# Continuous Delivery Reliable Software Releases Through Build Test And Deployment Automation Addison Wesley Signature

Over 70 recipes to effectively apply DevOps best practices and implement Agile, Git, CI-CD & Test automation using Azure DevOps Server (TFS) 2019 Key FeaturesLearn improving code quality using pull requests, branch policies, githooks and git branching designAccelerate the deployment of high quality software by automating build and releases using CI-CD Pipelines.Learn tried and tested techniques to automate database deployments, App Service & Function Deployments in Azure.Book Description Azure DevOps Server, previously known as Team Foundation Server (TFS), is a comprehensive on-premise DevOps toolset with a rich ecosystem of open source plugins. This book is your one stop guide to learn how to effectively use all of these Azure DevOps services to go from zero to DevOps. You will start by building high-quality scalable software targeting .NET, .NET core or Node.js applications. You will learn techniques that will help you to set up end-to-end traceability of your code changes from design through to release. Whether you are deploying software on-premise or in the cloud in App Service, Functions, or Azure VMs, this book will help you learn release management techniques to reduce release failures. Next, you will be able to secure application configuration by using Azure KeyVault. You will also learn how to create and release extensions to the Azure DevOps marketplace and reach million developer ecosystem for feedback. The working extension samples will allow you to iterate changes in your extensions easily and release updates to the marketplace quickly. By the end of this book, techniques provided in the book will help you break down the invisible silos between your software development teams. This will transform you from being a good software development team to an elite modern cross functional software development team. What you will learnSet up a team project for an Agile delivery team, importing requirements from ExcelPlan,track, and monitor progress using self updating boards, Sprint and Kanban boardsUnlock the features of Git by using branch policies, Git pull requests, forks, and Git hooksBuild and release .NET core, SQL and Node.js applications using Azure PipelineAutomate testing by integrating Microsoft and open source testing frameworksExtend Azure DevOps Server to a million developer ecosystemWho this book is for This book is for anyone looking to succeed with DevOps. The techniques in this book apply to all roles of the software development lifecycle including developers, testers, architects, configuration analysts, site reliability engineers and release managers. If you are a new user you'll learn how to get started; if you are an experienced user you'll learn how to launch your project into a modern and mature DevOps enabled software development team.

As the study of travel writing has grown in recent years, scholars have largely ignored the literature of modernist writers. Modernist Travel Writing: Intellectuals Abroad, by David Farley, addresses this gap by examining the ways in which a number of writers employed the techniques and stylistic innovations of modernism in their travel narratives to variously engage the political, social, and cultural milieu of the years between the world wars. Modernist Travel Writing argues that the travel book is a crucial genre for understanding the development of modernism in the years between the wars, despite the established view that travel writing during the interwar period was largely an escapist genre—one in which writers hearkened back to the realism of nineteenth-century literature in order to avoid interwar anxiety. Farley analyzes works that exist on the margins of modernism, generically and geographically, works that have yet to receive the critical attention they deserve, partly due to their classification as travel narratives and partly because of their complex modernist styles. The book begins by examining the ways that travel and the emergent travel regulations in the wake of the First World War helped shape Ezra Pound's Cantos. From there, it goes on to examine E. E. Cummings's frustrated attempts to navigate the "unworld" of Soviet Russia in his book Eimi,Wyndham Lewis's satiric journey through colonial Morocco in Filibusters in Barbary,and Rebecca West's urgent efforts to make sense of the fractious Balkan states in Black Lamb and Grey Falcon. These modernist writers traveled to countries that experienced most directly the tumult of revolution, the effects of empire, and the upheaval of war during the years between World War I and World War II. Farley's study focuses on the question of what constitutes "evidence" for Pound, Lewis, Cummings, and West as they establish their authority as eyewitnesses, translate what they see for an audience back home, and attempt to make sense of a transformed and transforming modern world. Modernist Travel Writing makes an original contribution to the study of literary modernism while taking a distinctive look at a unique subset within the growing field of travel writing studies. David Farley's work will be of interest to students and teachers in both of these fields as well as to early-twentieth-century literary historians and general enthusiasts of modernist studies.

Continuous Testing for DevOps Professionals is the definitive guide for DevOps teams and covers the best practices required to excel at Continuous Testing (CT) at each step of the DevOps pipeline. It was developed in collaboration with top industry experts from across the DevOps domain from leading companies such as CloudBees, Tricentis, Testim.io, Test.ai, Perfecto, and many more. The book is aimed at all DevOps practitioners, including software developers, testers, operations managers, and IT/business executives. It consists of 4 sections: 1. Fundamentals of Continuous Testing 2. Continuous Testing for Web Apps 3. Continuous Testing for Mobile Apps 4. Advancing Continuous Testing All profits from Continuous Testing for DevOps Professionals will be donated to code.org, which is a nonprofit dedicated to expanding access to computer science in schools and increasing participation by women and underrepresented minorities.

Learn to design, implement, measure, and improve DevOps programs that are tailored to your organization. This concise guide assists leaders who are accountable for the rapid development of high-quality software applications. In DevOps for Digital Leaders, deep collective experience on both sides of the dev–ops divide informs the global thought leadership and penetrating insights of the authors, all three of whom are cross-portfolio DevOps leaders at CA Technologies. Aruna Ravichandran, Kieran Taylor, and Peter Waterhouse analyze the organizational benefits, costs, freedoms, and constraints of DevOps. They chart the coordinated strategy of organizational change, metrics, lean thinking, and investment that an enterprise must undertake to realize the full potential of DevOps and reach the sweet spot where accelerating code deployments drive increasing customer satisfaction, revenue, and profitability. Digital leaders are charged to bridge the dev–ops disconnect if their organizations are to survive and flourish in a business world increasingly differentiated by the degree to which dynamic application software development harmonizes with operational resilience and reliability. This short book applies the DevOps perspective to the competitive challenge, faced by every high-performance IT organization today, of integrating and automating open source, cloud, and enterprise tools, processes, and techniques across the software development life cycle from requirements to release. What You Will Learn: Remove dependencies and constraints so that parallel practices can accelerate the development of defect-free software Automate continuous delivery across the software life cycle to eliminate release bottlenecks, manual labor waste, and technical debt accumulation Generate virtualized production-style testing of applications through real-time behavioral analytics Adopt agile practices so operations teams can support developer productivity with automated feedback, streamline infrastructure monitoring, spot and resolve operations issues before they impact production, and improve customer experience Identify the DevOps metrics appropriate to your organization and integrate DevOps with your existing best practices and investment Who This Book Is For: IT leaders in large companies and government agencies who have any level of responsibility for the rapid development of high-quality software applications. The secondary readership is members of development and operations teams, security professionals, and service managers.

