Refactoring Databases Evolutionary Database Design 1st First Edition By Ambler Scott W Sadalage Pramodkumar J Published By Addison Wesley Professional 2006

This IBM® Redbooks® publication is focused on melding industry preferred practices with the unique needs of the IBM i community and providing a holistic view of modernization. This book covers key trends for application structure, user interface, data access, and the database. Modernization is a broad term when applied to applications. It is more than a single event. It is a sequence of actions. But even more, it is a process of rethinking how to approach the creation and maintenance of applications. There are tangible deliveries when it comes to modernization, the most notable being a modern user interface (UI), such as a web browser or being able to access applications from a mobile device. The UI, however, is only the beginning. There are many more aspects to modernization. Using modern tools and methodologies can significantly improve productivity and reduce long-term cost while positioning applications for the next decade. It is time to put the past away. Tools and methodologies have undergone significant transformation, improving functionality, usability, and productivity. This is true of the plethora of IBM tools and the wealth of tools available from many Independent Solution Providers (ISVs). This publication is the result of work that was done by IBM, industry experts, and by representatives from many of the ISV Tool Providers. Some of their tools are referenced in the book. In addition to reviewing technologies based on context, there is an explanation of why modernization is important and a description of the business benefits of investing in modernization. This critical information is key for line-of-business executives who want to understand the benefits of a modernization project. This book is appropriate for CIOs, architects, developers, and business leaders. Related information Making the Case for Modernization, IBM Systems Magazine

Discover how graph databases can help you manage and query highly connected data. With this practical book, you'll learn how to design and implement a graph database that brings the power of graphs to bear on a broad range of problem domains. Whether you want to speed up your response to user queries or build a database that can adapt as your business evolves, this book shows you how to apply the schema-free graph model to real-world problems. Learn how different organizations are using graph databases to outperform their competitors. With this book's data modeling, query, and code examples, you'll quickly be able to implement your own solution. Model data with the Cypher query language and property graph model Learn best practices and common pitfalls when modeling with graphs Plan and implement a graph database solution in test-driven fashion Explore real-world examples to learn how and why organizations use a graph database Understand common patterns and components of graph database architecture Use analytical techniques and algorithms to mine graph database information

For any software developer who has spent days in "integration hell," cobbling together myriad software components, Continuous Integration: Improving Software Quality and Reducing Risk illustrates how to transform integration from a necessary evil into an everyday part of the development process. The key, as the authors show, is to integrate regularly and often using continuous integration (CI) practices and techniques. The authors first examine the concept of CI and its practices from the ground up and then move on to explore other effective processes performed by CI systems, such as database integration, testing, inspection, deployment, and feedback. Through more than forty CI-related practices using application examples in different languages, readers learn that CI leads to more rapid software development, produces deployable software at every step in the development lifecycle, and reduces the time between defect introduction and detection, saving time and lowering costs. With successful implementation of CI, developers reduce risks and repetitive manual processes, and teams receive better project visibility. The book covers How to make integration a "non-event" on your software development projects How to reduce the amount of repetitive processes you perform when building your software Practices and techniques for using CI effectively with your teams Reducing the risks of late defect discovery, low-quality software, lack of visibility, and lack of deployable software Assessments of different CI servers and related tools on the market The book's companion Web site, www.integratebutton.com, provides updates and code examples.

Microservices is an architectural style in which large, complex software applications are composed of one or more smaller services. Each of these microservices focuses on completing one task that represents a small business capability. These microservices can be developed in any programming language. This IBM® Redbooks® publication shows how to break out a traditional Java EE application into separate microservices and provides a set of code projects that illustrate the various steps along the way. These code projects use the IBM WebSphere® Application Server Liberty, IBM API ConnectTM, IBM Bluemix®, and other Open Source Frameworks in the microservices ecosystem. The sample projects highlight the evolution of monoliths to microservices with Java and Node.

Database Design for Mere Mortals

Support Constant Change

32nd International Conference, CAiSE 2020, Grenoble, France, June 8-12, 2020, Proceedings

Improving the Design of Existing Web Applications

The Art of Agile Development

Evolutionary Database Development (Digital Short Cut)

PSP(sm)

The Definitive Refactoring Guide, Fully Revamped for Ruby With refactoring, programmers can transform even the most chaotic software into well-designed systems that are far easier to evolve and maintain. What's more, they can do it one step at a time, through a series of simple, proven steps. Now, there's an authoritative and extensively updated version of Martin Fowler's classic refactoring book that utilizes Ruby examples and idioms throughout-not code adapted from Java or any other environment. The authors introduce a detailed catalog of more than 70 proven Ruby refactorings, with specific guidance on when to apply each of them, step-by-step

