oo analysis

OO Analysis: A Deep Dive into Object-Oriented Analysis Techniques

Introduction:

Are you ready to unlock the power of object-oriented programming? Understanding Object-Oriented Analysis (OOA) is crucial for building robust, scalable, and maintainable software systems. This comprehensive guide will demystify OOA, exploring its core principles, methodologies, and best practices. We'll journey from foundational concepts to advanced techniques, equipping you with the knowledge to effectively design and implement complex software projects. Forget confusing jargon; we'll break down everything in clear, concise language, making OOA accessible to developers of all skill levels. Prepare to master the art of OO analysis and revolutionize your approach to software development.

What is Object-Oriented Analysis (OOA)?

Object-Oriented Analysis (OOA) is a systematic approach to software development that focuses on identifying and modeling real-world entities as "objects." These objects possess properties (attributes) and behaviors (methods) that interact to fulfill the system's requirements. Unlike procedural programming, which emphasizes a sequence of steps, OOA emphasizes the interaction and relationships between objects. This object-centric perspective facilitates better understanding, design, and maintainability of complex software systems. It promotes modularity, reusability, and easier modification compared to traditional procedural approaches.

Key Principles of OOA:

Abstraction: Hiding complex implementation details and presenting only essential information to the user. Think of a car: you interact with the steering wheel, accelerator, and brakes, without needing to understand the intricate workings of the engine.
Encapsulation: Bundling data (attributes) and methods that operate on that data within a single unit (object). This protects data integrity and promotes modularity. Changes to internal object structure don't necessarily affect other parts of the system.
Inheritance: Creating new objects (classes) based on existing ones, inheriting their properties and behaviors. This promotes code reusability and reduces redundancy. A "sports car" inherits properties from a general "car" class, adding its own unique attributes (e.g., spoiler).
Polymorphism: The ability of objects of different classes to respond to the same method call in their own specific way. This allows for flexible and adaptable software designs. For example, a "draw()" method could be implemented differently for a "circle" object and a "square" object.

Methodologies for Conducting OO Analysis:

Several methodologies guide the OO analysis process. Some popular ones include:

UML (Unified Modeling Language): A standardized visual language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML diagrams, such as class diagrams, use case diagrams, and sequence diagrams, are essential tools for visualizing object interactions and system behavior.
CRC Cards (Class-Responsibility-Collaborator): A simple, lightweight technique involving index cards to define classes, their responsibilities, and their collaborations with other classes. This is a great method for brainstorming and early-stage design.
Use Case Modeling: Identifying and describing the interactions between users (actors) and the system to fulfill specific goals (use cases). This helps define the functional requirements of the system from a user's perspective.

Steps in the OO Analysis Process:

1. Requirements Gathering: Thoroughly understand the system's purpose and requirements through discussions with stakeholders, analyzing documentation, and creating user stories.

2. Object Identification: Identify the key objects and their attributes (data) based on the requirements gathered. Look for nouns in the requirements documentation, as they often represent objects.

3. Relationship Modeling: Define the relationships between the identified objects. These relationships might include association (general connection), aggregation (part-of), or composition (strong part-of).

4. Dynamic Modeling: Model the behavior of the objects and their interactions using techniques like sequence diagrams or state diagrams. This shows how objects interact over time to achieve specific tasks.

5. Validation and Refinement: Review and refine the models, ensuring they accurately represent the system requirements and are consistent. Iterative refinement is crucial for successful OOA.

Benefits of Using OO Analysis:

Improved Maintainability: Modular design makes it easier to modify and extend the system without affecting other parts.
Increased Reusability: Inheritance and polymorphism promote the reuse of code, saving development time and resources.
Enhanced Flexibility: Adapting to changing requirements becomes easier with a well-designed object-oriented system.
Better Collaboration: UML diagrams and other visual models facilitate communication and collaboration among developers and stakeholders.

Example: A Simple Library System

Let's consider a simple library system. Through OOA, we might identify objects like:

Book: Attributes: title, author, ISBN, availability. Methods: borrow(), return().
Member: Attributes: memberID, name, address. Methods: borrowBook(), returnBook().
Librarian: Attributes: librarianID, name. Methods: addBook(), removeBook(), manageMembers().