How HP Transformed LaserJet FutureSmart Firmware

Reignite Business with a Modern DevOps-Enabled Software Factory

Lessons Learned from Programming Over Time

Azure DevOps Server 2019 Cookbook

Release It!

Modern Software Engineering

Doing What Works to Build Better Software Faster

For thousands of companies, the Amazon Web Services (AWS) cloud is today's software development environment of choice. Now there's a complete guide to using DevOps and continuous delivery techniques on AWS -- so you can reliably deliver new features to users and customers at the click of a button. First, leading software development consultant Paul Duvall concisely reviews DevOps' principles, culture, and goals. Next, using a realistic reference implementation, he offers detailed hands-on guidance on applying automation throughout the entire AWS cloud software delivery process. Finally, he presents up-to-date case studies of companies applying DevOps throughout their own modern development environments: from Netflix to AMC Health to the U.S. government. Using principles, patterns, and examples you'll find here, you can make the most of DevOps and continuous delivery with today's most widely-used cloud platform. What's more, you'll master skills you can use as AWS evolves -- or with any other cloud platform you choose.

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.

Continuous delivery adds enormous value to the business and the entire software delivery lifecycle, but adopting this practice means mastering new skills typically outside of a developer's comfort zone. In this practical book, Daniel Bryant and Abraham Marín-Pérez provide guidance to help experienced Java developers master skills such as architectural design, automated quality assurance, and application packaging and deployment on a variety of platforms. Not only will you learn how to create a comprehensive build pipeline for continually delivering effective software, but you'll also explore how Java application architecture and deployment platforms have affected the way we rapidly and safely deliver new software to production environments. Get advice for beginning or completing your migration to continuous delivery Design architecture to enable the continuous delivery of Java applications Build application artifacts including fat JARs, virtual machine images, and operating system container (Docker) images Use continuous integration tooling like Jenkins, PMD, and find-sec-bugs to automate code quality checks Create a comprehensive build pipeline and design software to separate the deploy and release processes Explore why functional and system quality attribute testing is vital from development to delivery Learn how to effectively build and test applications locally and observe your system while it runs in production

Today, even the largest development organizations are turning to agile methodologies, seeking major productivity and quality improvements. However, large-scale agile development is difficult, and publicly available case studies have been scarce. Now, three agile pioneers at Hewlett-Packard present a candid, start-to-finish insider's look at how they've succeeded with agile in one of the company's most mission-critical software environments: firmware for HP LaserJet printers. This book tells the story of an extraordinary experiment and journey. Could agile principles be applied to re-architect an enormous legacy code base? Could agile enable both timely delivery and ongoing innovation? Could it really be applied to 400+ developers distributed across four states, three continents, and four business units? Could it go beyond delivering incremental gains, to meet the stretch goal of 10x developer productivity improvements? It could, and it did—but getting there was not easy. Writing for both managers and technologists, the authors candidly discuss both their successes and failures, presenting actionable lessons for other development organizations, as well as approaches that have proven themselves repeatedly in HP's challenging environment. They not only illuminate the potential benefits of agile in large-scale development, they also systematically show how these benefits can actually be achieved. Coverage includes: • Tightly linking agile methods and enterprise architecture with business objectives • Focusing agile practices on your worst development pain points to get the most bang for your buck • Abandoning classic agile methods that don't work at the largest scale • Employing agile methods to establish a new architecture • Using metrics as a "conversation starter" around agile process improvements • Leveraging continuous integration and quality systems to reduce costs, accelerate schedules, and automate the delivery pipeline • Taming the planning beast with "light-touch" agile planning and lightweight long-range forecasting • Implementing effective project management and ensuring accountability in large agile projects • Managing tradeoffs associated with key decisions about organizational structure • Overcoming U.S./India cultural differences that can complicate offshore development • Selecting tools to support quantum leaps in productivity in your organization • Using change management disciplines to support greater enterprise agility

Site Reliability Engineering

A Practical Guide from Industry Experts

Effective DevOps

Essential Tools and Best Practices for Deploying Code to Production

Lean Enterprise

Pulling Strings with Puppet

Unit Testing Principles, Practices, and Patterns

Continuous Delivery with Docker and Jenkins

Continuous DeliveryReliable Software Releases Through Build, Test, and Deployment AutomationAddison-Wesley Professional

What's the best way to get code from your laptop into a production environment? With this highly actionable guide, architects, developers, engineers, and others in the IT space will learn everything from local development builds to continuous integration and continuous delivery, and from mobile, IOT, and container-based deployments to infrastructure automation and testing. Author Sam Newman (Building Microservices) provides real-world examples to depict the end-to-end journey through today's state-of-the art build and deployment process. Much has changed in just a few years: the cloud has grown, mobile development has proliferated, DevOps is finding wider acceptance, and the IoT is very much in play. You'll start with the big picture and then dive into individual topics that govern modern software production. This book explores: Automated builds, version control, repeatable builds, and artifact storage What continuous integration includes--and what it doesn't Version control basics, cherry picking, release branching, gitflow, and pull requests Core tenets of continuous delivery and the technology behind it Application deployment topics, including artifact types and deployment platforms Why you should treat testing and monitoring/alerting as an inseparable whole How modules, static linking, and microservices enable you to deploy software quickly

The step-by-step guide to going live with new software releases faster - reducing risk and delivering more value sooner! * *Fast, simple, repeatable techniques for deploying working code to production in hours or days, not months! *Crafting custom processes that get developers from idea to value faster than ever. *Best practices for everything from source code control to dependency management and in-production tracing. *Common obstacles to rapid release - and pragmatic solutions. In too many organizations, build, testing, and deployment processes can take six months or more. That's simply far too long for today's businesses. But it doesn't have to be that way. It's possible to deploy working code to production in hours or days after development work is complete - and Go Live presents comprehensive processes and techniques for doing so. Written by two of the world's most experienced software project leaders, this book demonstrates how to dramatically increase speed while reducing risk and improving code quality at the same time. The authors cover all facets of build, testing, and deployment, including: configuration management, source code control, release planning, auditing, compliance, integration, build automation, and more. They introduce a wide range of advanced techniques, including inproduction monitoring and tracing, dependency management, and the effective use of virtualization. For each area, they explain the issues, show how to mitigate the risks, and present best practices. Throughout, Go Live focuses on powerful opportunities for individual improvement, clearly and simply explaining skills and techniques so they can be used every day on real projects. With this book's help, any development organization can move from idea to release faster -- and deliver far more value, far more rapidly.

