. Wazlawick clearly explains clearly how to build meaningful sequence diagrams. Object-Oriented Analysis and Design for Information Systems illustrates how and why building a class model is not just placing classes into a diagram. You will learn the necessary organizational patterns so that your software architecture will be maintainable. Learn how to build better class models, which are more maintainable and understandable. Write use cases in a more efficient and standardized way, using more effective and less complex diagrams. Build true object-oriented code with division of responsibility and delegation.
Notations and strategies are delivered for: designing the problem domain component; designing the human interaction component; designing the task management component; designing the data management component; applying object-oriented design with object-oriented programming language; applying object-oriented design criteria; and selecting CASE for object-oriented design.

This fifth edition continues to build upon previous issues with it hands-on approach to systems analysis and design with an even more in-depth focus on the core set of skills that all analysts must possess. Dennis continues to capture the experience of developing and analysing systems in a way that readers can understand and apply and develop a rich foundation of skills as a systems analyst.

More than ever, mission-critical and business-critical applications depend on object-oriented (OO) software. Testing techniques tailored to the unique challenges of OO technology are necessary to achieve high reliability and quality. “Testing Object-Oriented Systems: Models, Patterns, and Tools” is an authoritative guide to designing and automating test suites for OO applications. This comprehensive book explains why testing must be model-based and provides in-depth coverage of techniques to develop testable models from state machines, combinational logic, and the Unified Modeling Language (UML). It introduces the test design pattern and presents 37 patterns that explain how to design responsibility-based test suites, how to tailor integration and regression testing for OO code, how to test reusable components and frameworks, and how to develop highly effective test suites from use cases. Effective testing must be automated and must leverage object technology. The author describes how to design and code specification-based assertions to offset testability losses due to inheritance and polymorphism. Fifteen micro-patterns present oracle strategies--practical solutions for one of the hardest problems in test design. Seventeen design patterns explain how to automate your test suites with a coherent OO test harness framework. The author provides thorough coverage of testing issues such as: The bug hazards of OO programming and differences from testing procedural code How to design responsibility-based tests for classes, clusters, and subsystems using class invariants, interface data flow models, hierarchic state machines, class associations, and scenario analysis How to support reuse by effective testing of abstract classes, generic classes, components, and frameworks How to choose an integration strategy that supports iterative and incremental development How to achieve comprehensive system testing with testable use cases How to choose a regression test approach How to develop expected test results and evaluate the post-test state of an object How to automate testing with assertions, OO test drivers, stubs, and test frameworks Real-world experience, world-class best practices, and the latest research in object-oriented testing are included. Practical examples illustrate test design and test automation for Ada 95, C++, Eiffel, Java, Objective-C, and Smalltalk. The UML is used throughout, but the test design patterns apply to systems developed with any OO language or methodology. 0201809389B04062001

Object-Oriented Design with Applications has long been the essential reference to object-oriented technology, which, in turn, has evolved to join the mainstream of industrial-strength software development. In this third edition—the first revision in 13 years—readers can learn to apply object-oriented methods using new paradigms such as Java, the Unified Modeling Language (UML) 2.0, and .NET. The authors draw upon their rich and varied experience to offer improved methods for object development and numerous examples that tackle the complex problems faced by software engineers, including systems architecture, data acquisition, cryptoanalysis, control systems, and Web development. They illustrate essential concepts, explain the method, and show successful applications in a variety of fields. You'll also find pragmatic advice on a host of issues, including classification, implementation strategies, and cost-effective project management. New to this new edition are An introduction to the new UML 2.0, from the notation's most fundamental and advanced elements with an emphasis on key changes New domains and contexts A greatly enhanced focus on modeling—as eagerly requested by readers—with five chapters that each delve into one phase of the overall development lifecycle. Fresh approaches to reasoning about complex systems An examination of the conceptual foundation of the widely misunderstood fundamental elements of the object model, such as abstraction, encapsulation, modularity, and hierarchy How to allocate the resources of a team of developers and manage the risks associated with developing complex software systems An appendix on object-oriented programming languages This is the seminal text for anyone who wishes to use object-oriented technology to manage the complexity inherent in many kinds of systems. Sidebars Preface Acknowledgments About the Authors Section I: Concepts Chapter 1: Complexity Chapter 2: The Object Model Chapter 3: Classes and Objects Chapter 4: Classification Section II: Method Chapter 5: Notation Chapter 6: Process Chapter 7: Pragmatics Chapter 8: System Architecture: Satellite-Based Navigation Chapter 9: Control System: Traffic Management Chapter 10: Artificial Intelligence: Cryptanalysis Chapter 11: Data Acquisition: Weather Monitoring Station Chapter 12: Web Application: Vacation Packing System Appendix A: Object-Oriented Programming Languages Appendix B: Further Reading Notes Glossary Classified Bibliography Index

