

## The Legacy Trap

How to modernize applications that are holding you back, and why you need to start right now

By Michael Coté and Marc Zottner

*Get it!*



“ I could do it in  
a weekend.”



“IDC estimates there are

**500 million legacy apps**

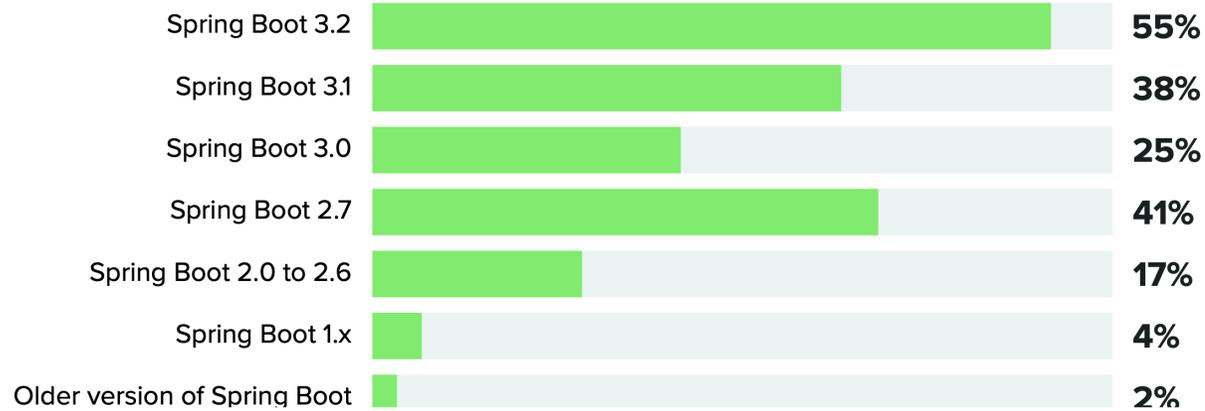
in use around the world.

The vast majority are found on x86 servers,  
with most running on virtual machines (VMs).”

**76%** of executives say  
legacy software is holding  
them back.

## Versions of Spring Boot in Use

2024



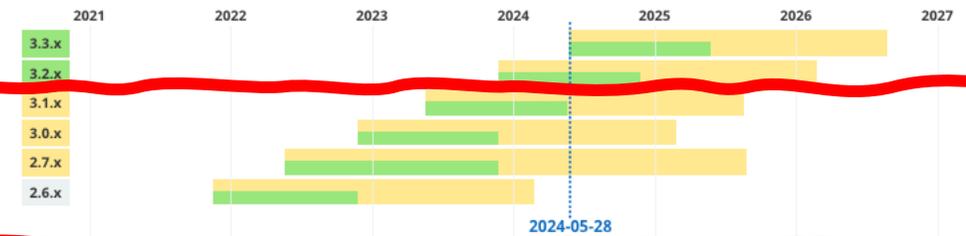
## Spring Boot 3.3.0



OVERVIEW LEARN **SUPPORT** SAMPLES

Branch	Initial Release	End of Support	End Commercial Support *
3.3.x	2024-05-23	2025-05-23	2026-08-23
3.2.x	2023-11-23	2024-11-23	2026-02-23
3.1.x	2023-05-18	2024-05-18	2025-08-18
3.0.x	2022-11-24	2023-11-24	2025-02-24
2.7.x	2022-05-19	2023-11-24	2025-08-24
2.6.x	2021-11-17	2022-11-24	2024-02-24

More ▼



More ▼

### ■ OSS support

Free security updates and bugfixes with support from the Spring community. See [VMware Tanzu OSS support policy](#).

### ■ Commercial support

Business support from Spring experts during the OSS timeline, plus extended support after OSS End-Of-Life.

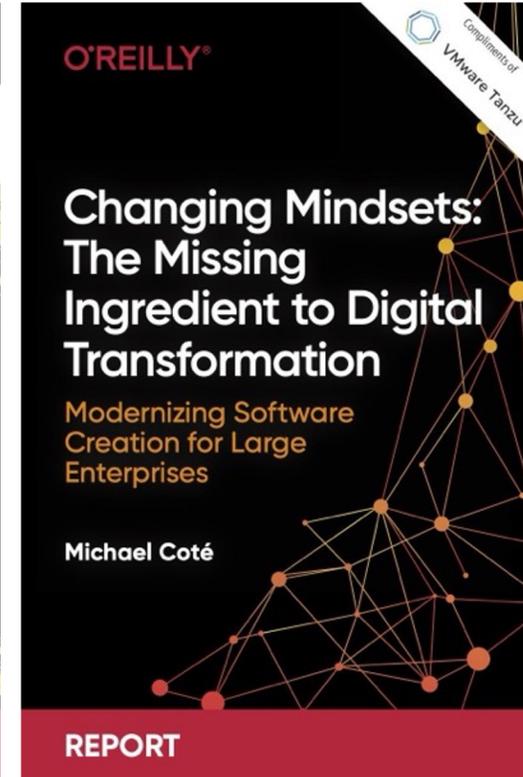
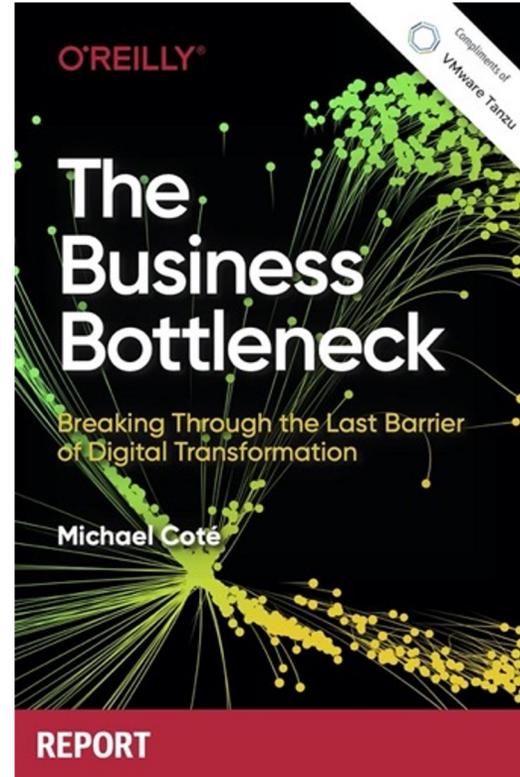
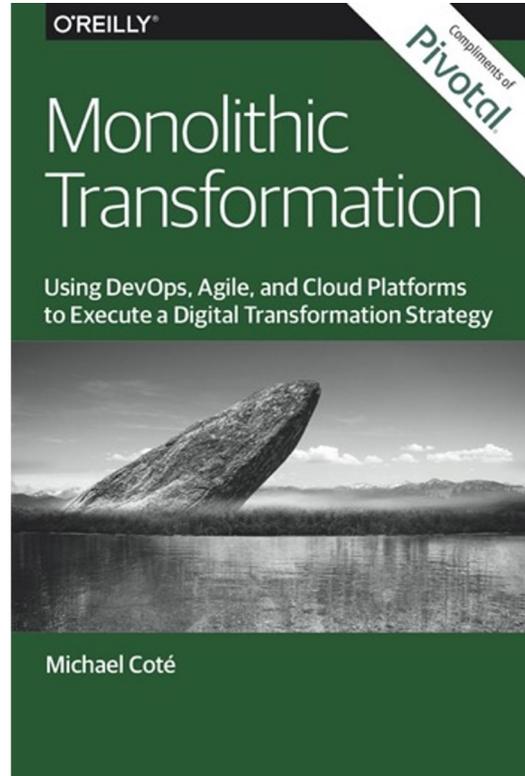
See [Tanzu Spring Runtime](#) for more details.