Radically improve your testing practice and software quality with new testing styles, good patterns, and reliable automation. Key Features A practical and results-driven approach to unit testing Refine your existing unit tests by implementing modern best practices Learn the four pillars of a good unit test Safely automate your testing process to save time and money Spot which tests need refactoring, and which need to be deleted entirely Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About The Book Great testing practices maximize your project quality and delivery speed by identifying bad code early in the development process. Wrong tests will break your code, multiply bugs, and increase time and costs. You owe it to yourself—and your projects—to learn how to do excellent unit testing. Unit Testing Principles, Patterns and Practices teaches you to design and write tests that target key areas of your code including the domain model. In this clearly written guide, you learn to develop professional-quality tests and test suites and integrate testing throughout the application life cycle. As you adopt a testing mindset, you'll be amazed at how better tests cause you to write better code. What You Will Learn Universal guidelines to assess any unit test Testing to identify and avoid anti-patterns Refactoring tests along with the production code Using integration tests to verify the whole system This Book Is Written For For readers who know the basics of unit testing. Examples are written in C# and can easily be applied to any language. About the Author Vladimir Khorikov is an author, blogger, and Microsoft MVP. He has mentored numerous teams on the ins and outs of unit testing. Table of Contents: PART 1 THE BIGGER PICTURE 1 ¦ The goal of unit testing 2 ¦ What is a unit test? 3 ¦ The anatomy of a unit test PART 2 MAKING YOUR TESTS WORK FOR YOU 4 ¦ The four pillars of a good unit test 5 ¦ Mocks and test fragility 6 ¦ Styles of unit testing 7 ¦ Refactoring toward valuable unit tests PART 3 INTEGRATION TESTING 8 ¦ Why integration testing? 9 ¦ Mocking best practices 10 ¦ Testing the database PART 4 UNIT TESTING ANTI-PATTERNS 11 ¦ Unit testing anti-patterns

Pipeline as Code

Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services

A Practical Approach to Large-Scale Agile Development

Improving Software Quality and Reducing Risk

Software Configuration Management Patterns

Faster, Safer Software Delivery

The Art of Readable Code

How Google Runs Production Systems

**The overwhelming majority of a software system's lifespan is spent in use, not in design or implementation. So, why does conventional wisdom insist that software engineers focus primarily on the design and development of large-scale computing systems? In this collection of essays and articles, key members of Google's Site Reliability Team explain how and why their commitment to the entire lifecycle has enabled the company to successfully build, deploy, monitor, and maintain some of the largest software systems in the world. You'll learn the principles and practices that enable Google engineers to make systems more scalable, reliable, and efficient—lessons directly applicable to your organization. This book is divided into four sections: Introduction—Learn what site reliability engineering is and why it differs from conventional IT industry practices Principles—Examine the patterns, behaviors, and areas of concern that influence the work of a site reliability engineer (SRE) Practices—Understand the theory and practice of an SRE's day-to-day work: building and operating large distributed computing systems Management—Explore Google's best practices for training, communication, and meetings that your organization can use**

**Follow this step-by-step guide for creating a continuous delivery pipeline using all of the new features in Jenkins 2.0 such as Pipeline as a Code, multi-branch pipeline, and more. You will learn three crucial elements for achieving a faster software delivery pipeline: a fungible build/test environment, manageable and reproducible pipelines, and a scalable build/test infrastructure. Pro Continuous Delivery demonstrates how to create a highly available, active/passive Jenkins server using some niche technologies. What You'll Learn Create a highly available, active/passive Jenkins server using CoreOS and Docker, and using Pacemaker and Corosync Use a Jenkins multi-branch pipeline to automatically perform continuous integration whenever there is a new branch in your source control system Describe your continuous delivery pipeline with Jenkinsfile Host Jenkins server on a cloud solution Run Jenkins inside a container using Docker Discover how the distributed nature of Git and the "merge before build" feature of Jenkins can be used to implement gated check-in Implement a scalable build farm using Docker and Kubernetes Who This Book Is For You have experience implementing continuous integration and continuous delivery using Jenkins freestyle**

Jobs and wish to use the new Pipeline as a Code feature introduced in Jenkins 2.0 Your source code is on a Git-like version control system (Git, GitHub, GitLab, etc.) and you wish to leverage the advantages of a multi-branch pipeline in Jenkins Your infrastructure is on a Unix-like platform and you wish to create a scalable, distributed build/test farm using Docker or Kubernetes You are in need of a highly available system for your Jenkins Server using open source tools and technologies

Getting started with the processes and the tools to continuously deliver high-quality software About This Book Incorporate popular development practices to prevent messy code Automate your build, integration, release, and deployment processes with Jenkins, Git, and Gulp?and learn how continuous integration (CI) can save you time and money Gain an end-to-end overview of Continuous Integration using different languages (JavaScript and C#) and tools (Gulp and Jenkins) Who This Book Is For This book is for developers who want to understand and implement Continuous Integration and Delivery in their daily work. A basic knowledge of at least JavaScript and HTML/CSS is required. Knowing C# and SQL will come in handy. Most programmers who have programmed in a (compiled) C-like language will be able to follow along. What You Will Learn Get to know all the aspects of Continuous Integration, Deployment, and Delivery Find out how Git can be used in a CI environment Set up browser tests using Karma and Selenium and unit tests using Jasmine Use Node.js, npm, and Gulp to automate tasks such as linting, testing, and minification Explore different Jenkins jobs to integrate with Node.js and C# projects Perform Continuous Delivery and Deployment using Jenkins Test and deliver a web API In Detail The challenge faced by many teams while implementing Continuous Deployment is that it requires the use of many tools and processes that all work together. Learning and implementing all these tools (correctly) takes a lot of time and effort, leading people to wonder whether it's really worth it. This book sets up a project to show you the different steps, processes, and tools in Continuous Deployment and the actual problems they solve. We start by introducing Continuous Integration (CI), deployment, and delivery as well as providing an overview of the tools used in CI. You'll then create a web app and see how Git can be used in a CI environment. Moving on, you'll explore unit testing using Jasmine and browser testing using Karma and Selenium for your app. You'll also find out how to automate tasks using Gulp and Jenkins. Next, you'll get acquainted with database integration for different platforms, such as MongoDB and PostgreSQL. Finally, you'll set up different Jenkins jobs to integrate with Node.js and C# projects, and Jenkins pipelines to make branching easier. By the end of the book, you'll have implemented Continuous Delivery and deployment from scratch. Style and approach This practical book takes a step-by-step approach to explaining all the concepts of Continuous Integration and delivery, and how it can help you deliver a high-quality product.