These objects interact to fulfill the system's functions. For instance, a member uses the `borrowBook()` method to borrow a book, and the librarian uses the `addBook()` method to add new books to the library's catalog.

Case Study Outline: Developing a Simple E-commerce System Using OO Analysis

I. Introduction: Brief overview of the e-commerce system and the application of OOA.

II. Requirements Gathering: Defining the core functionalities of the e-commerce platform (user accounts, product catalog, shopping cart, payment gateway integration).

III. Object Identification and Modeling: Identifying key objects like Customer, Product, Order, Payment, ShoppingCart, and their attributes and methods. Creating class diagrams using UML.

IV. Dynamic Modeling: Illustrating the interaction between objects using sequence diagrams. For example, the sequence of events involved in placing an order.

V. Conclusion: Summarizing the process and highlighting the advantages of using OOA for developing the e-commerce system.

Case Study Details:

(I) Introduction: This case study demonstrates how OOA principles can be applied to develop a robust and scalable e-commerce system. We will focus on identifying key objects and their interactions, illustrating the power of OOA in simplifying complex software design.

(II) Requirements Gathering: The system must allow users to: create accounts, browse products, add products to a shopping cart, proceed to checkout, provide payment information, and receive order confirmation. The system also needs inventory management capabilities for administrators.

(III) Object Identification and Modeling:

Customer: Attributes: customerID, name, address, email, password. Methods: createAccount(), login(), viewOrders(), updateProfile().
Product: Attributes: productID, name, description, price, quantityInStock. Methods: updateQuantity().
Order: Attributes: orderID, customerID, orderDate, totalAmount, orderItems. Methods: calculateTotal().
ShoppingCart: Attributes: items. Methods: addItem(), removeItem(), checkout().
Payment: Attributes: paymentID, amount, paymentMethod. Methods: processPayment().

These objects and their relationships would be visually represented using a UML class diagram.

(IV) Dynamic Modeling: A sequence diagram would depict the interaction between objects during the checkout process. For instance, the sequence might involve the `ShoppingCart` interacting with the `Order` and `Payment` objects to complete the transaction.

(V) Conclusion: This case study showcases how OOA effectively structures a complex e-commerce system. The identified objects and their interactions create a modular, maintainable, and scalable system, highlighting the power of object-oriented design principles.

FAQs:

1. What is the difference between OOA and OOD (Object-Oriented Design)? OOA focuses on analyzing and modeling the problem domain, while OOD focuses on designing the software solution based on the OOA model.

2. What are the limitations of OOA? OOA can be complex for very large and intricate systems, and requires experienced developers to utilize effectively.

3. Is UML essential for OOA? While UML is a powerful tool, other methods like CRC cards can be used for smaller projects.

4. How do I choose the right OOA methodology? The choice depends on the project's size, complexity, and team experience.

5. Can OOA be used for non-software systems? Yes, OOA principles can be applied to model any system composed of interacting entities.

6. What are some common mistakes in OOA? Overlooking crucial objects, failing to properly define relationships, and neglecting dynamic modeling are common pitfalls.

7. How do I improve my OOA skills? Practice, studying case studies, and using modeling tools are effective ways to improve.

8. What are the best resources for learning OOA? Numerous books, online courses, and tutorials are available.

9. Is OOA still relevant in the age of agile development? Yes, OOA principles remain highly relevant; agile methodologies often incorporate OOA techniques.

Related Articles:

1. UML Class Diagrams: A Beginner's Guide: Explains the fundamentals of creating UML

class diagrams.

2. Use Case Modeling for Software Development: Details how to create effective use case models.

3. Object-Oriented Design Principles: A deep dive into SOLID principles and other design patterns.

4. Agile Development and OOA: Exploring the synergy between agile methodologies and object-oriented analysis.

5. CRC Cards: A Practical Approach to Object-Oriented Design: A step-by-step guide to using CRC cards.

6. Sequence Diagrams in UML: Learn how to create and interpret sequence diagrams.

7. State Diagrams in UML: Understanding how to model object states and transitions.

8. Best Practices for Object-Oriented Analysis and Design: Tips for writing clean, efficient, and maintainable object-oriented code.