### ■ Future release

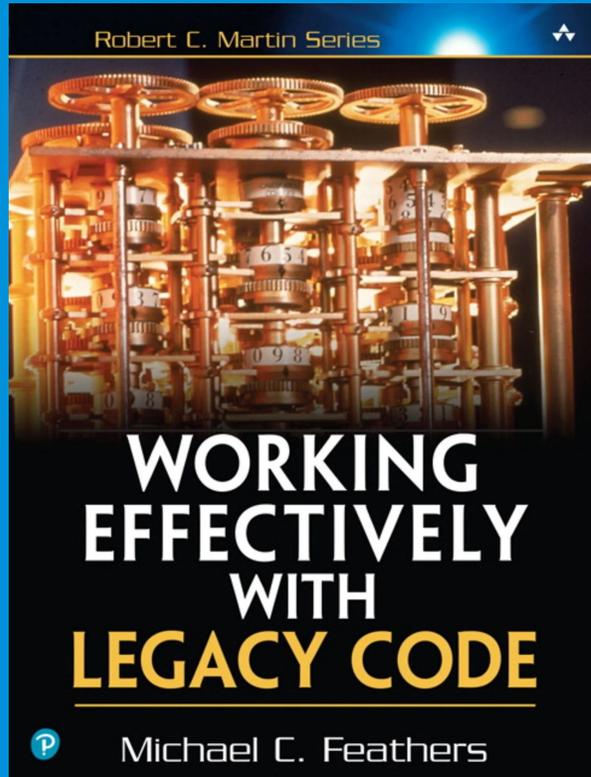
Generation not yet released, timeline is subject to changes.

# Coté

<https://newsletter.cote.io/> | [cote@broadcom.com](mailto:cote@broadcom.com)



# What you're looking for



## Application Portfolio Auditor

vmware Application Portfolio Auditor

Reports overview

Consolidated list of all generated reports for the **large** application group. (CSV)

52 APPS ANALYZED BY 15 ANALYSIS TOOLS

- CLOUD-READINESS**
  - Cloud Suitability Analyzer (CSA) v.4.1.9: Cloud readiness assessment with support for prioritization. (report - log)
  - Windup v.6.1.11: Detection of proprietary code, dependencies, and technologies. (report - csv - packages - log)
  - IBM WebSphere Application Migration Toolkit (WAMT) v.23.0.0.4: Identifies required changes for IBM WebSphere migrations. (reports - log)
- SECURITY**
  - Open Web Application Security Project (OWASP) Dependency-Check v.8.4.2: Identifies well-known security vulnerabilities in embedded libraries. (reports - log)
  - Find Security Bugs (FSB) v.1.12.0: Audits security of Java applications. (reports - log)
  - ShiftLeft SAST Scan v.2.1.1: Identifies security vulnerabilities focussing on the OWASP Top 10. (reports - log)
  - Insider SAST v.3.0.0: Identifies locally various kinds of security issues in application code. (reports - log)
  - Grype v.0.72.0 and Syft v.0.94.0: Identifies known security vulnerabilities in application binaries and code. (reports - log)
  - Trivy v.0.46.1: Find vulnerabilities, misconfigurations, secrets. (reports - log)
- QUALITY**
  - PMD Source Code Analyzer v.6.55.0: Extensible cross-language static code analyzer. (law & copy; paste reports - log)
  - ScanCode Toolkit v.32.0.8: Detects licenses, copyrights, and package manifests of embedded libraries. (report - log)
  - Microsoft Application Inspector v.1.9.14: Identifies what is in the applications using static analysis. (reports - log)
  - GitHub Linguist v.7.27.0 and CLOC v.1.98: Analyze language usage and generate language breakdown graphs. (report - log)

## Spring Application Advisor

Merged [Auto] Upgrading spring-boot to 3.0.x, spring-framework to 6.0.x and spring-security to 6.0.x spring-advisor-to-6.0.x-3... Into main

Overview 1 Commits 1 Pipelines 1 Changes 26 Add a to do

Search (e.g. \*.vue) (%P)

src/main/java/org/springframework/samples/petclinic/owner/Owner.java +10 -10 Viewed

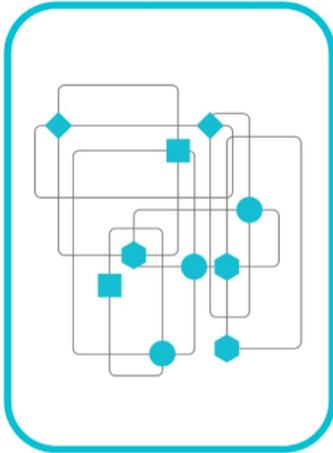
```
28 + import jakarta.persistence.Table;
29 + import jakarta.validation.constraints.Digits;
30 + import jakarta.validation.constraints.NotEmpty;
31 31
32 32 import org.springframework.core.style.ToStringCreator;
33 33 import org.springframework.samples.petclinic.model.Person;
```

src/main/java/org/springframework/samples/petclinic/owner/OwnerController.java +5 -5 Viewed

```
17 17 @@ -17,7 +17,7 @@ package org.springframework.samples.petclinic.owner;
18 18 import java.util.List;
19 19 import java.util.Map;
20 - import javax.validation.Valid;
20 + import jakarta.validation.Valid;
21 21 import org.springframework.data.domain.Page;
22 22 import org.springframework.data.domain.PageRequest;
23 23 import org.springframework.data.domain.Pageable;
24 24
25 25 @@ -56,7 +56,7 @@ class OwnerController {
56 56 }
57 57
58 58 @ModelAttribute("owner")
59 - public Owner findOwner(@PathVariable(name = "ownerId", required = false)
59 + Integer ownerId) {
60 60 public Owner findOwner(@PathVariable(required = false) Integer ownerId) {
61 61 return ownerId == null ? new Owner() : this.owners.findById(ownerId);
62 62 }
63 63
64 64 @@ -126,7 +126,7 @@ class OwnerController {
```

# The Swift Methodology

A series of proven practices that jump-start application modernization in days, not months



**1** Define objectives and key results 1-2 hours



Agree on objectives and determine the key results to measure using an objectives and key results (OKR) framework.