This book provides essential insights on the adoption of modern software engineering practices at large companies producing software-intensive systems, where hundreds or even thousands of engineers collaborate to deliver on new systems and new versions of already deployed ones. It is based on the findings collected and lessons learned at the Software Center (SC), a unique collaboration between research and industry, with Chalmers University of Technology, Gothenburg University and Malmö University as academic partners and Ericsson, AB Volvo, Volvo Car Corporation, Saab Electronic Defense Systems, Grundfos, Axis Communications, Jeppesen (Boeing) and Sony Mobile as industrial partners. The 17 chapters present the “Stairway to Heaven” model, which represents the typical evolution path companies move through as they develop and mature their software engineering capabilities. The chapters describe theoretical frameworks, conceptual models and, most importantly, the industrial experiences gained by the partner companies in applying novel software engineering techniques. The book’s structure consists of six parts. Part I describes the model in detail and presents an overview of lessons learned in the collaboration between industry and academia. Part II deals with the first step of the Stairway to Heaven, in which R&D adopts agile work practices. Part III of the book combines the next two phases, i.e., continuous integration (CI) and continuous delivery (CD), as they are closely intertwined. Part IV is concerned with the highest level, referred to as “R&D as an innovation system,” while Part V addresses a topic that is separate from the Stairway to Heaven and yet critically important in large organizations: organizational performance metrics that capture data, and visualizations of the status of software assets, defects and teams. Lastly, Part VI presents the perspectives of two of the SC partner companies. The book is intended for practitioners and professionals in the software-intensive systems industry, providing concrete models, frameworks and case studies that show the specific challenges that the partner companies encountered, their approaches to overcoming them, and the results. Researchers will gain valuable insights on the problems faced by large software companies, and on how to effectively tackle them in the context of successful cooperation projects.

Software Engineering at Google

Unraveling Software Maintenance and Evolution

Simple and Practical Techniques for Writing Better Code

Continuous Integration, Delivery, and Deployment

DevOps in Amazon Web Services

Reliable and faster software releases with automating builds, tests, and deployment

Continuous Software Engineering

Configuration Management Made Easy

***Competent system administrators know their success hinges upon being able to perform often tedious tasks with rigor and punctuality. Such metrics are often achieved only by instituting a considerable degree of automation, something that has become even more crucial as IT environments continue to scale both in terms of size and complexity. One of the most powerful system administration tools to be released is Puppet, a solution capable of automating nearly every aspect of a system administrator's job, from user management, to software installation, to even configuring server services such as FTP and LDAP. Pulling Strings with Puppet: Configuration Management Made Easy is the first book to introduce the powerful Puppet system administration tool. Author James Turnbull will guide you through Puppet's key features, showing you how to install and configure the software, create automated Puppet tasks, known as recipes, and even create reporting solutions and extend Puppet further to your own needs. A bonus chapter is included covering the Facter library, which makes it a breeze to automate the retrieval of server configuration details such as IP and MAC addresses.***

***A web application involves many specialists, but it takes people in web ops to ensure that everything works together throughout an application's lifetime. It's the expertise you need when your start-up gets an unexpected spike in web traffic, or when a new feature causes your mature application to fail. In this collection of essays and interviews, web veterans such as Theo Schlossnagle, Baron Schwartz, and Alistair Croll offer insights into this evolving field. You'll learn stories from the trenches--from builders of some of the biggest sites on the Web--on what's necessary to help a site thrive. Learn the skills needed in web operations, and why they're gained through experience rather than schooling Understand why it's important to gather metrics from both your application and infrastructure Consider common approaches to database architectures and the pitfalls that come with increasing scale Learn how to handle the human side of outages and degradations Find out how one company avoided disaster after a huge traffic deluge Discover what went wrong after a problem occurs, and how to prevent it from happening again Contributors include: John Allspaw Heather Champ Michael Christian Richard Cook Alistair Croll Patrick Debois Eric Florenzano Paul Hammond Justin Huff Adam Jacob Jacob Loomis Matt Massie Brian Moon Anoop Nagwani Sean Power Eric Ries Theo Schlossnagle Baron Schwartz Andrew Shafer***

***Today, software engineers need to know not only how to program effectively but also how to develop proper engineering practices to make their codebase sustainable and healthy. This book emphasizes this difference between programming and software engineering. How can software engineers manage a living codebase that evolves and responds to changing requirements and demands over the length of its life? Based on their experience at Google, software engineers Titus Winters and Hyrum Wright, along with technical writer Tom Manshreck, present a candid and insightful look at how some of the world’s leading practitioners construct and maintain software. This book covers Google’s unique engineering culture, processes, and tools and how these aspects contribute to the effectiveness of an engineering organization. You’ll explore three fundamental principles that software organizations should keep in mind when designing, architecting, writing, and maintaining code: How time affects the sustainability of software and how to make your code resilient over time How scale affects the viability of software practices within an engineering organization What trade-offs a typical engineer needs to make when evaluating design and development decisions***

***Continuous Architecture provides a broad architectural perspective for continuous delivery, and describes a new architectural approach that supports and enables it. As the pace of innovation and software releases increases, IT departments are tasked to deliver value quickly and inexpensively to their business partners. With a focus on getting software into end-users hands faster, the ultimate goal of daily software updates is in sight to allow teams to ensure that they can release every change to the system simply and efficiently. This book presents an architectural approach to support modern application delivery methods and provide a broader architectural perspective, taking architectural concerns into account when deploying agile or continuous delivery approaches. The authors explain how to solve the challenges of implementing continuous delivery at the project and enterprise level, and the impact on IT processes including application testing, software deployment and software architecture. Covering the application of enterprise and software architecture concepts to the Agile and Continuous Delivery models Explains how to create an architecture that can evolve with applications Incorporates techniques including refactoring, architectural analysis, testing, and feedback-driven development Provides insight into incorporating modern software development when structuring teams and organizations***