9. Object-Oriented Programming Languages: A Comparison: Comparing popular OOP languages like Java, C++, Python, and C#.

oo analysis: Object-Oriented Analysis and Design with Applications Grady Booch, Robert Maksimchuk, Michael Engle, Jim Conallen, Kelli Houston, Bobbi Young Ph.D., 2007-04-30 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 mange 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 Tracking System Appendix A: Object-Oriented Programming Languages Appendix B: Further Reading Notes Glossary Classified Bibliography Index

oo analysis: Object-oriented Analysis and Design with Applications Grady Booch, 1994 This revision of Grady Booch's classic offers the first industry-wide standard for notation in developing large scale object-oriented systems. Laying the groundwork for the development of complex systems based on the object model, the author works in C++ to provide five fully-developed design examples, along with many smaller applications. Three of these capstone projects are new with this edition, including an inventory tracking system which implements a client server. The other four span problem domains as diverse as data acquisition for scientific tools, framework, artificial intelligence, and command and control. To measure progress, metrics in object development are suggested so that the developer knows how the project is going. In addition, the author demonstrates good and bad object designs and shows how to manage the trade-offs in complex systems.

oo analysis: Object Oriented Data Analysis J. S. Marron, Ian L. Dryden, 2021-11-18 Object Oriented Data Analysis is a framework that facilitates inter-disciplinary research through new terminology for discussing the often many possible approaches to the analysis of complex data. Such data are naturally arising in a wide variety of areas. This book aims to provide ways of thinking that enable the making of sensible choices. The main points are illustrated with many real data examples, based on the authors' personal experiences, which have motivated the invention of a wide array of analytic methods. While the mathematics go far beyond the usual in statistics (including differential geometry and even topology), the book is aimed at accessibility by graduate students. There is deliberate focus on ideas over mathematical formulas. J. S. Marron is the Amos Hawley Distinguished Professor of Statistics, Professor of Biostatistics, Adjunct Professor of Computer Science, Faculty Member of the Bioinformatics and Computational Biology Curriculum and Research Member of the Lineberger Cancer Center and the Computational Medicine Program, at the University of North Carolina, Chapel Hill. Ian L. Dryden is a Professor in the Department of Mathematics and Statistics at Florida International University in Miami, has served as Head of School of Mathematical Sciences at the University of Nottingham, and is joint author of the acclaimed book Statistical Shape Analysis.

oo analysis: Object-oriented Systems Analysis Sally Shlaer, Stephen J. Mellor, 1988 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.

oo analysis: Head First Object-Oriented Analysis and Design Brett McLaughlin, Gary Pollice, David West, 2007 Provides information on analyzing, designing, and writing object-oriented software.

oo analysis: Object-Oriented Analysis and Design Sarnath Ramnath, Brahma Dathan, 2010-12-06 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.

oo analysis: Object-oriented Analysis Peter Coad, Edward Yourdon, 1990 A complete implementation guide to a new requirements analysis technique, based on an object-oriented paradigm, offering numerous case studies and step-by-step examples.

oo analysis: *Head First Object-Oriented Analysis and Design* Brett McLaughlin, Gary Pollice, David West, 2006-11-27 Head First Object Oriented Analysis and Design is a refreshing look at subject of OOAD. What sets this book apart is its focus on learning. The authors have made the content of OOAD accessible, usable for the practitioner. Ivar Jacobson, Ivar Jacobson Consulting I just finished reading HF OOA&D and I loved it! The thing I liked most about this book was its focus on why we do OOA&D-to write great software! Kyle Brown, Distinguished Engineer, IBM Hidden behind the funny pictures and crazy fonts is a serious, intelligent, extremely well-crafted presentation of OO Analysis and Design. As I read the book, I felt like I was looking over the shoulder of an expert designer who was explaining to me what issues were important at each step, and why. Edward Sciore,Associate Professor, Computer Science Department, Boston College Tired of reading Object Oriented Analysis and Design books that only makes sense after you're an expert? You've heard OOA&D can help you write great software every time-software that makes your boss happy, your customers satisfied and gives you more time to do what makes you happy. But how? Head First Object-Oriented Analysis & Design shows you how to analyze, design, and write serious object-oriented software: software that's easy to reuse, maintain, and extend; software that doesn't hurt your head; software that lets you add new features without breaking the old ones. Inside you will learn how to: Use OO principles like encapsulation and delegation to build applications that are flexible Apply the Open-Closed Principle (OCP) and the Single Responsibility Principle (SRP) to promote reuse of your code Leverage the power of design patterns to solve your problems more efficiently Use UML, use cases, and diagrams to ensure that all stakeholders arecommunicating clearly to help you deliver the right software that meets everyone's needs. By exploiting how your brain works, Head First Object-Oriented Analysis & Design compresses the time it takes to learn and retain complex information. Expect to have fun, expect to learn, expect to be writing great software consistently by the time you're finished reading this!