**Benefit**  
Ensure desired business outcomes always guide modernization practices.

**2** Event storm the system 1-2 days



Make sense of the complexity by discovering and visualizing business events that cover key features of the system.

**Benefit**  
Quickly create a common understanding between business experts and IT on the provided functionalities of the modernized system.

**3** Select one thin slice of functionalities 2 hours



Identify meaningful modernization starting points by selecting one thin slice of functionalities from the system.

**Benefit**  
Avoid analysis paralysis by having one meaningful, well-defined business narrative jump-start modernization.



Model relationships between business capabilities the system will support to draft the desired notional architecture.

**Benefit**  
Determine how the system wants to be designed and avoid pitfalls such as premature solutioning.



Capture user stories, required APIs, data, and connections for each business capability of the system.

**Benefit**  
Identify and prioritize a backlog of work to balance business value, technical risk, and effort.



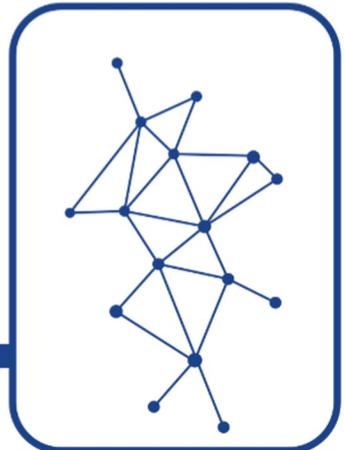
Make low-level design decisions, apply technical patterns, and implement capabilities within a time-bounded set of 1-week iterations.

**Benefit**  
Enable your team by pairing with Tanzu Labs engineers on a backlog. Then progressively transfer traffic from the existing system without disrupting business.

**4** Conduct a Boris exercise 2-4 days

**5** Fill the backlog 1-2 days

**6** Produce tested and working code Approx. 3 weeks



# Software that you need to change, but are afraid to change

“Legacy technology is any technology that makes it difficult for organizations to change their application systems to support changing business requirements. And, therefore, it impedes business agility.”

*Anne Thomas, Gartner,*  
*CIO Dive*

“Legacy code is code without unit tests.”

*Michael Feathers,*  
*Working Effectively with Legacy Code*

# What it means to be caught in the Legacy Trap

**Pace of change needed by the business over-weights application speed to market.**

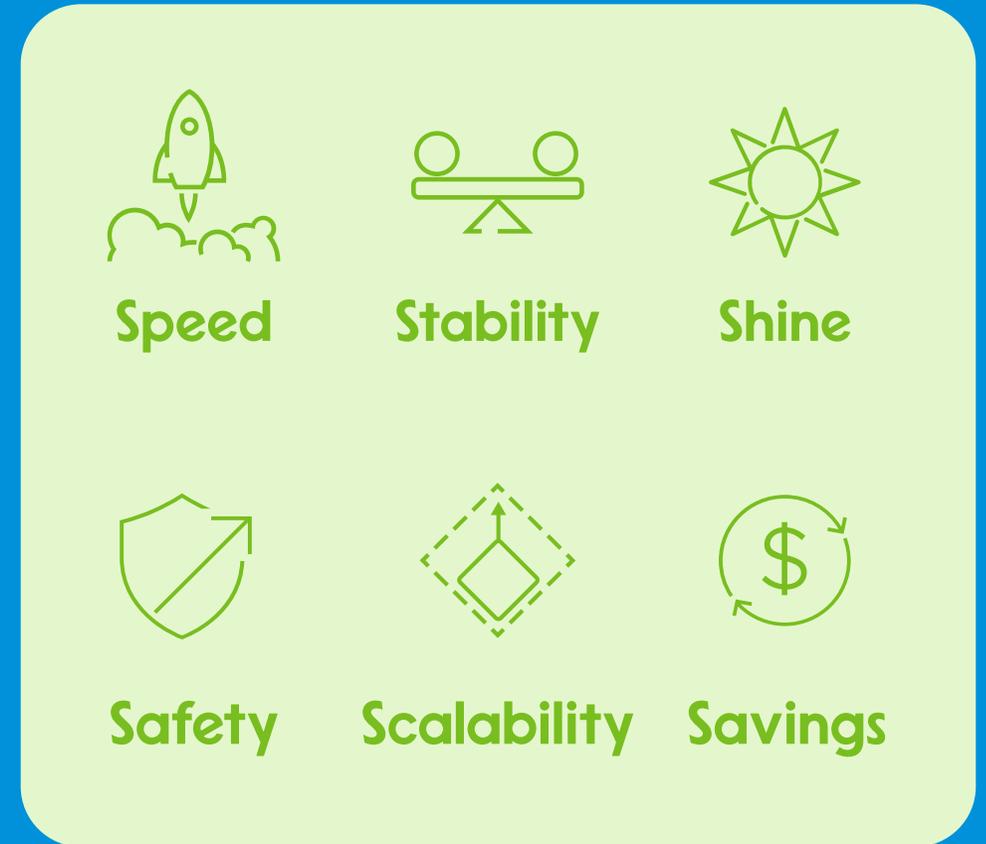
- **Debilitating technical debt**
- **Maintenance efforts hinder innovation**
- **Bulky processes obstruct progress**
- **Desperate lack of knowledge and skills**
- **Low test coverage**