***The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations***

***Effective Teamwork, Practical Integration***

***Continuous Architecture***

***Web Operations***

***What is DevOps?***

***Continuous Integration***

***Service Design Patterns***

***Infrastructure as Code***

Winner of the Shingo Publication Award Accelerate your organization to win in the marketplace. How can we apply technology to drive business value? For years, we've been told that the performance of software delivery teams doesn't matter?that it can't provide a competitive advantage to our companies. Through four years of groundbreaking research to include data collected from the State of DevOps reports conducted with Puppet, Dr. Nicole Forsgren, Jez Humble, and Gene Kim set out to find a way to measure software delivery performance?and what drives it?using rigorous statistical methods. This book presents both the findings and the science behind that research, making the information accessible for readers to apply in their own organizations. Readers will discover how to measure the performance of their teams, and what capabilities they should invest in to drive higher performance. This book is ideal for management at every level.

Winner of the 2011 Jolt Excellence Award! Getting software released to users is often a painful, risky, and time-consuming process. This groundbreaking new book sets out the principles and technical practices that enable rapid, incremental delivery of high quality, valuable new functionality to users. Through automation of the build, deployment, and testing process, and improved collaboration between developers, testers, and operations, delivery teams can get changes released in a matter of hours— sometimes even minutes–no matter what the size of a project or the complexity of its code base. Jez Humble and David Farley begin by presenting the foundations of a rapid, reliable, low-risk delivery process. Next, they introduce the “deployment pipeline,” an automated process for managing all changes, from check-in to release. Finally, they discuss the “ecosystem” needed to support continuous delivery, from infrastructure, data and configuration management to governance. The authors introduce state-of-the-art techniques, including automated infrastructure management and data migration, and the use of virtualization. For each, they review key issues, identify best practices, and demonstrate how to mitigate risks. Coverage includes • Automating all facets of building, integrating, testing, and deploying software • Implementing deployment pipelines at team and organizational levels • Improving collaboration between developers, testers, and operations • Developing features incrementally on large and distributed teams • Implementing an effective configuration management strategy • Automating acceptance testing, from analysis to implementation • Testing capacity and other non-functional requirements • Implementing continuous deployment and zero-downtime releases • Managing infrastructure, data, components and dependencies • Navigating risk management, compliance, and auditing Whether you’re a developer, systems administrator, tester, or manager, this book will help your organization move from idea to release faster than ever—so you can deliver value to your business rapidly and reliably.

Some companies think that adopting devops means bringing in specialists or a host of new tools. With this practical guide, you’ll learn why devops is a professional and cultural movement that calls for change from inside your organization. Authors Ryn Daniels and Jennifer Davis provide several approaches for improving collaboration within teams, creating affinity among teams, promoting efficient tool usage in your company, and scaling up what works throughout your organization’s inflection points. Devops stresses iterative efforts to break down information silos, monitor relationships, and repair misunderstandings that arise between and within teams in your organization. By applying the actionable strategies in this book, you can make sustainable changes in your environment regardless of your level within your organization. Explore the foundations of devops and learn the four pillars of effective devops Encourage collaboration to help individuals work together and build durable and long-lasting relationships Create affinity among teams while balancing differing goals or metrics Accelerate cultural direction by selecting tools and workflows that complement your organization Troubleshoot common problems and misunderstandings that can arise throughout the organizational lifecycle Learn from case studies from organizations and individuals to help inform your own devops journey

How well does your organization respond to changing market conditions, customer needs, and emerging technologies when building software-based products? This practical guide presents Lean and Agile principles and patterns to help you move fast at scale—and demonstrates why and how to apply these paradigms throughout your organization, rather than with just one department or team. Through case studies, you’ll learn how successful enterprises have rethought everything from governance and financial management to systems architecture and organizational culture in the pursuit of radically improved performance. Discover how Lean focuses on people and teamwork at every level, in contrast to traditional management practices Approach problem-solving experimentally by exploring solutions, testing assumptions, and getting feedback from real users Lead and manage large-scale programs in a way that empowers employees, increases the speed and quality of delivery, and lowers costs Learn how to implement ideas from the DevOps and Lean Startup movements even in complex, regulated environments

Domain-Specific Languages

The DevOps Handbook

Continuous Testing for DevOps Professionals

Releasing Software to Production at Any Time with AWS

Keeping the Data On Time

The Art of Capacity Planning

Design, User Experience, and Usability: Theory, Methodology, and Management

Hands-On Software Architecture with Golang

*A single dramatic software failure can cost a company millions of dollars - but can be avoided with simple changes to design and architecture. This new edition of the best-selling industry standard shows you how to create systems that run longer, with fewer failures, and recover better when bad things happen. New coverage includes DevOps, microservices, and cloud-native architecture. Stability antipatterns have grown to include systemic problems in large-scale systems. This is a must-have pragmatic guide to engineering for production systems. If you're a software developer, and you don't want to get alerts every night for the rest of your life, help is here. With a combination of case studies about huge losses - lost revenue, lost reputation, lost time, lost opportunity - and practical, down-to-earth advice that was all gained through painful experience, this book helps you avoid the pitfalls that cost companies millions of dollars in downtime and reputation. Eighty percent of project life-cycle cost is in production, yet few books address this topic. This updated edition deals with the production of today's systems - larger, more complex, and heavily virtualized - and includes information on chaos engineering, the discipline of applying randomness and deliberate stress to reveal systematic problems. Build systems that survive the real world, avoid downtime, implement zero-downtime upgrades and continuous delivery, and make cloud-native applications resilient. Examine ways to architect, design, and build software - particularly distributed systems - that stands up to the typhoon winds of a flash mob, a Slashdotting, or a link on Reddit. Take a hard look at software that failed the test and find ways to make sure your software survives. To skip the pain and get the experience...get this book.*