eBook: Object-Oriented Systems Analysis 4e

A catalog of solutions to commonly occurring design problems, presenting 23 patterns that allow designers to create flexible and reusable designs for object-oriented software. Describes the circumstances in which each pattern is applicable, and discusses the consequences and trade-offs of using the pattern within a larger design. Patterns are compiled from real systems, and include code for implementation in object-oriented programming languages like C++ and Smalltalk. Includes a bibliography. Annotation copyright by Book News, Inc., Portland, OR

This text is the first to present an object-oriented methodology from the outset for beginning Systems Analysis and Design students. It is the first book to introduce object-oriented methods without relying on classical methods to introduce key concepts and without requiring students to know Java or C++. The widely used UML notation --unified modeling language-- will be used throughout the book for all diagrams and model renderings. The key benefit to this approach is that it makes the course easier to teach since many students come to this course with limited backgrounds having only taken one introductory MIS course. Also, this approach is appealing because object-oriented methodology is widely used in industry.

This book explains how to model a problem domain by abstracting objects, attributes, and relationships from observations of the real world. It provides a wealth of examples, guidelines, and suggestions based on the authors' extensive experience in both real time and commercial software development. This book describes the first of three steps in the method of Object-Oriented Analysis. Subsequent steps are described in Object Lifecycles by the same authors.

eBook: Object-Oriented Systems Analysis and Design Using UML
The leading guide to real-time systems design-revised and updated This third edition of Phillip Laplante's bestselling, practical guide to building real-time systems maintains its predecessors' unique holistic, systems-based approach devised to help engineers write problem-solving software. Dr. Laplante incorporates a survey of related technologies and their histories, complete with time-saving practical tips, hands-on instructions, C code, and insights into decreasing ramp-up times. Real-Time Systems Design and Analysis, Third Edition is essential for students and practicing software engineers who want improved designs, faster computation, and ultimate cost savings. Chapters discuss hardware considerations and software requirements, software systems design, the software production process, performance estimation and optimization, and engineering considerations. This new edition has been revised to include: * Up-to-date information on object-oriented technologies for real-time including object-oriented analysis, design, and languages such as Java, C++, and C# * Coverage of significant developments in the field, such as: New life-cycle methodologies and advanced programming practices for real-time, including Agile methodologies Analysis techniques for commercial real-time operating system technology Hardware advances, including field-programmable gate arrays and memory technology * Deeper coverage of: Scheduling and rate-monotonic theories Synchronization and communication techniques Software testing and metrics Real-Time Systems Design and Analysis, Third Edition remains an unmatched resource for students and practicing software engineers who want improved designs, faster computation, and ultimate cost savings.

This book approaches system analysis and design with an object-oriented perspective, faithful to UML and others currently in use in many organizations. The SDC is central in the development of an information system; the book shows how each step of the SDC builds on itself. It provides readers with a strong systematic framework, linking one chapter to the next; this approach enables readers to easily learn object-oriented system analysis and design. All terminology and diagrams are UML compliant. A running case (The Pine Valley Furniture Webstore) is used throughout the book as an example. Readers can develop, propose, implement, and maintain a Webstore, learning through doing. The end-of-chapter case, Broadway Entertainment Company Inc., shows readers how a fictional video and record retailer develops an object-oriented application. Coverage includes: foundations for object-oriented systems development; project planning and management; systems analysis; systems design; and systems implementation and operation. An excellent "how-to" guide for systems analysts and designers.

The fourth edition of Object-Oriented Systems Analysis and Design has been revised and updated to reflect the most up-to-date approaches to information systems development. Still a best-seller in its field, Bennett’s, McRobb’s and Farmer’s text remains a key teaching resource for Systems Analysis and Design courses at both undergraduate and postgraduate level. The book provides a clear, practical framework for development that uses all the major techniques from UML 2.2. It follows an iterative and incremental approach based on the industry-standard Unified Process, placing systems analysis and design in the context of the whole systems lifecycle. Structured in four parts, the first provides the background to information systems analysis and design and to object-orientation. The second part focuses on the activities of requirements gathering and systems analysis, as well as the basic notation of UML. Part three covers the activities of systems architecture and design, and UML notation for object design, and the book concludes with the implementation of systems and the issues of how the systems life cycle is organized and how reusable components can be developed.

