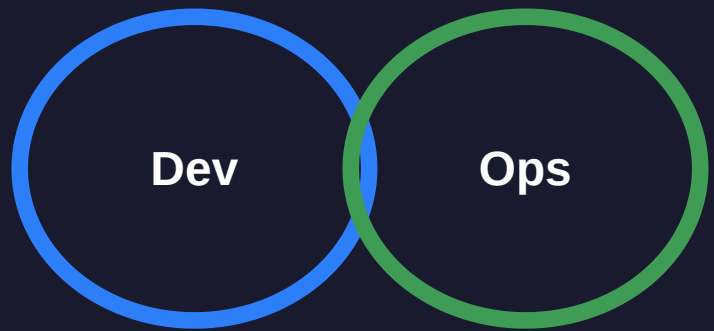


DevOps

From silos to flow — a history of problems and the tools that solved them

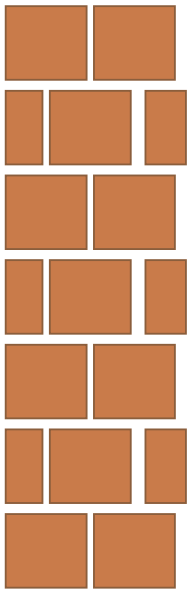
Culture • Architecture • Tooling — 2000 → 2026

The 2000s IT organization

DEV

writes code,
ships features

Java, C, C++ Shell, SQL ... ETC



throw over the wall

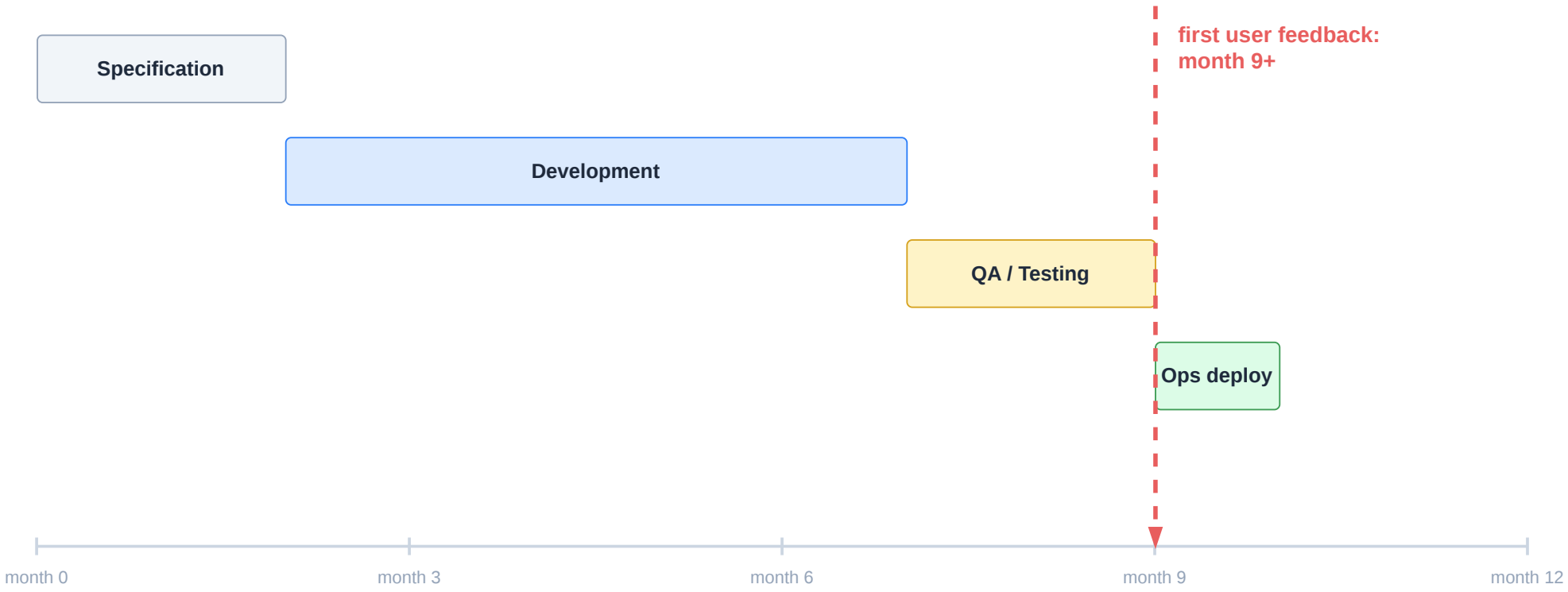
OPS

runs servers.
deploy apps.
keep it up

Privision Servers (HP, DELL ..) , Install OS (Linux / Windows) , Configure Network , Intall dependencis

Two teams, two goals, one wall.

Waterfall delivery



Nine months before a single user touches the feature.