instructions for using them, and example code illustrating how they work. Many of the authors' refactorings use powerful Ruby-specific features, and all code samples are available for download. Leveraging Fowler's original concepts, the authors show how to perform refactoring in a controlled, efficient, incremental manner, so you methodically improve your code's structure without introducing new bugs. Whatever your role in writing or maintaining Ruby code, this book will be an indispensable resource. This book will help you * Understand the core principles of refactoring and the reasons for doing it * Recognize "bad smells" in your Ruby code * Rework bad designs into well-designed code, one step at a time * Build tests to make sure your refactorings work properly * Understand the challenges of refactoring and how they can be overcome * Compose methods to package code properly * Move features between objects to place responsibilities where they fit best * Organize data to make it easier to work with * Simplify conditional expressions and make more effective use of polymorphism * Create interfaces that are easier to understand and use * Generalize more effectively * Perform larger refactorings that transform entire software systems and may take months or years * Successfully refactor Ruby on Rails code
Concise and easy-to-understand guidelines and standards for creating UML 2.0 diagrams.
This book constitutes the refereed proceedings of the 32nd International Conference on Advanced Information Systems Engineering, CAiSE 2020, held in Grenoble, France, in June 2020.* The 33 full papers presented in this volume were carefully reviewed and selected from 185 submissions. The book also contains one invited talk in full paper length. The papers were organized in topical sections named: distributed applications; AI and big data in IS; process mining and analysis; requirements and modeling; and information systems engineering. Abstracts on the CAiSE 2020 tutorials can be found in the back matter of the volume. *The conference was held virtually due to the COVID-19 pandemic.
Refactoring has proven its value in a wide range of development projects–helping software professionals improve system designs, maintainability, extensibility, and performance. Now, for the first time, leading agile methodologist Scott Ambler and renowned consultant Pramodkumar Sadalage introduce powerful refactoring techniques specifically designed for database systems. Ambler and Sadalage demonstrate how small changes to table structures, data, stored procedures, and triggers can significantly enhance virtually any database design–without changing semantics. You'll learn how to evolve database schemas in step with source code–and become far more effective in projects relying on iterative, agile methodologies. This comprehensive guide and reference helps you overcome the practical obstacles to refactoring real-world databases by covering every fundamental concept underlying database refactoring. Using start-to-finish examples, the authors walk you through refactoring simple standalone database applications as well as sophisticated multi-application scenarios. You'll master every task involved in refactoring database schemas, and discover best practices for deploying refactorings in even the most complex production environments. The second half of this book systematically covers five major categories of database refactorings. You'll learn how to use refactoring to enhance database structure, data quality, and referential integrity; and how to refactor both architectures and methods. This book provides an extensive set of examples built with Oracle and Java and easily adaptable for other languages, such as C#, C++, or VB.NET, and other databases, such as DB2, SQL Server, MySQL, and Sybase. Using this book's techniques and examples, you can reduce waste, rework, risk, and cost–and build database systems capable of evolving smoothly, far into the future.
Monolith to Microservices
Graph Databases
Choose your WoW
Agile Modeling
Agile Analytics
Modernizing IBM i Applications from the Database up to the User Interface and Everything in Between
Improving Software Quality and Reducing Risk
Most software-development groups have embarrassing records: By some accounts, more than half of all software projects are significantly late and over budget, and nearly a quarter of them are cancelled without ever being completed. Although developers recognize that unrealistic schedules, inadequate resources, and unstable requirements are often to blame for such failures, few know how to solve these problems. Fortunately, the Personal Software Process (PSP) provides a clear and proven solution. Comprising precise methods developed over many years by Watts S. Humphrey and the Software Engineering Institute (SEI), the PSP has successfully transformed work practices in a wide range of organizations and has already produced some striking results. This book describes the PSP and is the definitive guide and reference for its latest iteration. PSP training focuses on the skills required by individual software engineers to improve their personal performance. Once learned and effectively applied, PSP-trained engineers are qualified to participate on a team using the Team Software Process (TSP), the methods for which are described in the final chapter of the book. The goal for both PSP and TSP is to give developers exactly what they need to deliver quality products on predictable schedules. PSPSM: A Self-Improvement Process for Software Engineers presents a disciplined process for software engineers and anyone else involved in software development. This process includes defect management, comprehensive planning, and precise project tracking and reporting. The book first scales down industrial software practices to fit the needs of the module-sized program development, then walks

readers through a progressive sequence of practices that provide a sound foundation for large-scale software development. By doing the exercises in the book, and using the PSP methods described here to plan, evaluate, manage, and control the quality of your own work, you will be well prepared to apply those methods on ever larger and more critical projects. Drawing on the author’s extensive experience helping organizations to achieve their development goals, and with the PSP benefits well illustrated, the book presents the process in carefully crafted steps. The first chapter describes overall principles and strategies. The next two explain how to follow a defined process, as well as how to gather and use the data required to manage a programming job. Several chapters then cover estimating and planning, followed by quality management and design. The last two chapters show how to put the PSP to work, and how to use it on a team project. A variety of support materials for the book, as described in the Preface, are available on the Web. If you or your organization are looking for a way to improve your project success rate, the PSP could well be your answer.