*Have we entered the age of NoOps infrastructures? Hardly. Old-style system administrators may be disappearing in the face of automation and cloud computing, but operations have become more significant than ever. As this O’Reilly Radar Report explains, we’re moving into a more complex arrangement known as "DevOps." Mike Loukides, O’Reilly’s VP of Content Strategy, provides an incisive look into this new world of operations, where IT specialists are becoming part of the development team. In an environment with thousands of servers, these specialists now write the code that maintains the infrastructure. Even applications that run in the cloud have to be resilient and fault tolerant, need to be monitored, and must adjust to huge swings in load. That was underscored by Amazon’s EBS outage last year. From the discussions at O’Reilly’s Velocity Conference, it’s evident that many operations specialists are quickly adapting to the DevOps reality. But as a whole, the industry has just scratched the surface. This report tells you why.*

*As programmers, we’ve all seen source code that’s so ugly and buggy it makes our brain ache. Over the past five years, authors Dustin Boswell and Trevor Foucher have analyzed hundreds of examples of "bad code" (much of it their own) to determine why they’re bad and how they could be improved. Their conclusion? You need to write code that minimizes the time it would take someone else to understand it—even if that someone else is you. This book focuses on basic principles and practical techniques you can apply every time you write code. Using easy-to-digest code examples from different languages, each chapter dives into a different aspect of coding, and demonstrates how you can make your code easy to understand. Simplify naming, commenting, and formatting with tips that apply to every line of code Refine your program’s loops, logic, and variables to reduce complexity and confusion Attack problems at the function level, such as reorganizing blocks of code to do one task at a time Write effective test code that is thorough and concise—as well as readable "Being aware of how the code you create affects those who look at it later is an important part of developing software. The authors did a great job in taking you through the different aspects of this challenge, explaining the details with instructive examples." —Michael Hunger, passionate Software Developer*

*Success on the web is measured by usage and growth. Web-based companies live or die by the ability to scale their infrastructure to accommodate increasing demand. This book is a hands-on and practical guide to planning for such growth, with many techniques and considerations to help you plan, deploy, and manage web application infrastructure. The Art of Capacity Planning is written by the manager of data operations for the world-famous photo-sharing site Flickr.com, now owned by Yahoo! John Allspaw combines personal anecdotes from many phases of Flickr's growth with insights from his colleagues in many other industries to give you solid guidelines for measuring your growth, predicting trends, and making cost-effective preparations. Topics include: Evaluating tools for measurement and deployment Capacity analysis and prediction for storage, database, and application servers Designing architectures to easily add and measure capacity Handling sudden spikes Predicting exponential and explosive growth How cloud services such as EC2 can fit into a capacity strategy In this book, Allspaw draws on years of valuable experience, starting from the days when Flickr was relatively small and had to deal with the typical growth pains and cost/performance trade-offs of a typical company with a Web presence. The advice he offers in The Art of Capacity Planning will not only help you prepare for explosive growth, it will save you tons of grief.*

*A Handbook for Building, Deploying, Testing and Releasing Software*

*How to Create World-Class Agility, Reliability, and Security in Technology Organizations*

*Scaling Web Resources*

*Thinking Outside the Box*

*Accelerate*

*Intellectuals Abroad*

*Sustainable Architecture in an Agile and Cloud-Centric World*

*Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader)*

*Software maintenance work is often considered a dauntingly rigid activity - this book proves the opposite: it demands high levels of creativity and thinking outside the box. Highlighting the creative aspects of software maintenance and combining analytical and systems thinking in a holistic manner, the book motivates readers not to blithely follow the beaten tracks of “technical rationality”. It delivers the content in a pragmatic fashion using case studies which are woven into long running story lines. The book is organized in four parts, which can be read in any order, except*

*for the first chapter, which introduces software maintenance and evolution and presents a number of case studies of software failures. The "Introduction to Key Concepts" briefly introduces the major elements of software maintenance by highlighting various core concepts that are vital in order to see the forest for the trees. Each such concept is illustrated with a worked example. Next, the "Forward Engineering" part debunks the myth that being fast and successful during initial development is all that matters. To this end, two categories of forward engineering are considered: an inept initial project with a multitude of hard evolutionary phases and an effective initial project with multiple straightforward future increments. "Reengineering and Reverse Engineering" shows the difficulties of dealing with a typical legacy system, and tackles tasks such as retrofitting tests, documenting a system, restructuring a system to make it amenable for further improvements, etc. Lastly, the "DevOps" section focuses on the importance and benefits of crossing the development versus operation chasm and demonstrates how the DevOps paradigm can turn a loosely coupled design into a loosely deployable solution. The book is a valuable resource for readers familiar with the Java programming language, and with a basic understanding and/or experience of software construction and testing. Packed with examples for every elaborated concept, it offers complementary material for existing courses and is useful for students and professionals alike.*

*Unleash the combination of Docker and Jenkins in order to enhance the DevOps workflow About This Book Build reliable and secure applications using Docker containers. Create a complete Continuous Delivery pipeline using Docker, Jenkins, and Ansible. Deliver your applications directly on the Docker Swarm cluster. Create more complex solutions using multi-containers and database migrations. Who This Book Is For This book is indented to provide a full overview of deep learning. From the beginner in deep learning and artificial intelligence to the data scientist who wants to become familiar with Theano and its supporting libraries, or have an extended understanding of deep neural nets. Some basic skills in Python programming and computer science will help, as well as skills in elementary algebra and calculus. What You Will Learn Get to grips with docker fundamentals and how to dockerize an application for the Continuous Delivery process Configure Jenkins and scale it using Docker-based agents Understand the principles and the technical aspects of a successful Continuous Delivery pipeline Create a complete Continuous Delivery process using modern tools: Docker, Jenkins, and Ansible Write acceptance tests using Cucumber and run them in the Docker ecosystem using Jenkins Create multi-container applications using Docker Compose Managing database changes inside the Continuous Delivery process and understand effective frameworks such as Cucumber and Flyweight Build clustering applications with Jenkins using Docker Swarm Publish a built Docker image to a Docker Registry and deploy cycles of Jenkins pipelines using community best practices In Detail The combination of Docker and Jenkins improves your Continuous Delivery pipeline using fewer resources. It also helps you scale up your builds, automate tasks and speed up Jenkins performance with the benefits of Docker containerization. This book will explain the advantages of combining Jenkins and Docker to improve the continuous integration and delivery process of app development. It will start with setting up a Docker server and configuring Jenkins on it. It will then provide steps to build applications on Docker files and integrate them with Jenkins using continuous delivery processes such as continuous integration, automated acceptance testing, and configuration management. Moving on you will learn how to ensure quick application deployment with Docker containers along with scaling Jenkins using Docker Swarm. Next, you will get to know how to deploy applications using Docker images and testing them with Jenkins. By the end of the book, you will be enhancing the DevOps workflow by integrating the functionalities of Docker and Jenkins. Style and approach The book is aimed at DevOps Engineers, developers and IT Operations who want to enhance the DevOps culture using Docker and Jenkins.*

