

# The Patterns of Cloud-native Architecture

---



**Richard Seroter**

SENIOR DIRECTOR OF PRODUCT, PIVOTAL

@rseroter



# Overview



Application architecture patterns

Application delivery patterns

Application infrastructure patterns

Application team patterns

Summary



“Cloud-native software is built for scale, built for continuous change, built to tolerate failure, built for manageability.”

**Richard Seroter**



# Application Architecture Patterns

---



# 12-Factor Applications



1. One codebase in source control
2. Declared dependencies
3. Config stored in the environment
4. Backing services as attached resources
5. Separate build and run stages
6. App executed as stateless processes
7. Services exported as port bindings
8. Scale out processes
9. Disposability
10. Environment parity
11. Treat logs like event streams
12. Run admin processes as one-off processes

# Microservices Architecture

**Boundaries found  
via domain-driven  
design**

**Loosely coupled  
components**

**Continuously  
delivered**

**Surgical scaling**

**Contract-driven  
tests**

**Organized around  
teams**



# The Supporting Infrastructure for Microservices



Service discovery



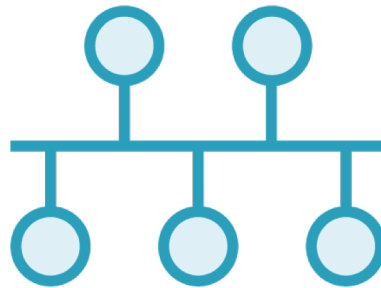
Circuit breaker



Externalized  
configuration



Token-based security



Messaging



API gateway

# Modern Data Management



Scalable, on-demand databases per microservice



Favor event sourcing and CQRS pattern



Use intelligent caching to improve resilience





# Application Delivery Patterns

---



# Fast Feedback via Continuous integration



Version control is a must

Trunk-based development

Often coupled with test-driven development

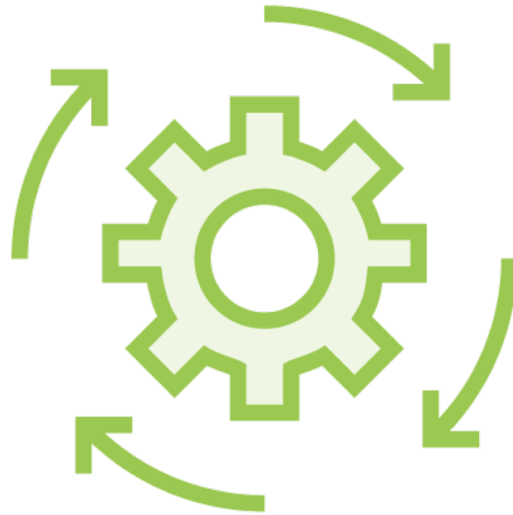
CI jobs triggered on code check-in

Test in production-like environments

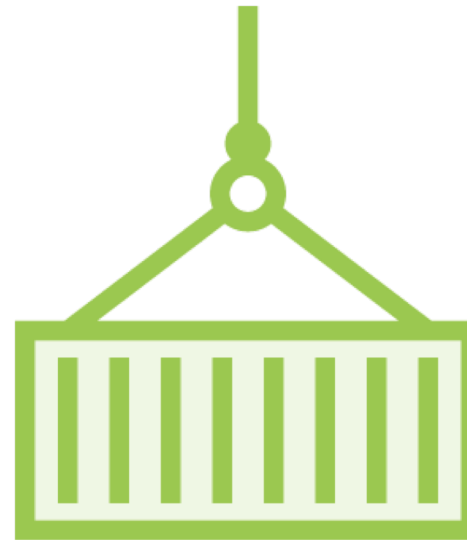
# Packaging up Software



**Include  
configuration  
and code**



**Build service  
generates  
artifacts**



**Containers are a  
useful  
packaging  
structure**



**Artifact  
repositories play  
a key role**



# Continuously Deliver Value

**Small changes, regularly shipped**

**Same binaries in each environment**

**Multiple strategies for low-impact deployments**

**Smoke test and watch metrics to ensure healthy releases**

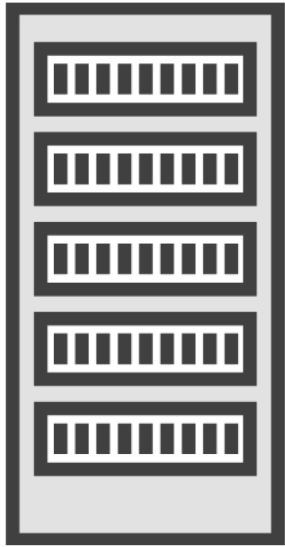


# Application Infrastructure Patterns

---



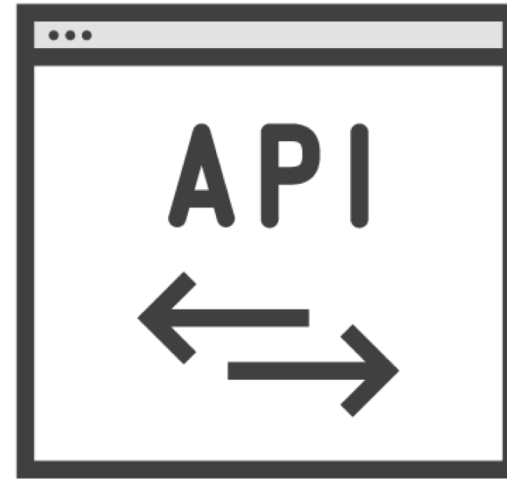
# API-driven Infrastructure



Immutable  
infrastructure



Observable  
systems



APIs for  
interacting with  
infrastructure



Chaos  
engineering to  
fight fragility

# Application Team Patterns

---



# Empowered, Customer-Focused Teams

DevOps-style  
teams build and  
run services

Platform Ops for  
managing  
underlying  
systems

Site Reliability  
Engineering  
applies software  
engineering  
approach to  
operations





# Summary



## Overview

Application architecture patterns

Application delivery patterns

Application infrastructure patterns

Application team patterns