When carefully selected and used, Domain-Specific Languages (DSLs) may simplify complex code, promote effective communication with customers, improve productivity, and unclog development bottlenecks. In Domain-Specific Languages , noted software development expert Martin Fowler first provides the information software professionals need to decide if and when to utilize DSLs. Then, where DSLs prove suitable, Fowler presents effective techniques for building them, and guides software engineers in choosing the right approaches for their applications. This book’s techniques may be utilized with most modern object-oriented languages; the author provides numerous examples in Java and C#, as well as selected examples in Ruby. Wherever possible, chapters are organized to be self-standing, and most reference topics are presented in a familiar patterns format. Armed with this wide-ranging book, developers will have the knowledge they need to make important decisions about DSLs—and, where appropriate, gain the significant technical and business benefits they offer. The topics covered include: How DSLs compare to frameworks and libraries, and when those alternatives are sufficient Using parsers and parser generators, and parsing external DSLs Understanding, comparing, and choosing DSL language constructs Determining whether to use code generation, and comparing code generation strategies Previewing new language workbench tools for creating DSLs

The software development ecosystem is constantly changing, providing a constant stream of new tools, frameworks, techniques, and paradigms. Over the past few years, incremental developments in core engineering practices for software development have created the foundations for rethinking how architecture changes over time, along with ways to protect important architectural characteristics as it evolves. This practical guide ties those parts together with a new way to think about architecture and time.

The need to handle increasingly larger data volumes is one factor driving the adoption of a new class of nonrelational “NoSQL” databases. Advocates of NoSQL databases claim they can be used to build systems that are more performant, scale better, and are easier to program. NoSQL Distilled is a concise but thorough introduction to this rapidly emerging technology. Pramod J. Sadalage and Martin Fowler explain how NoSQL databases work and the ways that they may be a superior alternative to a traditional RDBMS. The authors provide a fast-paced guide to the concepts you need to know in order to evaluate whether NoSQL databases are right for your needs and, if so, which technologies you should explore further. The first part of the book concentrates on core concepts, including schemaless data models, aggregates, new distribution models, the CAP theorem, and map-reduce. In the second part, the authors explore architectural and design issues associated with implementing NoSQL. They also present realistic use cases that demonstrate NoSQL databases at work and feature representative examples using Riak, MongoDB, Cassandra, and Neo4j. In addition, by drawing on Pramod Sadalage's pioneering work, NoSQL Distilled shows how to implement evolutionary design with schema migration: an essential technique for applying NoSQL databases. The book concludes by describing how NoSQL is ushering in a new age of Polyglot Persistence, where multiple data-storage worlds coexist, and architects can choose the technology best optimized for each type of data access.

The Application Developer's Guide to Object-Orientation and the UML

Database and Expert Systems Applications

Building Evolutionary Architectures

Professional Sql Server 2000 Database Design

Introduction to Databases and Data Warehouses

Software Security

Beyond Software Architecture

& Most software practitioners deal with inherited code; this book teaches them how to optimize it & & Workbook approach facilitates the learning process & & Helps you identify where problems in a software application exist or are likely to exist

Various topics of data mining techniques are identified and described throughout, including clustering, association rules, rough set theory, probability theory, neural networks, classification, and fuzzy logic. Each of these techniques is explored with a theoretical introduction and its effectiveness is demonstrated with various chapter examples.

Describes Agile Modeling Driven Design (AMDD) and Test-Driven Design (TDD) approaches, database refactoring, database encapsulation strategies, and tools that support evolutionary techniques Agile software developers often use object and relational database (RDB) technology together and as a result must overcome the impedance mismatch The author covers techniques for mapping objects to RDBs and for implementing concurrency control, referential integrity, shared business logic, security access control, reports, and XML An agile foundation describes fundamental skills that all agile software developers require, particularly Agile DBAs Includes object modeling, UML data modeling, data normalization, class normalization, and how to deal with legacy databases Scott W. Ambler is author of Agile Modeling (0471202827), a contributing editor with Software Development (www.sdmagazine.com), and a featured speaker at software conferences worldwide

This two volume set LNCS 9827 and LNCS 9828 constitutes the refereed proceedings of the 27th International Conference on Database and Expert Systems Applications, DEXA 2016, held in Porto, Portugal, September 2016. The 39 revised full papers presented together with 29 short papers were carefully reviewed and selected from 137 submissions. The papers discuss a range of topics including: Temporal, Spatial, and High Dimensional Databases; Data Mining; Authenticity, Privacy, Security, and Trust; Data Clustering; Distributed and Big Data Processing; Decision Support Systems, and Learning; Data Streams; Data Integration, and Interoperability; Semantic Web, and Data Semantics; Social Networks, and Network Analysis; Linked Data; Data Analysis; NoSQL, NewSQL; Multimedia Data; Personal Information Management; Semantic Web and Ontologies; Database and Information System Architectures; Query Answering and Optimization; Information Retrieval, and Keyword Search; Data Modelling, and Uncertainty.