Object-Oriented Information Engineering: Analysis, Design, and Implementation discusses design, both its object-oriented and traditional development and analysis, on which the book gives much focus. The book begins with an introduction to information engineering and its phases, object-oriented information engineering, and object orientation. The text then moves on to more specific topics, such as business information requirements; detailed object modeling; business functions and subject areas; and individual object behaviors and object interactions. The book also explains the integration and validation of analysis models; object structure designs; and system designs and its different applications. The text is recommended for undergraduates and practitioners of computer and/or information engineers who want more about object-oriented design, its relation with traditional design, and its analysis. The book is also for those who wish to contribute and conduct further studies in the field of object-oriented design.

Covers O-O concepts, tools, development life cycle, problem solving, modeling, analysis, and design, while utilizing UML (Unified Modeling Language) for O-O modeling. UML has become the standard notation for modeling O-O systems and is being embraced by major software developers like Microsoft and Oracle.

This guide deals with the design and implementation of advanced information systems. It covers object-oriented data management systems, distributed environments, and advanced user interfaces i.e. those integrating text, pictures, video and sound. This book also focuses on migration issues involved in going from relational database management systems to object-oriented database management issues, and discusses the advantages/disadvantages of both types of systems. The authors have developed a unique Frame-Object Analysis Methodology for advanced modelling. It also shows the reader what constitutes an advanced distributed information system and how to design and implement one. The handbook will benefit database analysts, database administrators, programmers and members of technical staff interested in data models. Andeleigh is the author of UNIX SYSTEM ARCHITECTURE.

The Model Driven Architecture defines an approach where the specification of the functionality of a system can be separated from its implementation on a particular technology platform. The idea being that the architecture will be able to easily be adapted for different situations, whether they be legacy systems, different languages or yet to be invented platforms. MDA is therefore, a significant evolution of the object-oriented approach to system development. Advanced System Design with Java, UML and MDA describes the factors involved in designing and constructing large systems, illustrating the design process through a series of examples, including a Scrabble player, a jukebox using web streaming, a security system, and others. The book first considers the challenges of software design, before introducing the Unified Modelling Language and Object Constraint Language. The book then moves on to discuss systems design as a whole, covering internet systems design, web services, Flash, XML, XSLT, SOAP, Servlets, Javascript and JSP. In the final section of the book, the concepts and terminology of the Model Driven Architecture are discussed. To get the most from this book, readers will need introductory knowledge of software engineering, programming in Java and basic knowledge of HTML. * Examines issues raised by the Model-Driven Architecture approach to development * Uses easy to grasp case studies to illustrate complex concepts * Focused on the internet applications and technologies that are essential for students in the online age.
"This book consists of a series of high-level discussions on technical and managerial issues related to object-oriented development"—Provided by publisher.

This work introduces students to the overall process of systems analysis and design. It can be used as a course book for students who are first encountering systems analysis and design at any level. This second edition contains many updates, including the latest version of the UML standard, and reflects the most up to date approaches to the information systems development process. It provides a clear and comprehensive treatment of UML 1.4 in the context of the systems development life cycle, without assuming previous knowledge of analysis and design. It also discusses implementation issues in detail and gives code fragments to show possible mappings to implementation technology. Extensive use of examples and exercises from two case studies provides the reader with many opportunities to practise the application of UML.

This guide deals with the design and implementation of advanced information systems. It covers object-oriented data management systems, distributed environments, and advanced user interfaces i.e. those integrating text, pictures, video and sound. This book also focuses on migration issues involved in going from relational database management systems to object-oriented database management issues, and discusses the advantages/disadvantages of both types of systems. The authors have developed a unique Frame-Object Analysis Methodology for advanced modelling. It also shows the reader what constitutes an advanced distributed information system and how to design and implement one. The handbook will benefit database analysts, database administrators, programmers and members of technical staff interested in data models. Andeleigh is the author of UNIX SYSTEM ARCHITECTURE.

Computer Science Workbench is a monograph series which will provide you with an in-depth working knowledge of current developments in computer technology. Every volume in this series will deal with a topic of importance in computer science and elaborate on how you yourself can build systems related to the main theme. You will be able to develop a variety of systems, including computer software tools, computer graphics, computer animation, database management systems, and computer-aided design and manufacturing systems. Computer Science Workbench represents an important new contribution in the field of practical computer technology. Tosiyasu L. Kunii Preface The goal of this book is to give concrete answers to questions such as what object oriented databases are, why they are needed, how they are implemented, and how they are applied, by describing a research prototype object-oriented database system called Jasmine. That is, this book is aimed at creating a consistent view to object-oriented databases. The contents of this book are directly based on the results of the Jasmine project conducted at Fujitsu Laboratories, Ltd. The book is a polished version of my doctoral dissertation, which includes research papers which I have authored and published.