oo analysis: *UML 2 and the Unified Process* Jim Arlow, Ila Neustadt, 2005-06-27 This book manages to convey the practical use of UML 2 in clear and understandable terms with many examples and guidelines. Even for people not working with the Unified Process, the book is still of great use. UML 2 and the Unified Process, Second Edition is a must-read for every UML 2 beginner and a helpful guide and reference for the experienced practitioner. --Roland Leibundgut, Technical Director, Zuehlke Engineering Ltd. This book is a good starting point for organizations and individuals who are adopting UP and need to understand how to provide visualization of the different aspects needed to satisfy it. --Eric Naiburg, Market Manager, Desktop Products, IBM Rational Software This thoroughly revised edition provides an indispensable and practical guide to the complex process of object-oriented analysis and design using UML 2. It describes how the process of OO analysis and design fits into the software development lifecycle as defined by the Unified Process (UP). UML 2 and the Unified Process contains a wealth of practical, powerful, and useful techniques that you can apply immediately. As you progress through the text, you will learn OO analysis and design techniques, UML syntax and semantics, and the relevant aspects of the UP. The book provides you with an accurate and succinct summary of both UML and UP from the point of view of the OO analyst and designer. This book provides Chapter roadmaps, detailed diagrams, and margin notes allowing you to focus on your needs Outline summaries for each chapter, making it ideal for revision, and a comprehensive index that can be used as a reference New to this edition: Completely revised and updated for UML 2 syntax Easy to understand explanations of the new UML 2 semantics More real-world examples A new section on the Object Constraint Language (OCL) Introductory

material on the OMG's Model Driven Architecture (MDA) The accompanying website provides A complete example of a simple e-commerce system Open source tools for requirements engineering and use case modeling Industrial-strength UML course materials based on the book

oo analysis: Object-oriented Analysis and Design James Martin, James J. Odell, 1992 This guide covers the underlying philosophy of object orientation and demonstrates its practical usage, exploring both the analysis and the design phases of applying object-oriented techniques. The authors use an innovative approach based not on reality, but rather the way reality is understood by people (not computers). Topics covered include project management of object-oriented programs, making the transition from 00 analysis to 00 design, 00 databases and AI tools.

oo analysis: Object-Oriented Analysis and Design for Information Systems Raul Sidnei Wazlawick, 2014-01-28 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 understand 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.

oo analysis: Object Oriented Analysis & Design With Application Grady Booch, 2006-02

oo analysis: Advanced Object-Oriented Analysis and Design Using UML James J. Odell, 1998-02-13 This 1998 book conveys the essence of object-oriented programming and software building through the Unified Modeling Language.

oo analysis: Analysis Patterns Martin Fowler, 1997 Martin Fowler is a consultant specializing in object-oriented analysis and design. This book presents and discusses a number of object models derived from various problem domains. All patterns and models presented have been derived from the author's own consulting work and are based on real business cases.

oo analysis: Object-oriented Analysis and Design John Deacon, 2005 John Deacon's in-depth, highly pragmatic approach to object-oriented analysis and design, demonstrates how to lay the foundations for developing the best possible software. Students will learn how to ensure that analysis and design remain focused and productive. By working through the book, they will gain a solid working knowledge of best practices in software development. The focus of the text is on typical development projects and technologies, showing exactly what the different development activities are, and emphasising what they should and should not be trying to accomplish. This fresh, comprehensive examination of object-oriented analysis and design in the context of today's systems and technologies will be a valuable addition to the bookshelves of undergraduates and graduates on systems analysis and design courses.