# Why Our Customers Modernize?

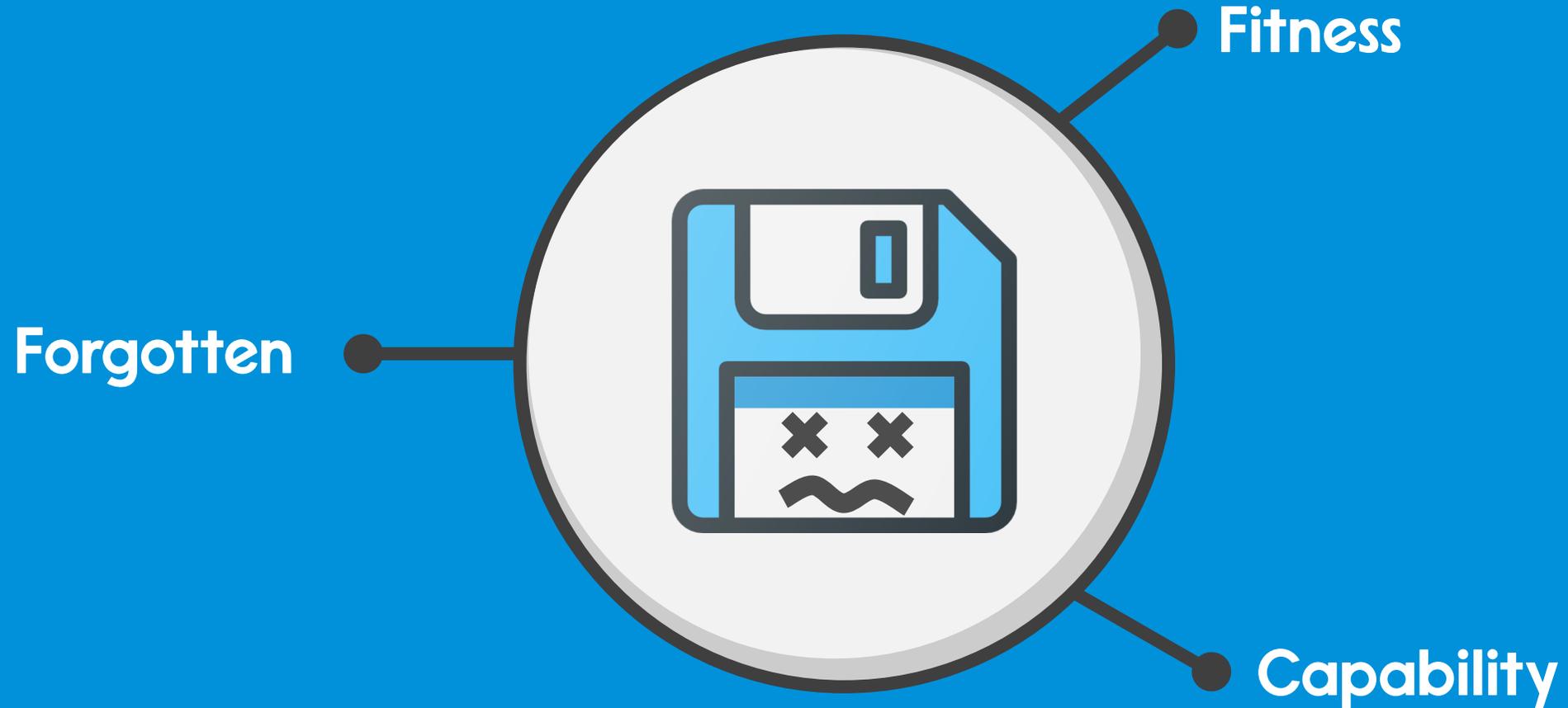


**Modernization Trends**



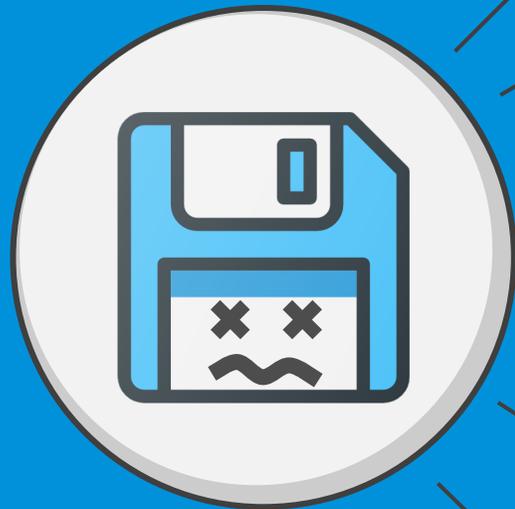
**Business Outcomes (6S)**

# How Software Goes Bad



# Common Fixes

## Modernization Strategies - The 7 Rs



### Retain

Keep and don't touch for now.

### Retire

Decommission end-of-life application.

### Rehost (lift and shift)

Repackage and move existing applications with as few changes as possible.

### Replatform

Change the underlying platform (runtime, framework, middleware, OS).

### Refactor/Rewrite

Redesign code to take advantage of the new platform (extend, strangle, rewrite).

### Repurchase / Replace

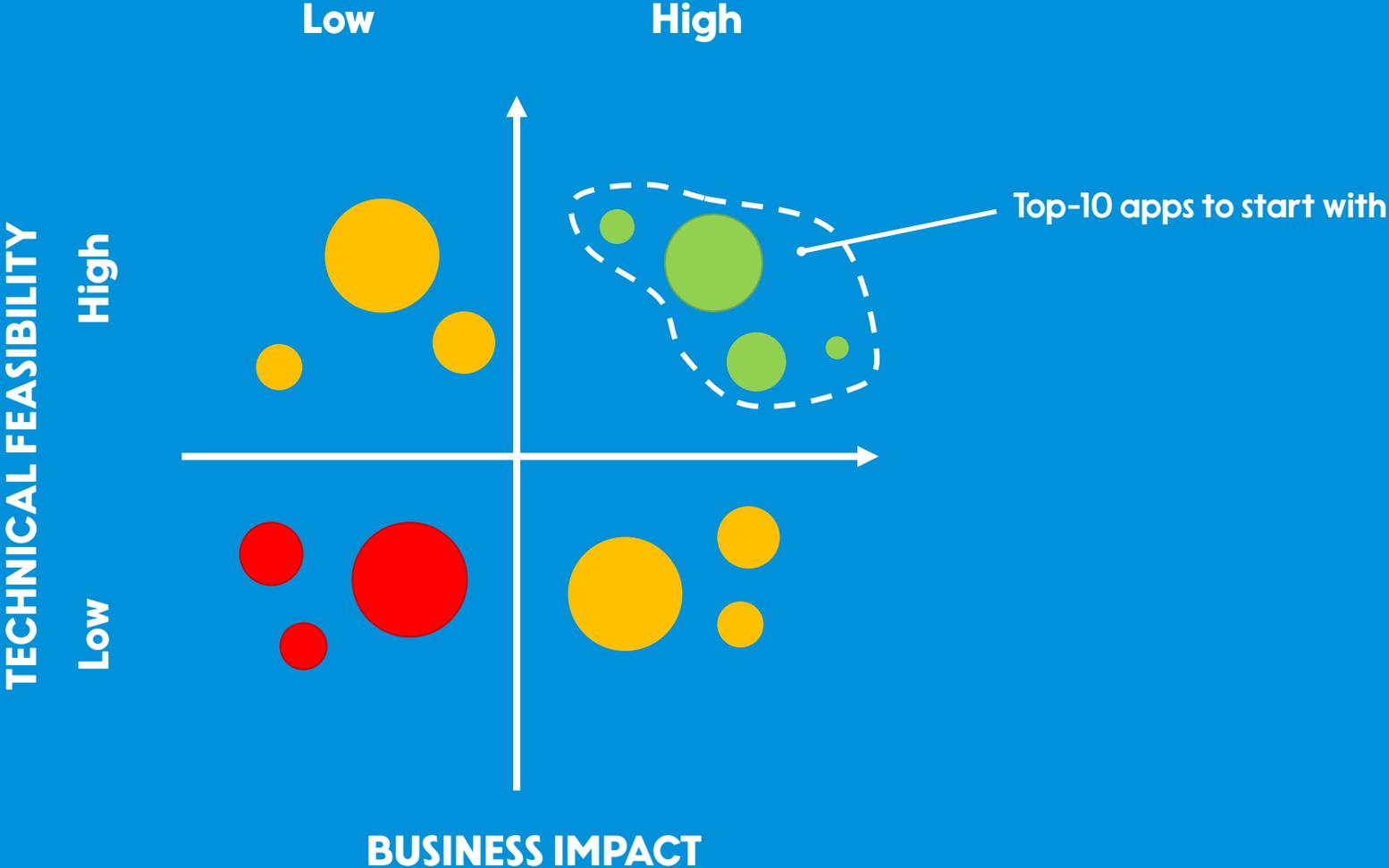
Replace by commercial off-the-shelf (COTS) or Software-as-a-Service (SaaS).