This book is intended for Graduate and Post-graduate students in Computer Science and Engineering, Information Technology for the purpose of Object Oriented System Analysis and Design. This book covers details of UML (Unified Modeling Language) which is used to model software intensive systems.

Provides information on analyzing, designing, and writing object-oriented software.

This is billed as the only book that puts all the features of the UML notation system into the context of a fully developed example—an order processing system. Contains the unique insights of an experienced consultant who has coached companies on object-oriented design and programming.

Object-oriented analysis and design (OOAD) has over the years, become a vast field, encompassing such diverse topics as design process and principles, documentation tools, refactoring, and design and architectural patterns. For most students the learning experience is incomplete without implementation. This new textbook provides a comprehensive introduction to OOAD. The salient points of its coverage are: • A sound footing on object-oriented concepts such as classes, objects, interfaces, inheritance, polymorphism, dynamic linking, etc. • A good introduction to the stage of requirements analysis. • Use of UML to document user requirements and design. • An extensive treatment of the design process. • Coverage of implementation issues. • Appropriate use of design and architectural patterns. • Introduction to the art and craft of refactoring. • Pointers to resources that further the reader’s knowledge. All the main case-studies used for this book have been implemented by the authors using Java. The text is liberally peppered with snippets of code, which are short and fairly self-explanatory and easy to read. Familiarity with a Java-like syntax and a broad understanding of the structure of Java would be helpful in using the book to its full potential.

An introduction to powerful methods for accurate and complete system analysis and specification.

With this book, software engineers, project managers, and tool builders will be able to better understand the role of analysis and design in the object-oriented (OO) software development process. This book presents a minimum set of notions and shows the reader how to use these notions for OO software construction. The emphasis is on development principles and implementation.

Object-Oriented Design with UML and Java provides an integrated introduction to object-oriented design with the Unified Modelling Language (UML) and the Java programming language. The book demonstrates how Java applications, no matter how small, can benefit from some design during their construction. Fully road-tested by students on the authors' own courses, the book shows how these complementary technologies can be used effectively to create quality software. It requires no prior knowledge of object orientation, though readers must have some experience of Java or other high level programming language. This book covers object technology; object-oriented analysis and design; and implementation of objects with Java. It includes two case studies dealing with library applications. The UML has been incorporated into a graphical design tool called ROME, which can be downloaded from the book's website. This object modelling environment allows readers to prepare and edit various UML diagrams. ROME can be used alongside a Java compiler to generate Java code from a UML class diagram then compile and run the resulting application for hands-on learning. This text would be a valuable resource for undergraduate students taking courses on O-O analysis and design, O-O modelling, Java programming, and modelling with UML. * Integrates design and implementation, using Java and UML * Includes case studies and exercises * Bridges the gap between programming texts and high level analysis books on design
Overview: This text will be the first to present an object-oriented methodology from the outset for beginning Systems Analysis and Design students. It is the first book to introduce object-oriented methods without relying on classical methods to introduce key concepts or without requiring students to know Java or C++. It will presume no knowledge whatsoever about process modeling or data modeling. The widely used UML notation (unified modeling language) will be used throughout the book for all diagrams and model renderings. The key benefit to this approach is that it makes the course easier to teach and learn since many students come to this course with limited backgrounds having only taken one introductory MIS course. Also, this approach is appealing because object-oriented methodology is widely used in industry.

Appropriate for all introductory level courses on object-oriented system analysis, design, and/or programming. This book systematically introduces the concepts and methods of object-oriented systems analysis and design to students with little or no object experience. Rigorous yet extremely readable, it introduces the entire process of information system design, providing a thorough grounding in object-oriented techniques, UML, and step-by-step system development. Two of the field's most experienced instructors carefully link information systems analysis and design issues to general systems theory, offering a domain-independent view of design that maintains a clear conceptual distinction between requirements and design. After introducing basic systems concepts and the Rational Unified Process, they turn to object-oriented analysis, covering business event analysis, use cases, system sequence diagrams, domain modeling, and more. Part III focuses on system design, including overall system design based on a three-tier architecture, object-oriented program design, communication between the application layer and database, and user interface design. Finally, in Part IV, the authors offer a practical, real-world discussion of both information gathering and software project management. To support effective learning, every chapter begins with clear learning objectives and ends with summaries, lists of key terminology, review materials, exercises, discussion points, and wherever appropriate, case studies for project assignments.

Copyright code: 55be78f11a987b720dc6e1ab0105c1c9