*Increase profitability, elevate work culture, and exceed productivity goals through DevOps practices. More than ever, the effective management of technology is critical for business competitiveness. For decades, technology leaders have struggled to balance agility, reliability, and security. The consequences of failure have never been greater—whether it's the healthcare.gov debacle, cardholder data breaches, or missing the boat with Big Data in the cloud. And yet, high performers using DevOps principles, such as Google, Amazon, Facebook, Etsy, and Netflix, are routinely and reliably deploying code into production hundreds, or even thousands, of times per day. Following in the footsteps of The Phoenix Project, The DevOps Handbook shows leaders how to replicate these incredible outcomes, by showing how to integrate Product Management, Development, QA, IT Operations, and Information Security to elevate your company and win in the marketplace.*

*Six years ago, Infrastructure as Code was a new concept. Today, as even banks and other conservative organizations plan moves to the cloud, development teams for companies worldwide are attempting to build large infrastructure codebases. With this practical book, Kief Morris of ThoughtWorks shows you how to effectively use principles, practices, and patterns pioneered by DevOps teams to manage cloud-age infrastructure. Ideal for system administrators, infrastructure engineers, software developers, team leads, and architects, this updated edition demonstrates how you can exploit cloud and automation technology to make changes easily, safely, quickly, and responsibly. You'll learn how to define everything as code and apply software design and engineering practices to build your system from small, loosely coupled pieces. This book covers: Foundations: Use Infrastructure as Code to drive continuous change and raise the bar of operational quality, using tools and technologies to build cloud-based platforms Working with infrastructure stacks: Learn how to define, provision, test, and continuously deliver changes to infrastructure resources Working with servers and other platforms: Use patterns to design provisioning and configuration of servers and clusters Working with large systems and teams: Learn workflows, governance, and architectural patterns to create and manage infrastructure elements*

*Design and architect highly scalable and robust applications using Go*

*Design and Deploy Production-Ready Software*

*With Jenkins 2.0*

*6th International Conference, DUXU 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9-14, 2017, Proceedings, Part I*

*Pro Continuous Delivery*

*Continuous Delivery*

*DevOps for Digital Leaders*

*Building a Culture of Collaboration, Affinity, and Tooling at Scale*

*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*

*Writing for students at all levels of experience, Farley illuminates durable principles at the heart of effective software development. He distills the discipline into two core exercises: first, learning and exploration, and second, managing complexity. For each, he defines principles that can help students improve everything from their mindset to the quality of their code, and describes approaches proven to promote success. Farley's ideas and techniques cohere into a unified, scientific, and foundational approach to solving practical software development problems within realistic economic constraints. This general, durable, and pervasive approach to software engineering can help students solve problems they haven't encountered yet, using today's technologies and tomorrow's. It offers students deeper insight into what they do every day, helping them create better software, faster, with more pleasure and personal fulfillment.*

*Using Continuous Delivery, you can bring software into production more rapidly, with greater reliability. A Practical Guide to Continuous Delivery is a 100% practical guide to building Continuous Delivery pipelines that automate rollouts, improve reproducibility, and dramatically reduce risk. Eberhard Wolff introduces a proven Continuous Delivery technology stack, including Docker, Chef, Vagrant, Jenkins, Graphite, the ELK stack, JBehave, and Gatling. He guides you through applying these technologies throughout build, continuous integration, load testing, acceptance testing, and monitoring. Wolff's start-to-finish example projects offer the basis for your own experimentation, pilot programs, and full-fledged deployments. A Practical Guide to Continuous Delivery is for everyone who wants to introduce Continuous Delivery, with or without DevOps. For managers, it introduces core processes, requirements, benefits, and technical consequences. Developers, administrators, and architects will gain essential skills for implementing and managing pipelines, and for integrating Continuous Delivery smoothly into software architectures and IT organizations. Understand the problems that Continuous Delivery solves, and how it solves them Establish an infrastructure for maximum software automation Leverage virtualization and Platform as a Service (PAAS) cloud solutions Implement build automation and continuous integration with Gradle, Maven, and Jenkins Perform static code reviews with SonarQube and repositories to store build artifacts Establish automated GUI and textual acceptance testing with behavior-driven design Ensure appropriate performance via capacity testing Check new features and problems with exploratory testing Minimize risk throughout automated production software rollouts Gather and analyze metrics and logs with Elasticsearch, Logstash, Kibana (ELK), and Graphite Manage the introduction of Continuous Delivery into your enterprise Architect software to facilitate Continuous Delivery of new capabilities*

*Understand the principles of software architecture with coverage on SOA, distributed and messaging systems, and database modeling Key FeaturesGain knowledge of architectural approaches on SOA and microservices for architectural decisionsExplore different architectural patterns for building distributed applicationsMigrate applications written in Java or Python to the Go languageBook Description Building software requires careful planning and architectural considerations; Golang was developed with a fresh perspective on building next-generation applications on the cloud with distributed and concurrent computing concerns. Hands-On Software Architecture with Golang starts with a brief introduction to architectural elements, Go, and a case study to demonstrate architectural principles. You'll then move on to look at code-level aspects such as modularity, class design, and constructs specific to Golang and implementation of design patterns. As you make your way through the chapters, you'll explore the core objectives of architecture such as effectively managing complexity, scalability, and reliability of software systems. You'll also work through creating distributed systems and their communication before moving on to modeling and scaling of data. In the concluding chapters, you'll learn to deploy architectures and plan the migration of applications from other languages. By the end of this book, you will have gained insight into various design and architectural patterns, which will enable you to create robust, scalable architecture using Golang. What you will learnUnderstand architectural paradigms and deep dive into MicroservicesDesign parallelism/concurrency patterns and learn object-oriented design patterns in GoExplore API-driven systems architecture with introduction to REST and GraphQL standardsBuild event-driven architectures and make your architectures anti-fragileEngineer scalability and learn how to migrate to Go from other languagesGet to grips with deployment considerations with CICD pipeline, cloud deployments, and so onBuild an end-to-end e-commerce (travel) application backend in GoWho this book is for Hands-On Software Architecture with Golang is for software developers, architects, and CTOs looking to use Go in their software architecture to build enterprise-grade applications. Programming knowledge of Golang is assumed.*