Practical Applications of Data Mining

Ruby Edition: Ruby Edition

xUnit Test Patterns

Refactoring for Software Design Smells

A Hands-on Guide to Relational Database Design

Effective Practices for eXtreme Programming and the Unified Process

Agile Model-Driven Development with UML 2.0

The acclaimed beginner's book on object technology now presents UML 2.0, Agile Modeling, and the latest in object development techniques.

"This comprehensive guide and reference helps you overcome the practical obstacles to refactoring real-world databases by covering every fundamental concept underlying database refactoring. Using start-to-finish examples, the authors walk you through refactoring simple standalone database applications as well as sophisticated multi-application scenarios. You'll master every task involved in refactoring database schemas, and discover best practices for deploying refactorings in even the most complex production environments."--Jacket.

Like any other software system, Web sites gradually accumulate “cruft” over time. They slow down. Links break. Security and compatibility problems mysteriously appear. New features don’t integrate seamlessly. Things just don’t work as well. In an ideal world, you’d rebuild from scratch. But you can’t: there’s no time or money for that. Fortunately, there’s a solution: You can refactor your Web code using easy, proven techniques, tools, and recipes adapted from the world of software development. In Refactoring HTML, Elliotte Rusty Harold explains how to use refactoring to improve virtually any Web site or application. Writing for programmers and non-programmers alike, Harold shows how to refactor for better reliability, performance, usability, security, accessibility, compatibility, and even search engine placement. Step by step, he shows how to migrate obsolete code to today’s stable Web standards, including XHTML, CSS, and REST—and eliminate chronic problems like presentation-based markup, stateful applications, and “tag soup.” The book’s extensive catalog of detailed refactorings and practical “recipes for success” are organized to help you find specific solutions fast, and get maximum benefit for minimum effort. Using this book, you can quickly improve site performance now—and make your site far easier to enhance, maintain, and scale for years to come. Topics covered include • Recognizing the “smells” of Web code that should be refactored • Transforming old HTML into well-formed, valid XHTML, one step at a time • Modernizing existing layouts with CSS • Updating old Web applications: replacing POST with GET, replacing old contact forms, and refactoring JavaScript • Systematically refactoring content and links • Restructuring sites without changing the URLs your users rely upon This book will be an indispensable resource for Web designers, developers, project managers, and anyone who maintains or updates existing sites. It will be especially helpful to Web professionals who learned HTML years ago, and want to refresh their knowledge with today’s standards-compliant best practices. This book will be an indispensable resource for Web designers, developers, project managers, and anyone who maintains or updates existing sites. It will be especially helpful to Web professionals who learned HTML years ago, and want to refresh their knowledge with today’s standards-compliant best practices.

This book, first published in 2000, Illustrates rules of Java code-writing with parallel examples of correct and incorrect usage.

Software Architecture: The Hard Parts

Refactoring HTML

NoSQL Distilled

Agile Database Techniques

Creating and Sustaining Winning Solutions

Refactoring Workbook

The Elements of UML(TM) 2.0 Style

In 1994, Design Patterns changed the landscape of object-oriented development by introducing classic solutions to recurring design problems. In 1999, Refactoring revolutionized design by introducing an effective process for improving code. With the highly anticipated Refactoring to Patterns , Joshua Kerievsky has changed our approach to design by forever uniting patterns with the evolutionary process of refactoring. This book introduces the theory and practice of pattern-directed refactorings: sequences of low-level refactorings that allow designers to safely move designs to, towards, or away from pattern implementations. Using code from real-world projects, Kerievsky documents the thinking and steps underlying over two dozen pattern-based design transformations. Along the way he offers insights into pattern differences and how to implement patterns in the simplest possible ways. Coverage includes: A catalog of twenty-seven pattern-directed refactorings, featuring real-world code examples Descriptions of twelve design smells that indicate the need for this book’s refactorings General information and new insights about patterns and refactoring Detailed implementation mechanics: how low-level refactorings are combined to implement high-level patterns Multiple ways to implement the same pattern-and when to use each Practical ways to get started even if you have little experience with patterns or

refactoring Refactoring to Patterns reflects three years of refinement and the insights of more than sixty software engineering thought leaders in the global patterns, refactoring, and agile development communities. Whether you're focused on legacy or "greenfield" development, this book will make you a better software designer by helping you learn how to make important design changes safely and effectively.