Low efforts  
Low value



High efforts  
High value

# Create Your Portfolio Modernization Framework



# Create Your Portfolio Modernization Framework

## Technical Considerations

- Framework, runtime
- Architecture design
- Statefulness & data
- Use of proprietary tooling
- Dependencies, integration
- Usage, workload

Technical Feasibility

Business Impact

## Business Considerations

- Operational and license costs
- Time-to-market factors
- Revenue opportunities
- Business criticality
- Risk tolerance
- Change frequency

## Decisioning Model

A framework for disposition planning, prioritization and governance

## Organizational and People Factors

- Domain expert availability
- Org structure, maturity
- Lifecycle stage
- Calendar dependencies



RETAIN



RETIRE



REHOST



REPLatform



REFACTOR



REWRITE



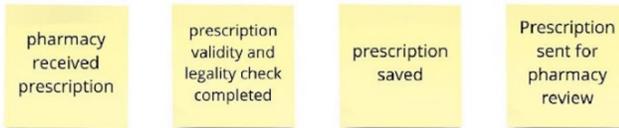
REPLACE

A top-down view of a large assortment of pharmaceutical pills and capsules. The pills are in various shapes, including round tablets with a score line and elongated capsules. The colors are diverse, including shades of blue, green, yellow, orange, red, pink, purple, and white. Some capsules are filled with small, multi-colored beads or granules. The background is a plain, light-colored surface.

