O'REILLY®

Kubernetes Patterns

Reusable Elements for Designing Cloud-Native Applications

Grayscale Edition

For Sale in the Indian Subcontinent & Select Countries Only*

*Refer Back Cover

SPD®

Bilgin Ibryam &
Roland Huß

Kubernetes Patterns

Reusable Elements for Designing Cloud-Native Applications

Bilgin Ibryam and Roland Huß

Beijing • Boston • Farnham • Sebastopol • Tokyo

O'REILLY®

SPD®

SHROFF PUBLISHERS & DISTRIBUTORS PVT. LTD.

Mumbai Bangalore Kolkata New Delhi

Kubernetes Patterns

by Bilgin Ibryam and Roland Huß

 ISBN: 978-1-492-05028-5
Originally printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles *(http://safari.oreilly.com)*. For more information, contact our corporate/institutional sales department: (800) 998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: John Devins
Developmental Editors: Virginia Wilson
Production Editor: Katherine Tozer
Copyeditor: Christine Edwards
Proofreader: Sharon Wilkey
Indexer: Judith McConville
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

Printing History:
May 2019: First Edition

Revision History for the First Edition 2019-04-04: First Release

See https://www.oreilly.com/catalog/errata.csp?isbn=9781492050285 for release details.

First Indian Reprint: April 2019

ISBN: 978-93-5213-826-5

Published by **Shroff Publishers & Distributors Pvt. Ltd.** B-103, Railway Commercial Complex, Sector 3, Sanpada (E), Navi Mumbai 400705 • TEL: (91 22) 4158 4158 • FAX: (91 22) 4158 4141 E-mail:spdorders@shroffpublishers.com•Web:www.shroffpublishers.com Printed at Jasmine Art Printers Pvt. Ltd., Mumbai.

Table of Contents

Part IV. Configuration Patterns

Part V. Advanced Patterns

Foreword

When Craig, Joe, and I started Kubernetes nearly five years ago, I think we all recognized its power to transform the way the world developed and delivered software. I don't think we knew, or even hoped to believe, how quickly this transformation would come. Kubernetes is now the foundation for the development of portable, reliable systems spanning the major public clouds, private clouds, and bare-metal environments. However, even as Kubernetes has become ubiquitous to the point where you can spin up a cluster in the cloud in less than five minutes, it is still far less obvious to determine where to go once you have created that cluster. It is fantastic that we have seen such significant strides forward in the operationalization of Kubernetes itself, but it is only a part of the solution. It is the foundation on which applications will be built, and it provides a large library of APIs and tools for building these applications, but it does little to provide the application architect or developer with any hints or guidance for how these various pieces can be combined into a complete, reliable system that satisfies their business needs and goals.

Although the necessary perspective and experience for what to do with your Kubernetes cluster can be achieved through past experience with similar systems, or via trial and error, this is expensive both in terms of time and the quality of systems delivered to our end users. When you are starting to deliver mission-critical services on top of a system like Kubernetes, learning your way via trial and error simply takes too much time and results in very real problems of downtime and disruption.

This then is why Bilgin and Roland's book is so valuable. *Kubernetes Patterns* enables you to learn from the previous experience that we have encoded into the APIs and tools that make up Kubernetes. Kubernetes is the by-product of the community's experience building and delivering many different, reliable distributed systems in a variety of different environments. Each object and capability added to Kubernetes represents a foundational tool that has been designed and purpose-built to solve a specific need for the software designer. This book explains how the concepts in Kubernetes solve real-world problems and how to adapt and use these concepts to build the system that you are working on today.

In developing Kubernetes, we always said that our North Star was making the development of distributed systems a CS 101 exercise. If we have managed to achieve that goal successfully, it is books like this one that are the textbooks for such a class. Bilgin and Roland have captured the essential tools of the Kubernetes developer and distilled them into segments that are easy to approach and consume. As you finish this book, you will become aware not just of the components available to you in Kubernetes, but also the "why" and "how" of building systems with those components.

— Brendan Burns,
Cofounder, Kubernetes

Preface

With the evolution of microservices and containers in recent years, the way we design, develop, and run software has changed significantly. Today's applications are optimized for scalability, elasticity, failure, and speed of change. Driven by new principles, these modern architectures require a different set of patterns and practices. This book aims to help developers create cloud-native applications with Kubernetes as a runtime platform. First, let's take a brief look at the two primary ingredients of this book: Kubernetes and design patterns.

Kubernetes

Kubernetes is a container orchestration platform. The origin of Kubernetes lies somewhere in the Google data centers where Google's internal container orchestration platform, Borg, was born. Google used Borg for many years to run its applications. In 2014, Google decided to transfer its experience with Borg into a new open source project called "Kubernetes" (Greek for "helmsman" or "pilot"), and in 2015, it became the first project donated to the newly founded Cloud Native Computing Foundation (CNCF).