oo analysis: Object-oriented Analysis & Design Andrew Haigh, 2001 This volume provides an exploration of the four stages of software development: analysis, design, implementation, and troubleshooting. Appropriate for programmers of all levels, it contains both working examples and design concepts using non-technical language.

oo analysis: Functional and Object Oriented Analysis and Design: An Integrated Methodology Shoval, Peretz, 2006-07-31 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.

oo analysis: *Developing Software with UML* Bernd Oestereich, 2002 This book shows us how to use UML and apply it in object-oriented software development. Part 1 of the book guides the reader step-by-step through the development process while part 2 explains the basics of UML in detail.

oo analysis: Ebook: Object-Oriented Systems Analysis and Design Using UML BENNETT, 2010-04-16 Ebook: Object-Oriented Systems Analysis and Design Using UML

oo analysis: Object-oriented Analysis and Design with the Unified Process John W. Satzinger, Robert B. Jackson, Stephen D. Burd, 2005 This pure Object-Oriented approach gives students a cutting edge approach to the future of the design and analysis market.

oo analysis: Aliasing in Object-Oriented Programming David Clarke, Tobias Wrigstad, James Noble, 2013-03-21 This book presents a survey of the state-of-the-art on techniques for dealing with aliasing in object-oriented programming. It marks the 20th anniversary of the paper The Geneva Convention On The Treatment of Object Aliasing by John Hogg, Doug Lea, Alan Wills, Dennis de Champeaux and Richard Holt. The 22 revised papers were carefully reviewed to ensure the highest quality.The contributions are organized in topical sections on the Geneva convention, ownership, concurrency, alias analysis, controlling effects, verification, programming languages, and visions.

oo analysis: Object-Oriented Analysis and Design Using UML MAHESH P. MATHA, 2008-04-09 A modern computer program, such as the one that controls a rocket's journey to moon, is like a medieval cathedral—vast, complex, layered with circuits and mazes. To write such a program, which probably runs into a hundred thousand lines or more, knowledge of an object-oriented language like Java or C++ is not enough. Unified Modelling Language (UML), elaborated in detail in this book, is a methodology that assists in the design of software systems. The first task in the making of a software product is to gather requirements from the client. This well-organized and clearly presented text develops a formal method to write down these requirements as Use Cases in UML. Besides, it also develops the concepts of static and dynamic modelling and the Unified Process that suggests incremental and iterative development of software, taking client feedback at every step. The concept of Design Patterns which provide solutions to problems that occur repeatedly during software development is discussed in detail in the concluding chapters. Two appendices provide solutions to two real-life problems. Case Studies, mapping of examples into Java code that are executable on computers, summary and Review Questions at the end of every chapter make the book reader friendly. The book will prove extremely useful to undergraduate and postgraduate students of Computer Science and Engineering, Information Technology, and Master of Computer Applications (MCA). It will also benefit professionals who wish to sharpen their programming skills using UML.

oo analysis: Applying UML and Patterns Training Course Craig Larman, 2002-07-01 Second Edition of the UML video course based on the book Applying UML and Patterns. This VTC will focus on object-oriented analysis and design, not just drawing UML.

oo analysis: Systems Analysis and Design Alan Dennis, Barbara Wixom, David Tegarden, 2020-11-17 Systems Analysis and Design: An Object-Oriented Approach with UML, Sixth Edition helps students develop the core skills required to plan, design, analyze, and implement information systems. Offering a practical hands-on approach to the subject, this textbook is designed to keep students focused on doing SAD, rather than simply reading about it. Each chapter describes a specific part of the SAD process, providing clear instructions, a detailed example, and practice exercises. Students are guided through the topics in the same order as professional analysts working on a typical real-world project. Now in its sixth edition, this edition has been carefully updated to reflect current methods and practices in SAD and prepare students for their future roles as systems analysts. Every essential area of systems analysis and design is clearly and thoroughly covered, from project management, to analysis and design modeling, to construction, installation, and operations. The textbook includes access to a range of teaching and learning resources, and a running case study of a fictitious healthcare company that shows students how SAD concepts are applied in real-life scenarios.

oo analysis: Multispectral Image Analysis Using the Object-Oriented Paradigm Kumar