The first book to cover Agile Modeling, a new modeling techniquecreated specifically for XP projects eXtreme Programming (XP) hascreated a buzz in the software development community-much likeDesign Patterns did several years ago. Although XP presents amethodology for faster software development, many developers findthat XP does not allow for modeling time, which is critical toensure that a project meets its proposed requirements. They havealso found that standard modeling techniques that use the UnifiedModeling Language (UML) often do not work with this methodology. Inthis innovative book, Software Development columnist Scott Amblerpresents Agile Modeling (AM)-a technique that he created formodeling XP projects using pieces of the UML and Rational's UnifiedProcess (RUP). Ambler clearly explains AM, and shows readers how toincorporate AM, UML, and RUP into their development projects withthe help of numerous case studies integrated throughout the book. AM was created by the author for modeling XP projects-anelement lacking in the original XP design The XP community and its creator have embraced AM, which shouldgive this book strong market acceptance Companion Web site at www.agilemodeling.com features updates,links to XP and AM resources, and ongoing case studies about agilemodeling.

Awareness of design smells - indicators of common design problems - helps developers or software engineers understand mistakes made while designing, what design principles were overlooked or misapplied, and what principles need to be applied properly to address those smells through refactoring. Developers and software engineers may "know" principles and patterns, but are not aware of the "smells" that exist in their design because of wrong or mis-application of principles or patterns. These smells tend to contribute heavily to technical debt - further time owed to fix projects thought to be complete - and need to be addressed via proper refactoring. Refactoring for Software Design Smells presents 25 structural design smells, their role in identifying design issues, and potential refactoring solutions. Organized across common areas of software design, each smell is presented with diagrams and examples illustrating the poor design practices and the problems that result, creating a catalog of nuggets of readily usable information that developers or engineers can apply in their projects. The authors distill their research and experience as consultants and trainers, providing insights that have been used to improve refactoring and reduce the time and costs of managing software projects. Along the way they recount anecdotes from actual projects on which the relevant smell helped address a design issue. Contains a comprehensive catalog of 25 structural design smells (organized around four fundamental design principles) that contribute to technical debt in software projects Presents a unique naming scheme for smells that helps understand the cause of a smell as well as points toward its potential refactoring Includes illustrative examples that showcase the poor design practices underlying a smell and the problems that result Covers pragmatic techniques for refactoring design smells to manage technical debt and to create and maintain high-quality software in practice Presents insightful anecdotes and case studies drawn from the trenches of real-world projects

An introductory, yet comprehensive, database textbook intended for use in undergraduate and graduate information systems database courses. This text also provides practical content to current and aspiring information systems, business data analysis, and decision support industry professionals. Database Systems: Introduction to Databases and Data Warehouses covers both analytical and operations database as knowledge of both is integral to being successful in today's business environment. It also provides a solid theoretical foundation and hands-on practice using an integrated web-based data-modeling suite.

Test-Driven Database Development
Evolutionary Database Design (paperback)
Advanced Information Systems Engineering
Building Security in
A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working
Evolutionary Patterns to Transform Your Monolith
Recipes for Continuous Database Integration

For those considering Extreme Programming, this book provides no-nonsense advice on agile planning, development, delivery, and management taken from the authors' many years of experience. While plenty of books address the what and why of agile development, very few offer the information users can apply directly.

The practice of enterprise application development has benefited from the emergence of many new enabling technologies. Multi-tiered object-oriented platforms, such as Java and .NET, have become commonplace. These new tools and technologies are capable of building powerful applications, but they are not easily implemented. Common failures in enterprise applications often occur because their developers do not understand the architectural lessons that experienced object developers have learned. Patterns of Enterprise Application Architecture is written in direct response to the stiff challenges that face enterprise application developers. The author, noted object-oriented designer Martin Fowler, noticed that despite changes in technology--from Smalltalk to CORBA to Java to .NET--the same basic design ideas can be adapted and applied to solve common problems. With the help of an expert group of contributors, Martin distills over forty recurring solutions into patterns. The result is an indispensable handbook of solutions that are applicable to any enterprise application platform. This book is actually two books in one. The first section is a short tutorial on developing enterprise applications, which you can read from start to finish to understand the scope of the book's lessons. The next section, the bulk of the book, is a detailed reference to the patterns themselves. Each pattern provides usage and implementation information, as well as detailed code examples in Java or C#. The entire book is also richly

illustrated with UML diagrams to further explain the concepts. Armed with this book, you will have the knowledge necessary to make important architectural decisions about building an enterprise application and the proven patterns for use when building them. The topics covered include · Dividing an enterprise application into layers · The major approaches to organizing business logic · An in-depth treatment of mapping between objects and relational databases · Using Model-View-Controller to organize a Web presentation · Handling concurrency for data that spans multiple transactions · Designing distributed object interfaces

Users can dramatically improve the design, performance, and manageability of object-oriented code without altering its interfaces or behavior. "Refactoring" shows users exactly how to spot the best opportunities for refactoring and exactly how to do it, step by step.