**Decompose capability, not code**

# Capture business events with Event Storming

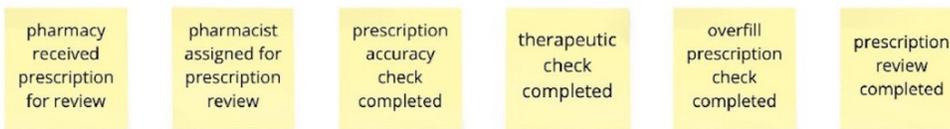
## **prescription ingestion**



## **prescription technical review**



## **prescription safety review**



## **prescription preparation**



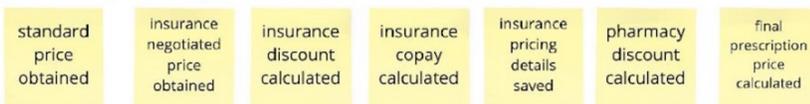
## **prescription authenticity check**



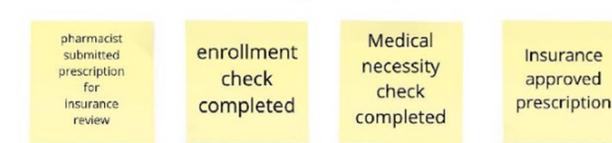
## **prescription dispensing**



## **prescription price calculation**



## **insurance coverage verification**



# Identify business domains, find shared services, design hygiene

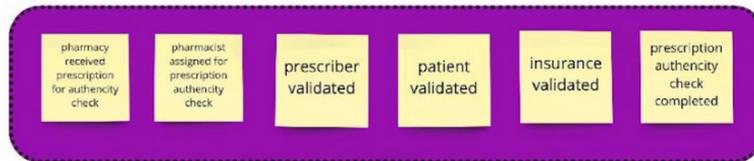
## prescription ingestion



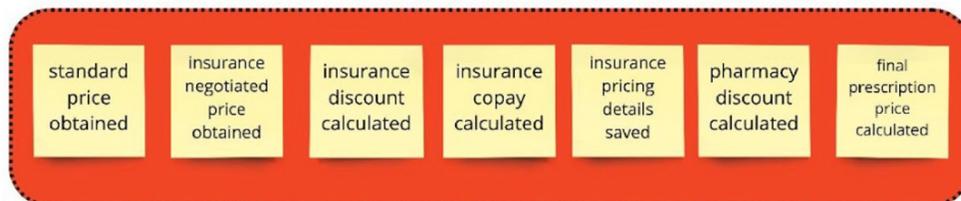
## prescription safety review



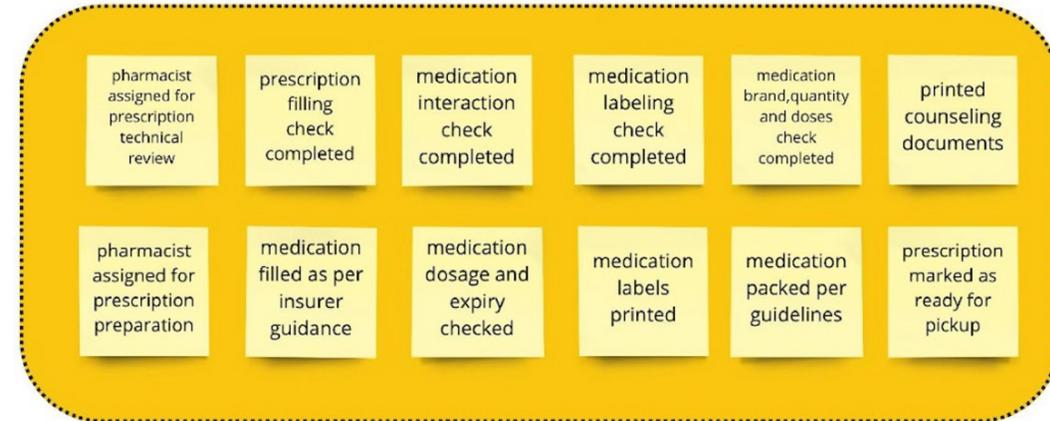
## prescription authenticity



## prescription pricing



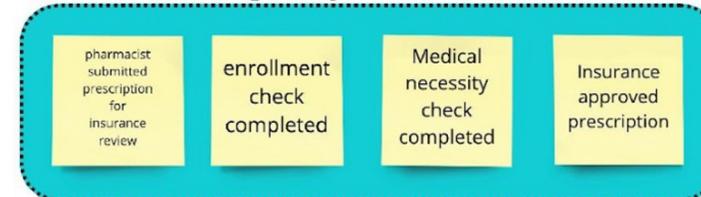
## prescription filled by pharmacy



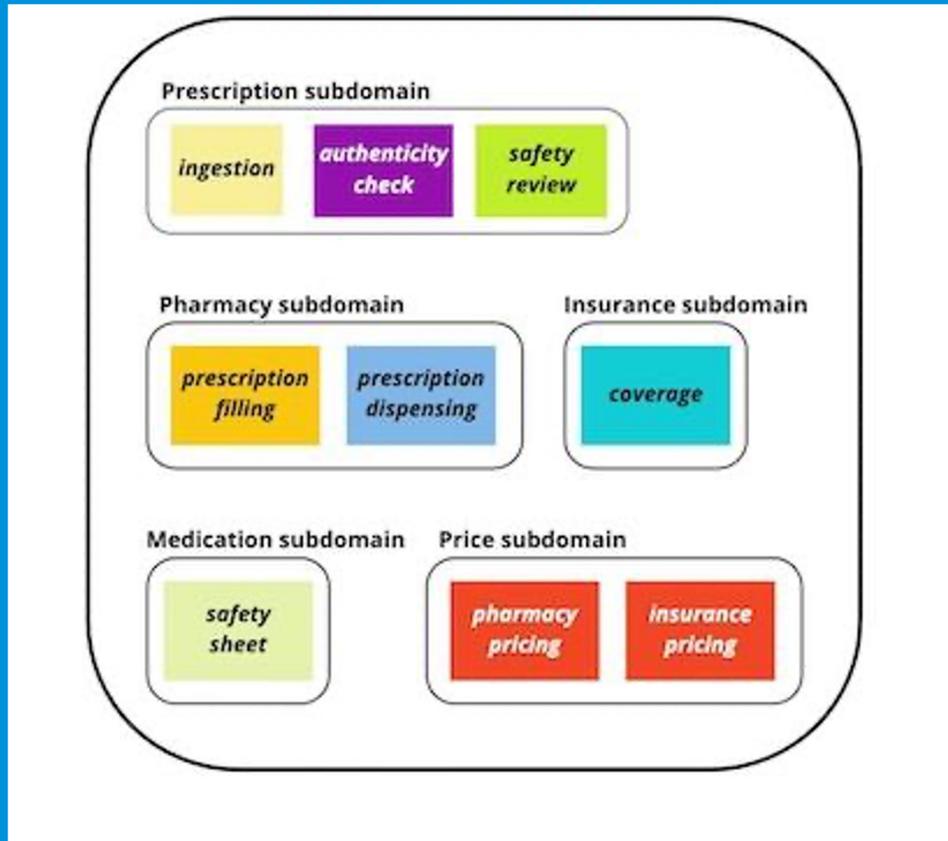
## prescription dispensed by pharmacy



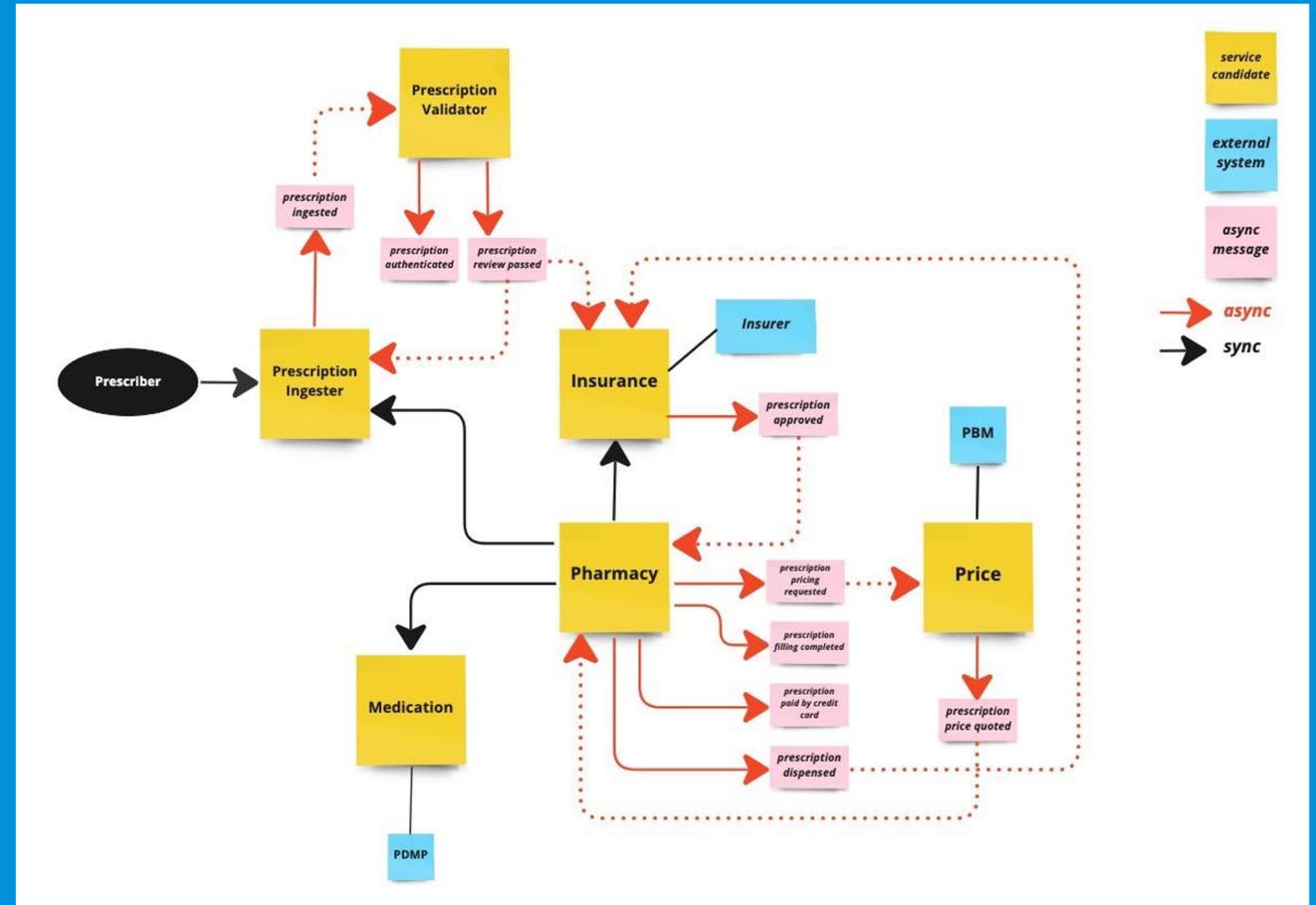