Right from the start, Kubernetes gained a whole community of users, and the number of contributors grew at an incredibly fast pace. Today, Kubernetes is considered one of the most active projects on GitHub. It is probably fair to claim that at the time of this writing, Kubernetes is the most commonly used and feature-rich container orchestration platform. Kubernetes also forms the foundation of other platforms built on top of it. The most prominent of those Platform-as-a-Service systems is Red Hat OpenShift, which provides various additional capabilities to Kubernetes, including ways to build applications within the platform. These are only some of the reasons we chose Kubernetes as the reference platform for the cloud-native patterns in this book.

This book assumes you have some basic knowledge of Kubernetes. In Chapter 1, we recapitulate the core Kubernetes concepts and lay out the foundation for the following patterns.

Design Patterns

The concept of *design patterns* dates back to the 1970s and from the field of architecture. Christopher Alexander, an architect and system theorist, and his team published the groundbreaking *A Pattern Language* (Oxford University Press) in 1977, which describes architectural patterns for creating towns, buildings, and other construction projects. Sometime later this idea was adopted by the newly formed software industry. The most famous book in this area is *Design Patterns—Elements of Reusable Object-Oriented Software* by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides—the Gang of Four (Addison-Wesley). When we talk about the famous Singleton, Factories, or Delegation patterns, it's because of this defining work. Many other great pattern books have been written since then for various fields with different levels of granularity, like *Enterprise Integration Patterns* by Gregor Hohpe and Bobby Woolf (Addison-Wesley) or *Patterns of Enterprise Application Architecture* by Martin Fowler (Addison-Wesley).

In short, a pattern describes a *repeatable solution to a problem*.[1] It is different from a recipe because instead of giving step-by-step instructions to solving a problem, a pattern provides a blueprint for solving a whole class of similar problems. For example, the Alexandrian pattern "Beer Hall" describes how public drinking halls should be constructed where "strangers and friends are drinking companions" and not "anchors of the lonely." All halls built after this pattern look different, but share common characteristics such as open alcoves for groups of four to eight and a place where a hundred people can meet with beverages, music, and other activities.

However, a pattern does more than provide a solution. It is also about forming a language. The unique pattern names form a dense, noun-centric language in which each pattern carries a unique *name*. When this language is established, these names automatically evoke similar mental representations when people speak about these patterns. For example, when we talk about a table, anyone speaking English assumes we are talking about a piece of wood with four legs and a top on which you can put things. The same thing happens in software engineering when we talk about a "factory." In an object-oriented programming language context, we immediately associate with a "factory" an object that produces other objects. Because we immediately know the solution behind the pattern, we can move on to tackle yet unsolved problems.

1 Christopher Alexander and his team defined the original meaning in the context of architecture as follows: "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice," (*A Pattern Language*, Christopher Alexander et al., 1977, p. x). We think this definition works for the patterns we describe in this book, except that we probably don't have as much variability in our solutions.

O'REILLY®

Kubernetes Patterns

The way developers design, build, and run software has changed significantly with the evolution of microservices and containers. These modern architectures offer new distributed primitives that require a different set of practices than many developers, tech leads, and architects are accustomed to. This focused guide provides common, reusable patterns and principles for designing and implementing cloud-native applications on Kubernetes.

Each pattern includes a description of the problem and a Kubernetes-specific solution. All patterns are backed by and demonstrated with concrete code examples. This book is ideal for developers and architects already familiar with basic Kubernetes concepts who want to learn how to solve common cloud native challenges with proven design patterns.

You'll learn about the following pattern categories:

- **Foundational patterns** cover core principles and practices for building container-based cloud-native applications
- **Behavioral patterns** explore finer-grained concepts for managing container and platform interactions
- **Structural patterns** help you organize containers within a Pod for addressing specific use cases
- **Configuration patterns** provide insight into how application configurations can be handled in Kubernetes
- **Advanced patterns** cover more complex topics such as operators and autoscaling

Bilgin Ibryam (@bibryam) is a principal architect at Red Hat and a committer to multiple projects at the Apache Software Foundation.

Roland Huß (@ro14nd) is a principal software engineer at Red Hat and a member of the serverless team working on Knative.

"As you finish this book, you become aware of not just the components that are available to you in Kubernetes but also the why and how of building systems with those components."

—**Brendan Burns**
Cofounder, Kubernetes

"A unique approach that introduces key Kubernetes concepts in a format that developers can understand and quickly take advantage of."

—**Andrew Block**
Senior Principal Consultant, Red Hat

"A great book that explains Kubernetes aligned with real-world tasks and challenges."

—**Michael Hüttermann**
Principal DevOps Consultant, Huettermann.net

KUBERNETES / SOFTWARE DEVELOPMENT

MRP: ₹ 750.00

Twitter: @oreillymedia
facebook.com/oreilly

SPD®
SHROFF PUBLISHERS & DISTRIBUTORS PVT. LTD.

ISBN : 978-93-5213-826-5

9 789352 138265

First Edition/2019/Paperback/English