*Reliable Software Releases Through Build, Test, and Deployment Automation*

*Modernist Travel Writing*

*Design, Build, Ship*

*Proven recipes to accelerate your DevOps journey with Azure DevOps Server 2019 (formerly TFS), 2nd Edition*

*Continuous Delivery with Jenkins, Kubernetes, and Terraform*

*A Practical Guide to Continuous Delivery*

*Continuous Delivery in Java*

Web services have been used for many years. In this time, developers and architects have encountered a number of recurring design challenges related to their usage, and have learned that certain service design approaches work better than others to solve certain problems. In Service Design Patterns, Rob Daigneau codifies proven design solutions for web services that follow the REST architectural style or leverage the SOAP/WSDL specifications. This catalogue identifies the fundamental topics in web service design and lists the common design patterns for each topic. All patterns identify the context in which they may be used, explain the constituent design elements, and explore the relative strengths and trade-offs. Code examples are provided to help you better understand how the patterns work but are kept general so that you can see how the solutions may be applied to disparate technologies that will inevitably change in the years to come. This book will help readers answer the following questions: How do you create a web service API, what are the common API styles, and when should a particular style be used? How can clients and web services communicate, and what are the foundations for creating complex conversations in which multiple parties exchange data over extended periods of time? What are the options for implementing web service logic, and when should a particular approach be used? How can clients become less coupled to the underlying systems used by a service? How can information about a web service be discovered? How can generic functions like authentication, validation, caching, and logging be supported on the client or service? What changes to a service cause clients to break? What are the common ways to version a service? How can web services be designed to support the continuing evolution of business logic without forcing clients to constantly upgrade? This book is an invaluable resource for enterprise architects, solution architects, and developers who use web services to create enterprise IT applications, commercial or open source products, and Software as a Service (SaaS) products that leverage emerging Cloud platforms.

Start thinking about your development pipeline as a mission-critical application. Discover techniques for implementing code-driven infrastructure and CI/CD workflows using Jenkins, Docker, Terraform, and cloud-native services. In Pipeline as Code, you will master: Building and deploying a Jenkins cluster from scratch Writing pipeline as code for cloud-native applications Automating the deployment of Dockerized and Serverless applications Containerizing applications with Docker and Kubernetes Deploying Jenkins on AWS, GCP and Azure Managing, securing and monitoring a Jenkins cluster in production Key principles for a successful DevOps culture Pipeline as Code is a practical guide to automating your development pipeline in a cloud-native, service-driven world. You'll use the latest infrastructure-as-code tools like Packer and Terraform to develop reliable CI/CD pipelines for numerous cloud-native applications. Follow this book's insightful best practices, and you'll soon be delivering software that's quicker to market, faster to deploy, and with less last-minute production bugs. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the technology Treat your CI/CD pipeline like the real application it is. With the Pipeline as Code approach, you create a collection of scripts that replace the tedious web UI wrapped around most CI/CD systems. Code-driven pipelines are easy to use, modify, and maintain, and your entire CI pipeline becomes more efficient because you directly interact with core components like Jenkins, Terraform, and Docker. About the book In Pipeline as Code you'll learn to build reliable CI/CD pipelines for cloud-native applications. With Jenkins as the backbone, you'll programmatically control all the pieces of your pipeline via modern APIs. Hands-on examples include building CI/CD workflows for distributed Kubernetes applications, and serverless functions. By the time you're finished, you'll be able to swap manual UI-based adjustments with a fully automated approach! What's inside Build and deploy a Jenkins cluster on scale Write pipeline as code for cloud-native applications Automate the deployment of Dockerized and serverless applications Deploy Jenkins on AWS, GCP, and Azure Grasp key principles of a successful DevOps culture About the reader For developers familiar with Jenkins and Docker. Examples in Go. About the author Mohamed Labouardy is the CTO and co-founder of Crew.work, a Jenkins contributor, and a DevSecOps evangelist. Table of Contents PART 1 GETTING STARTED WITH JENKINS 1 What's CI/CD? 2 Pipeline as code with Jenkins PART 2 OPERATING A SELF-HEALING JENKINS CLUSTER 3 Defining Jenkins architecture 4 Baking machine images with Packer 5 Discovering Jenkins as code with Terraform 6 Deploying HA Jenkins on multiple cloud providers PART 3 HANDS-ON CI/CD PIPELINES 7 Defining a pipeline as code for microservices 8 Running automated tests with Jenkins 9 Building Docker images within a CI pipeline 10 Cloud-native applications on Docker Swarm 11 Dockerized microservices on K8s 12 Lambda-based serverless functions PART 4 MANAGING, SCALING, AND MONITORING JENKINS 13 Collecting continuous delivery metrics 14 Jenkins administration and best practices

The three-volume set LNCS 10288, 10289, and 10290 constitutes the proceedings of the 6th International Conference on Design, User Experience, and Usability, DUXU 2017, held as part of the 19th International Conference on Human-Computer Interaction, HCII 2017, in Vancouver, BC, Canada, in July 2017, jointly with 14 other thematically similar conferences. The total of 1228 papers presented at the HCII 2017 conferences were carefully reviewed and selected from 4340 submissions. These papers address the latest research and development efforts and highlight the human aspects of design and use of computing systems. The papers accepted for presentation thoroughly cover the entire field of Human-Computer Interaction, addressing major advances in knowledge and effective use of computers in a variety of application areas. The total of 168 contributions included in the DUXU proceedings were carefully reviewed and selected for inclusion in this three-volume set. LNCS 10288: The 56 papers included in this volume are organized in topical sections on design thinking and design philosophy; aesthetics and perception in design; user experience evaluation methods and tools; user centered design in the software development lifecycle; DUXU education and training. LNCS 10289: The 56 papers included in this volume are organized in topical sections on persuasive and emotional design; mobile DUXU; designing the playing experience; designing the virtual, augmented and tangible experience; wearables and fashion technology. LNCS 10290: The 56 papers included in this volume are organized in topical sections on information design; understanding the user; DUXU for children and young users; DUXU for art, culture, tourism and environment; DUXU practice and case studies.