Navulur, 2006-12-05 Bringing a fresh new perspective to remote sensing, object-based image analysis is a paradigm shift from the traditional pixel-based approach. Featuring various practical examples to provide understanding of this new modus operandi, Multispectral Image Analysis Using the Object-Oriented Paradigm reviews the current image analysis methods and demonstrates advantages to improve information extraction from imagery. This reference describes traditional image analysis techniques, introduces object-oriented technology, and discusses the benefits of object-based versus pixel-based classification. It examines the creation of object primitives using image segmentation approaches and the use of various techniques for object classification. The author covers image enhancement methods, how to use ancillary data to constrain image segmentation, and concepts of semantic grouping of objects. He concludes by addressing accuracy assessment approaches. The accompanying downloadable resources present sample data that enable the use of different approaches to problem solving. Integrating remote sensing techniques and GIS analysis, Multispectral Image Analysis Using the Object-Oriented Paradigm distills new tools to extract information from remotely sensed data.

oo analysis: Object-Oriented Design with UML and Java Kenneth Barclay, John Savage, 2003-12-17 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

oo analysis: *Object-Oriented Analysis and Design with Applications, 3/e, Professional Edition (HB).* Booch, 1900 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 sys.

oo analysis: *Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Interative Development: 3rd Edition* Craig Larman, 2012

oo analysis: Object-Based Image Analysis Thomas Blaschke, Stefan Lang, Geoffrey Hay, 2008-08-09 This book brings together a collection of invited interdisciplinary persp- tives on the recent topic of Object-based Image Analysis (OBIA). Its c- st tent is based on select papers from the 1 OBIA International Conference held in Salzburg in July 2006, and is enriched by several invited chapters. All submissions have passed through a blind peer-review process resulting in what we believe is a timely volume of the highest scientific, theoretical and technical standards. The concept of OBIA first gained widespread interest within the GIScience (Geographic Information Science) community circa 2000, with the advent of the first commercial software for what was then termed ‘obje- oriented image analysis’. However, it is widely agreed that OBIA builds on older segmentation,

edge-detection and classification concepts that have been used in remote sensing image analysis for several decades. Nevert- less, its emergence has provided a new critical bridge to spatial concepts applied in multiscale landscape analysis, Geographic Information Systems (GIS) and the synergy between image-objects and their radiometric char- teristics and analyses in Earth Observation data (EO).

oo analysis: Object-oriented Systems Analysis and Design Ronald J. Norman, 1996 Evolutionary in approach, this book explores informatino systems development--both analysis and design--using an object-oriented methodology combined with a relational database as part of the implementation.

oo analysis: OOPSLA ECOOP '90 Jerry L. Archibald, 1991

oo analysis: Magnifying Object-oriented Analysis and Design GOPAL ARPITA, Patil Netra, 2010-11 A firm grounding in the theory of object-oriented analysis and design and its practical application is essential for understanding how to build good software. This book, the third of the Magnifying Series, attempts to explain the object-oriented analysis and design of software through case studies covering various business domains. The book describes various software development models and techniques before introducing the concepts and principles of object-oriented analysis and design. It explains analysis models with the help of business process diagrams, use-case diagrams, class diagrams and object diagrams. The book elaborates design models through sequence diagrams, collaboration diagrams, statechart diagrams and activity diagrams. It also deals with implementation models with the help of component and deployment diagrams. For each diagram, its purpose, notations and design guidelines are given. In addition, the book explains existing object-oriented methodologies. KEY FEATURES: Develops a framework for analysis of business cases followed by design of software solutions for them. Includes several case studies to depict the application of object-oriented analysis and design. Presents chapter-end exercises for the students' comprehension of the subject matter. The text is designed for the students of computer applications (BCA/MCA), computer science (B.Sc./M.Sc.), and computer science and engineering (BE/B.Tech).

oo analysis: Object-oriented Analysis and Design James Martin, James J. Odell, 1992 This guide covers the underlying philosophy of object orientation and demonstrates its practical usage, exploring both the analysis and the design phases of applying object-oriented techniques. The authors use an innovative approach based not on reality, but rather the way reality is understood by people (not computers). Topics covered include project management of object-oriented programs, making the transition from 00 analysis to 00 design, 00 databases and AI tools.