Hundreds of organizations around the world have already benefited from DisciplinedAgile Delivery (DAD). Disciplined Agile (DA) is the only comprehensive tool kit available for guidance on building high-performance agile teams and optimizing your way of working(WoW). As a hybrid of all the leading agile and lean approaches, it provides hundreds of strategies to help you make better decisions within your agile teams, balancing self-organization with the realities and constraints of your unique enterprise context.The highlights of this handbook include:&•As the official source of knowledge on DAD, it includes greatly improved and enhanced strategies with a revised set of goal diagrams based upon learnings from applying DAD in the field.&•It is an essential handbook to help coaches and teams make better decisions in their daily work, providing a wealth of ideas for experimenting with agile and lean techniques while providing specific guidance and trade-offs for those &"it depends&" questions.&•It makes a perfect study guide for Disciplined Agile certification.Why &"fail fast&" (as our industry likes to recommend) when you can learn quickly on your journey to high performance? With this handbook, you can make better decisions based upon proven, context-based strategies, leading to earlier success and better outcomes.

Refactoring to Patterns

Fowler

Unlocking Agility

Improving the Design of Existing Code

Evolve the Monolith to Microservices with Java and Node

Managing Technical Debt

Refactoring Test Code

The only complete, proven, start-to-finish blueprint for successful 'just-in-time' agile database development! * *Knowledge virtually every agile shop needs, because nearly all of them must build and run databases *New agile approaches to ensuring that databases are consistent and stable in fast-changing environments, and test-driving designs to identify problems upfront, when they're cheaper to fix *Based on author Max Guernsey III's pioneering NetObjectives course in database agility. Design and build truly agile databases that can be changed frequently, safely, and painlessly, no matter how much existing data they must manage! With this book, you'll finally get past old-fashioned 'batch-and-queue' database development, and construct a truly agile database development environment that works! Pioneering agile database expert Max Guernsey III combines a complete foundation of theoretical knowledge with concrete examples and real solutions to the impediments that have prevented database developers from going agile. Guernsey especially shows how to adapt agile principles to handle massive amounts of existing data that makes database change more difficult. Test-Driven Database Development is based on the training curricula for the author's pioneering NetObjectives course, Database Agility Online Training, which has helped hundreds of database professionals master critical technical skills for designing databases that can be changed frequently, safely, and painlessly. Reflecting his immense experience with agile database development, Guernsey helps you make sure all databases and data remain consistent in agile environments; ensure stability no matter how fast databases change; and test-drive designs to find and fix errors before they're 'baked into' the system. This book will be an invaluable resource for virtually every database analyst and DBA in agile organizations; for many database team, project, and group managers; and for even more agile development team members in organizations that rely on large and complex databases.

Using Agile methods, you can bring far greater innovation, value, and quality to any data warehousing (DW), business intelligence (BI), or analytics project. However, conventional Agile methods must be carefully adapted to address the unique characteristics of DW/BI projects. In Agile Analytics, Agile pioneer Ken Collier shows how to do just that. Collier introduces platform-agnostic Agile solutions for integrating infrastructures consisting of diverse operational, legacy, and specialty systems that mix commercial and custom code. Using working examples, he shows how to manage analytics development teams with widely diverse skill sets and how to support enormous and fast-growing data volumes. Collier's techniques offer optimal value whether your projects involve "back-end" data management, "front-end" business analysis, or both. Part I focuses on Agile project management techniques and delivery team coordination, introducing core practices that shape the way your Agile DW/BI project community can collaborate toward success Part II presents technical methods for enabling continuous delivery of business value at production-quality levels, including evolving superior designs; test-driven DW development; version control; and project automation Collier brings together proven solutions you can apply right now--whether you're an IT decision-maker, data warehouse professional, database administrator, business intelligence specialist, or database developer. With his help, you can mitigate project risk, improve business alignment, achieve better results--and have fun along the way.

Describes how to put software security into practice, covering such topics as risk management frameworks, architectural risk analysis, security testing, and penetration testing.

How do you detangle a monolithic system and migrate it to a microservice architecture? How do you do it while maintaining business-as-usual? As a companion to Sam Newman's extremely popular Building Microservices, this new book details a proven method for transitioning an existing monolithic system to a microservice architecture. With many illustrative examples, insightful

migration patterns, and a bevy of practical advice to transition your monolith enterprise into a microservice operation, this practical guide covers multiple scenarios and strategies for a successful migration, from initial planning all the way through application and database decomposition. You'll learn several tried and tested patterns and techniques that you can use as you migrate your existing architecture. Ideal for organizations looking to transition to microservices, rather than rebuild Helps companies determine whether to migrate, when to migrate, and where to begin Addresses communication, integration, and the migration of legacy systems Discusses multiple migration patterns and where they apply Provides database migration examples, along with synchronization strategies Explores application decomposition, including several architectural refactoring patterns Delves into details of database decomposition, including the impact of breaking referential and transactional integrity, new failure modes, and more