## insurance coverage verification



# Pick an important but small slice, then architect it for scale

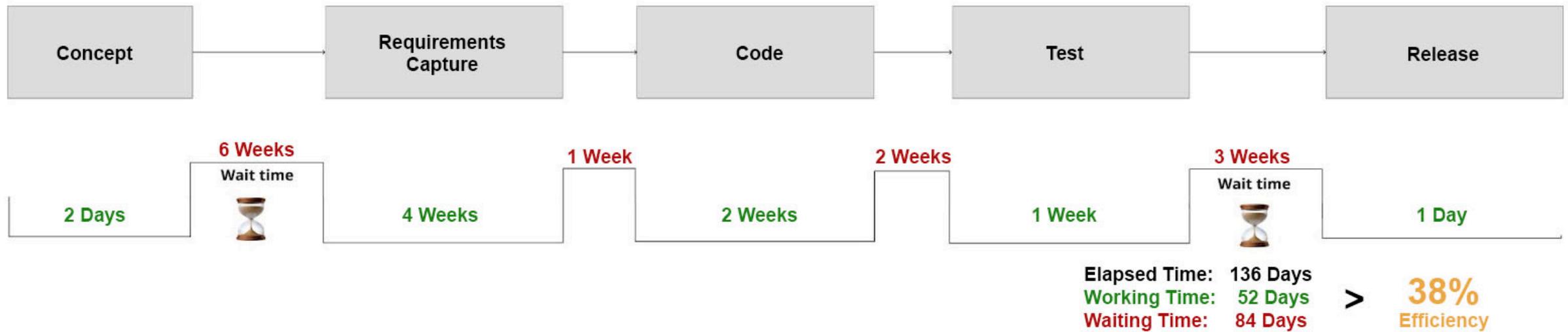


Thin Slice: Prescription fulfillment



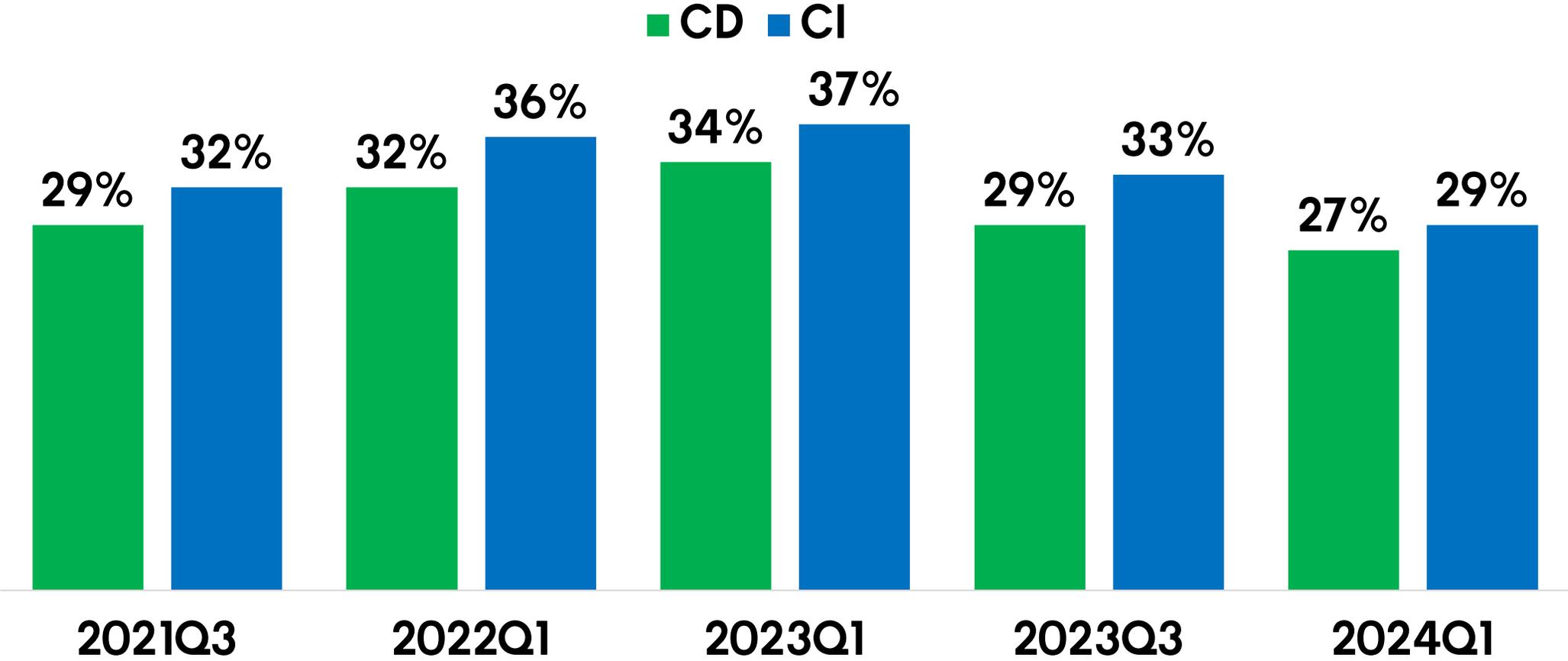
Notional Architecture: Boris exercise

# CI/CD is the foundation



# Legacy tools

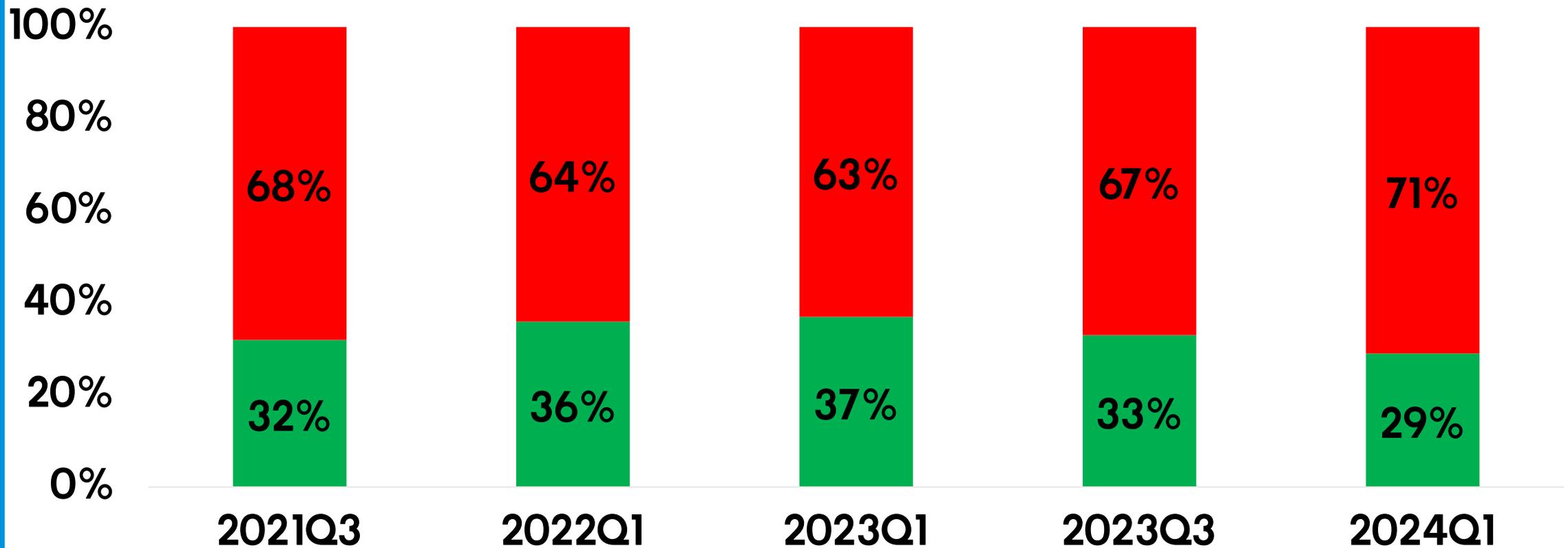
## CI and CD Usage, 2021 to 2024



Question: Which of the following technologies have you used as part of your development activities in the last 12 months? Source: CD Foundation Surveys (Slashdata).

## CI Usage, 2021 to 2024

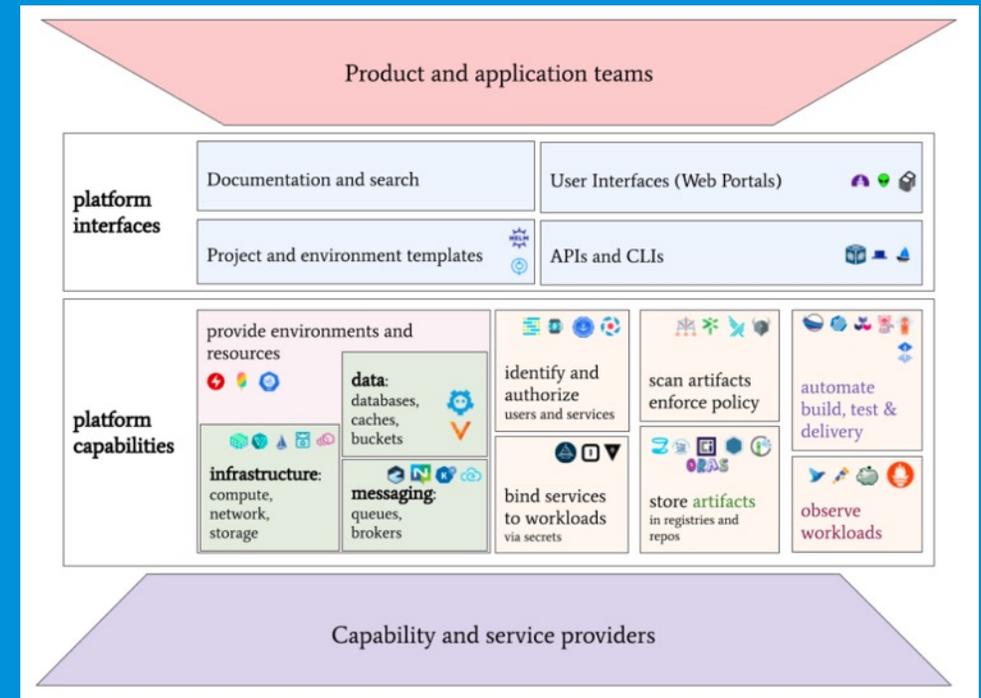
■ CI ■ No CI



Question: Which of the following technologies have you used as part of your development activities in the last 12 months? Source: CD Foundation Surveys (Slashdata).

# Accidental platforms

Kubernetes use in Spring environments continued to grow this year, reaching 65% of respondents. More than half (52%) run a *Kubernetes distribution* (DIY, TKG, Rancher, EKS, etc.), a third (33%) use a *platform based on Kubernetes* (OpenShift, TAP, etc.), and more than a quarter (26%) use a *non-Kubernetes based platform* (CloudFoundry, Heroku, etc.). We find the fact that half start with a *Kubernetes distribution* rather than a more complete platform a little surprising since so much extra work is required.



When this guy does not care



## Motivating "them" to priorities modernization

1. Use a crisis: need features, pricing hikes, no more support, no one knows COBOL.
2. Fit into annual plans: cost/performance improvements (e.g., Spring), migrate to cloud mania
3. Trick metrics: libyears, DevX, dependency, etc.
4. Relax.

# Thank You

## The Legacy Trap

How to modernize applications that are holding you back, and why you need to start right now

By Michael Coté and Marc Zottner

White Paper: August 2022

## Developer Toil: The Hidden Tech Debt

How to use the Developer Toil Audit to speed up software release cycles, make developers more productive, and increase the business value of your apps.

By Susie Forbath, Tyson McNulty, and Michael Coté

vmware

Get it!



-  <https://newsletter.cote.io/>
-  <https://cote.io/legacytrap/>
-  <https://via.vmw.com/legacy-trap>
-  <https://via.vmware.com/devtoil>