oo analysis: *Entity-Relationship Approach - ER '93* Ramez A. Elmasri, 1994-07-28 This monograph is devoted to computational morphology, particularly to the construction of a two-dimensional or a three-dimensional closed object boundary through a set of points in arbitrary position. By applying techniques from computational geometry and CAGD, new results are developed in four stages of the construction process: (a) the gamma-neighborhood graph for describing the structure of a set of points; (b) an algorithm for constructing a polygonal or polyhedral boundary (based on (a)); (c) the flintstone scheme as a hierarchy for polygonal and polyhedral approximation and localization; (d) and a Bezier-triangle based scheme for the construction of a smooth piecewise cubic boundary.

oo analysis: 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.

oo analysis: Software Abstractions Daniel Jackson, 2012 An approach to software design that introduces a fully automated analysis giving designers immediate feedback, now featuring the latest version of the Alloy language. In Software Abstractions Daniel Jackson introduces an approach to software design that draws on traditional formal methods but exploits automated tools to find flaws as early as possible. This approach—which Jackson calls “lightweight formal methods” or “agile modeling”—takes from formal specification the idea of a precise and expressive notation based on a

tiny core of simple and robust concepts but replaces conventional analysis based on theorem proving with a fully automated analysis that gives designers immediate feedback. Jackson has developed Alloy, a language that captures the essence of software abstractions simply and succinctly, using a minimal toolkit of mathematical notions. This revised edition updates the text, examples, and appendixes to be fully compatible with Alloy 4.

oo analysis: Object-oriented Analysis & Design Lars Mathiassen, 2000

oo analysis: The Object-oriented Thought Process Matt A. Weisfeld, 2009 The Object-Oriented Thought Process Third Edition Matt Weisfeld An introduction to object-oriented concepts for developers looking to master modern application practices. Object-oriented programming (OOP) is the foundation of modern programming languages, including C++, Java, C#, and Visual Basic .NET. By designing with objects rather than treating the code and data as separate entities, OOP allows objects to fully utilize other objects' services as well as inherit their functionality. OOP promotes code portability and reuse, but requires a shift in thinking to be fully understood. Before jumping into the world of object-oriented programming languages, you must first master The Object-Oriented Thought Process. Written by a developer for developers who want to make the leap to object-oriented technologies as well as managers who simply want to understand what they are managing, The Object-Oriented Thought Process provides a solution-oriented approach to object-oriented programming. Readers will learn to understand object-oriented design with inheritance or composition, object aggregation and association, and the difference between interfaces and implementations. Readers will also become more efficient and better thinkers in terms of object-oriented development. This revised edition focuses on interoperability across various technologies, primarily using XML as the communication mechanism. A more detailed focus is placed on how business objects operate over networks, including client/server architectures and web services. Programmers who aim to create high quality software-as all programmers should-must learn the varied subtleties of the familiar yet not so familiar beasts called objects and classes. Doing so entails careful study of books such as Matt Weisfeld's The Object-Oriented Thought Process. -Bill McCarty, author of Java Distributed Objects, and Object-Oriented Design in Java Matt Weisfeld is an associate professor in business and technology at Cuyahoga Community College in Cleveland, Ohio. He has more than 20 years of experience as a professional software developer, project manager, and corporate trainer using C++, Smalltalk, .NET, and Java. He holds a BS in systems analysis, an MS in computer science, and an MBA in project management. Weisfeld has published many articles in major computer trade magazines and professional journals.

oo analysis: Object-oriented Software Construction Bertrand Meyer, 1988 Software -- Software Engineering.

oo analysis: Object Oriented Technologies: Opportunities and Challenges Gibson, Rick, 1999-07-01 The continual evolution of object oriented technologies creates both opportunities and challenges. However, despite the growing popularity of object oriented technology, there are numerous issues that have contributed to its inability to firmly entrench itself and take over for the older, proven technologies. Object oriented technology's image problem has created a highly difficult decision making process for corporations considering widespread adoption of these technologies. Object Oriented Technologies: Opportunities and Challenges addresses concerns, opportunities and technology trends in the application of object oriented technologies. The chapters of this book were selected to represent a variety of perspectives concerning the present and future of this broad sub-field of software development.

Additional Resources

oo analysis

Oo Analysis

Oo Analysis

Related Articles

- ohio state business ranking
- phet states of matter basics answer key
- of mice and men book analysis

Back to Home