Refactoring Databases

The Object Primer

Refactoring

A Value-driven Approach to Business Intelligence and Data Warehousing

The Elements of Java(TM) Style

Evolutionary Database Design

"This book takes the somewhat daunting process of database design and breaks it into completely manageable and understandable components. Mike's approach whilst simple is completely professional, and I can recommend this book to any novice database designer." --Sandra Barker, Lecturer, University of South Australia, Australia "Databases are a critical infrastructure technology for information systems and today's business. Mike Hernandez has written a literate explanation of database technology--a topic that is intricate and often obscure. If you design databases yourself, this book will educate you about pitfalls and show you what to do. If you purchase products that use a database, the book explains the technology so that you can understand what the vendor is doing and assess their products better." --Michael Blaha, consultant and trainer, author of A Manager's Guide to Database Technology "If you told me that Mike Hernandez could improve on the first edition of Database Design for Mere Mortals I wouldn't have believed you, but he did! The second edition is packed with more real-world examples, detailed explanations, and even includes database-design tools on the CD-ROM! This is a must-read for anyone who is even remotely interested in relational database design, from the individual who is called upon occasionally to create a useful tool at work, to the seasoned professional who wants to brush up on the fundamentals. Simply put, if you want to do it right, read this book!" --Matt Greer, Process Control Development, The Dow Chemical Company "Mike's approach to database design is totally common-sense based, yet he's adhered to all the rules of good relational database design. I use Mike's books in my starter database-design class, and I recommend his books to anyone who's interested in learning how to design databases or how to write SQL queries." --Michelle Poolet, President, MVDS, Inc. "Slapping together sophisticated applications with poorly designed data will hurt you just as much now as when Mike wrote his first edition, perhaps even more. Whether you're just getting started developing with data or are a seasoned pro; whether you've read Mike's previous book or this is your first; whether you're happier letting someone else design your data or you love doing it yourself--this is the book for you. Mike's ability to explain these concepts in a way that's not only clear, but fun, continues to amaze me." --From the Foreword by Ken Getz, MCW Technologies, coauthor ASP.NET Developer's JumpStart "The first edition of Mike Hernandez's book Database Design for Mere Mortals was one of the few books that survived the cut when I moved my office to smaller quarters. The second edition expands and improves on the original in so many ways. It is not only a good, clear read, but contains a remarkable quantity of clear, concise thinking on a very complex subject. It's a must for anyone interested in the subject of database design." --Malcolm C. Rubel, Performance Dynamics Associates "Mike's excellent guide to relational database design deserves a second edition. His book is an essential tool for fledgling Microsoft Access and other desktop database developers, as well as for client/server pros. I recommend it highly to all my readers." --Roger Jennings, author of Special Edition Using Access 2002 "There are no silver bullets! Database technology has advanced dramatically, the newest crop of database servers perform operations faster than anyone could have imagined six years ago, but none of these technological advances will help fix a bad database design, or capture data that you forgot to include! Database Design for Mere Mortals(TM), Second Edition, helps you design your database right in the first place!" --Matt Nunn, Product Manager, SQL Server, Microsoft Corporation "When my brother started his professional career as a developer, I gave him Mike's book to help him understand database concepts and make real-world application of database technology. When I need a refresher on the finer points of database design, this is the book I pick up. I do not think that there is a better testimony to the value of a book than that it gets used. For this reason I have wholeheartedly recommended to my peers and students that they utilize this book in their day-to-day development tasks." --Chris Kunicki, Senior Consultant, OfficeZealot.com "Mike has always had an incredible knack for taking the most complex topics, breaking them down, and explaining them so that anyone can 'get it.' He has honed and polished his first very, very good edition and made it even better. If you're just starting out building database applications, this book is a must-read cover to cover. Expert designers will find Mike's approach fresh and enlightening and a source of great material for training others." --John Viescas, President, Viescas Consulting, Inc., author of Running Microsoft Access 2000 and coauthor of SQL Queries for Mere Mortals "Whether you need to learn about relational database design in general, design a relational database, understand relational database terminology, or learn best practices for implementing a relational database, Database Design for Mere Mortals(TM), Second Edition, is an indispensable book

that you'll refer to often. With his many years of real-world experience designing relational databases, Michael shows you how to analyze and improve existing databases, implement keys, define table relationships and business rules, and create data views, resulting in data integrity, uniform access to data, and reduced data-entry errors." --Paul Cornell, Site Editor, MSDN Office Developer Center Sound database design can save hours of development time and ensure functionality and reliability. Database Design for Mere Mortals(TM), Second Edition, is a straightforward, platform-independent tutorial on the basic principles of relational database design. It provides a commonsense design methodology for developing databases that work. Database design expert Michael J. Hernandez has expanded his best-selling first edition, maintaining its hands-on approach and accessibility while updating its coverage and including even more examples and illustrations. This edition features a CD-ROM that includes diagrams of sample databases, as well as design guidelines, documentation forms, and examples of the database design process. This book will give you the knowledge and tools you need to create efficient and effective relational databases.

Refactoring DatabasesEvolutionary Database Design (paperback)Pearson Education

This text aims to help all members of the development team make the correct nuts-and-bolts architecture decisions that ensure project success.

Automated testing is a cornerstone of agile development. An effective testing strategy will deliver new functionality more aggressively, accelerate user feedback, and improve quality. However, for many developers, creating effective automated tests is a unique and unfamiliar challenge. xUnit Test Patterns is the definitive guide to writing automated tests using xUnit, the most popular unit testing framework in use today. Agile coach and test automation expert Gerard Meszaros describes 68 proven patterns for making tests easier to write, understand, and maintain. He then shows you how to make them more robust and repeatable--and far more cost-effective. Loaded with information, this book feels like three books in one. The first part is a detailed tutorial on test automation that covers everything from test strategy to in-depth test coding. The second part, a catalog of 18 frequently encountered "test smells," provides trouble-shooting guidelines to help you determine the root cause of problems and the most applicable patterns. The third part contains detailed descriptions of each pattern, including refactoring instructions illustrated by extensive code samples in multiple programming languages.

A Brief Guide to the Emerging World of Polyglot Persistence

Continuous Integration

Domain-Specific Languages

Database Systems

Effective Strategies for the Agile Software Developer

A Self-Improvement Process for Software Engineers

Pattern Enterpr Applica Arch

Scott Ambler, author of Building Object Applications that Work, Process Patterns, and More Process Patterns, has revised his acclaimed first book, The Object Primer. Long prized in its original edition by both students and professionals as the best introduction to object-oriented technology, now this book is completely up-to-date with new material in every chapter. There are also new chapters on good OO programming techniques and OO software testing. All modeling notation has been rewritten in UML notation. Review questions at the end of each chapter allow readers to test their newly acquired knowledge. In addition, the author takes time to reflect on the lessons learned over the past few years by discussing the proven benefits and drawbacks of the technology. This is the perfect book for any software development professional or student seeking an introduction to the concepts and terminology of object technology.

This is the eBook version of the printed book. The past few years have seen the rise of agile or evolutionary methods in software development. These methods embrace change in requirements even late in the project. The ability to change software is because of certain practices that are followed within teams, such as Test Driven Development, Pair Programming, and Continuous Integration. Continuous Integration provides a way for software teams to integrate their work more than once a day, and promotes confidence in the software that is being developed by the team. It is thought that this practice is difficult to apply when continuously integrating the database with application code; hence, Evolutionary Database Development is considered a mismatch with agile methods. Pramod Sadalage shows that this is not necessarily true. Continuous Integration changed the way software is written. Why not extend and make the database part of the same Continuous Integration cycle so that you can see integrated results of your application as well as your database? Delivered in PDF format for quick and easy access, Recipes for Continuous Database Integration shows how the database can be brought under the preview of Continuous Integration, allowing all teams to integrate not only their application code, but also their database. This Short Cut presents a recipe for each task that needs to be done. Each recipe starts with a statement of a problem, followed by an explanation and solution. It provides concrete ways and examples to implement ideas in Refactoring Databases: Evolutionary Database Design by Scott W Ambler and Pramod Sadalage. Table of Contents What This Short Cut Covers Introduction Recipe 1 Continuously Integrating? Recipe 2 Extracting Your Database in Scripts Recipe 3 Using Version Control for Your Database Recipe 4 Automating Database or Schema Creation Recipe 5 Creating Objects in Your Database Recipe 6 Removing Database Objects Recipe 7 Removing Your Database Recipe 8 Using the Build Property Files Recipe 9 Re-Creating Your Application Database for Any Build Recipe 10 Making It Easy for New Developers to Join the Team Recipe 11 Integrating on Every Check-In Recipe 12 Naming Upgrade Scripts Recipe 13 Automating Database Change Script Creation Recipe 14 Implementing Database Version Checking Recipe 15 Sending Upgrades to Customers Sample Code Further Reading About the Author What's in the Companion Book Related Publication

There are no easy decisions in software architecture. Instead, there are many hard parts--difficult

problems or issues with no best practices--that force you to choose among various compromises. With this book, you'll learn how to think critically about the trade-offs involved with distributed architectures. Architecture veterans and practicing consultants Neal Ford, Mark Richards, Pramod Sadalage, and Zhamak Dehghani discuss strategies for choosing an appropriate architecture. By interweaving a story about a fictional group of technology professionals--the Sysops Squad--they examine everything from how to determine service granularity, manage workflows and orchestration, manage and decouple contracts, and manage distributed transactions to how to optimize operational characteristics, such as scalability, elasticity, and performance. By focusing on commonly asked questions, this book provides techniques to help you discover and weigh the trade-offs as you confront the issues you face as an architect. Analyze trade-offs and effectively document your decisions Make better decisions regarding service granularity Understand the complexities of breaking apart monolithic applications Manage and decouple contracts between services Handle data in a highly distributed architecture Learn patterns to manage workflow and transactions when breaking apart applications

27th International Conference, DEXA 2016, Porto, Portugal, September 5-8, 2016, Proceedings, Part I