Agile Manifesto for Software Development

إعلان أجايل لتطوير البرمجيات

إننا نكشف النقاب عن طرق أفضل لتطوير البرمجيات
وذلك من خلال تطبيقها ومساعدة الآخرين على ذلك.
وعبر هذا العمل توصلنا إلى اعتبار:

الأفراد وتعاملهم فيما بينهم فوق المنظومات والأدوات

البرمجيات الصالحة للاستعمال فوق التوثيق الكامل

تعاون ومشاركة العميل فوق التفاوض حول العقد

الاستجابة للتغييرات فوق الالتزام بمخطط عمل محدد

ويعني ذلك أنه على الرغم من كون العناصر على الجانب الأيسر ذات قيمة،
فإننا نعطي قيمة أكبر للعناصر على الجانب الأيمن.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

Conflicting incentives

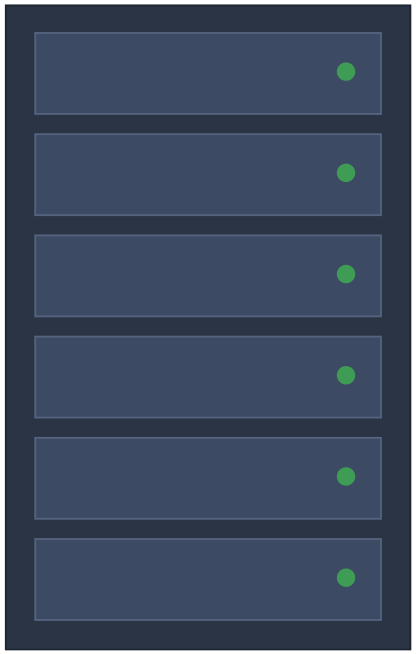


Dev bonus: features delivered

Ops bonus: zero incidents

Speed vs stability — the conflict is by design, not by character.

The physical server era



rack #12

Order hardware (2–4 weeks) → Rack & cable → Install OS by hand → Configure & harden

= 6 weeks to production

Capacity planning by guessing two years ahead.

Release night — the deployment ritual

02:13 AM

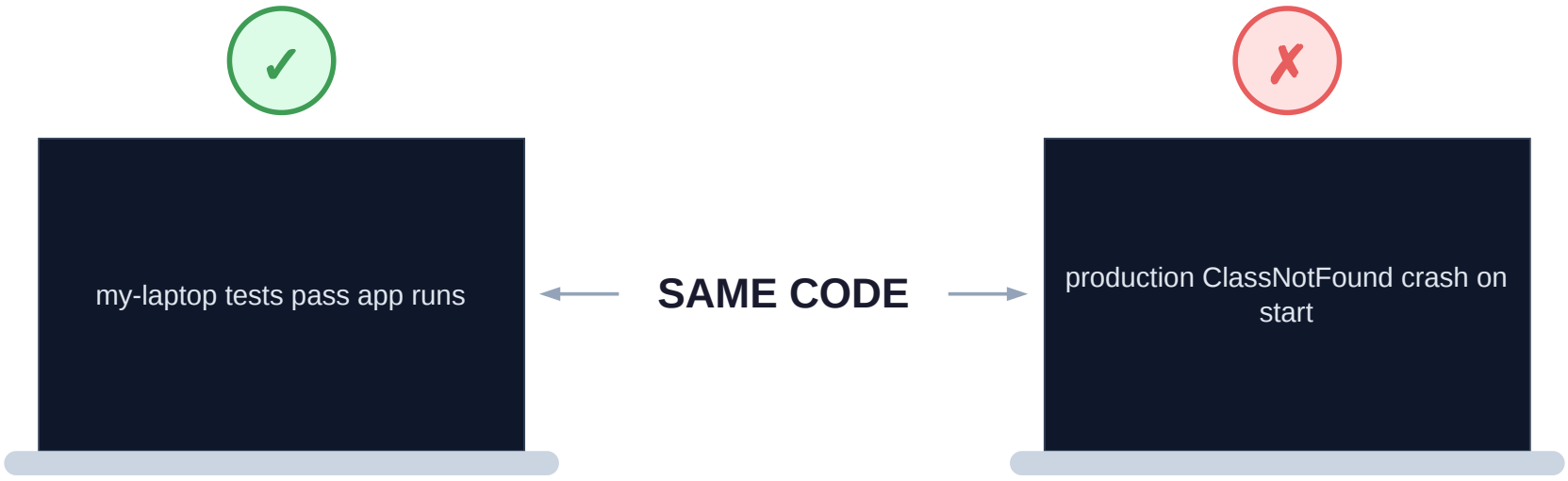
```
> scp app-v2.3.war prod01:/opt/tomcat/ > ssh prod01
"./stop.sh && ./start.sh" > tail -f catalina.out . . .
praying . . .
```

Runbook_v14 _FINAL2.doc 40 pages of manual steps

One release per quarter. Everyone holds their breath.

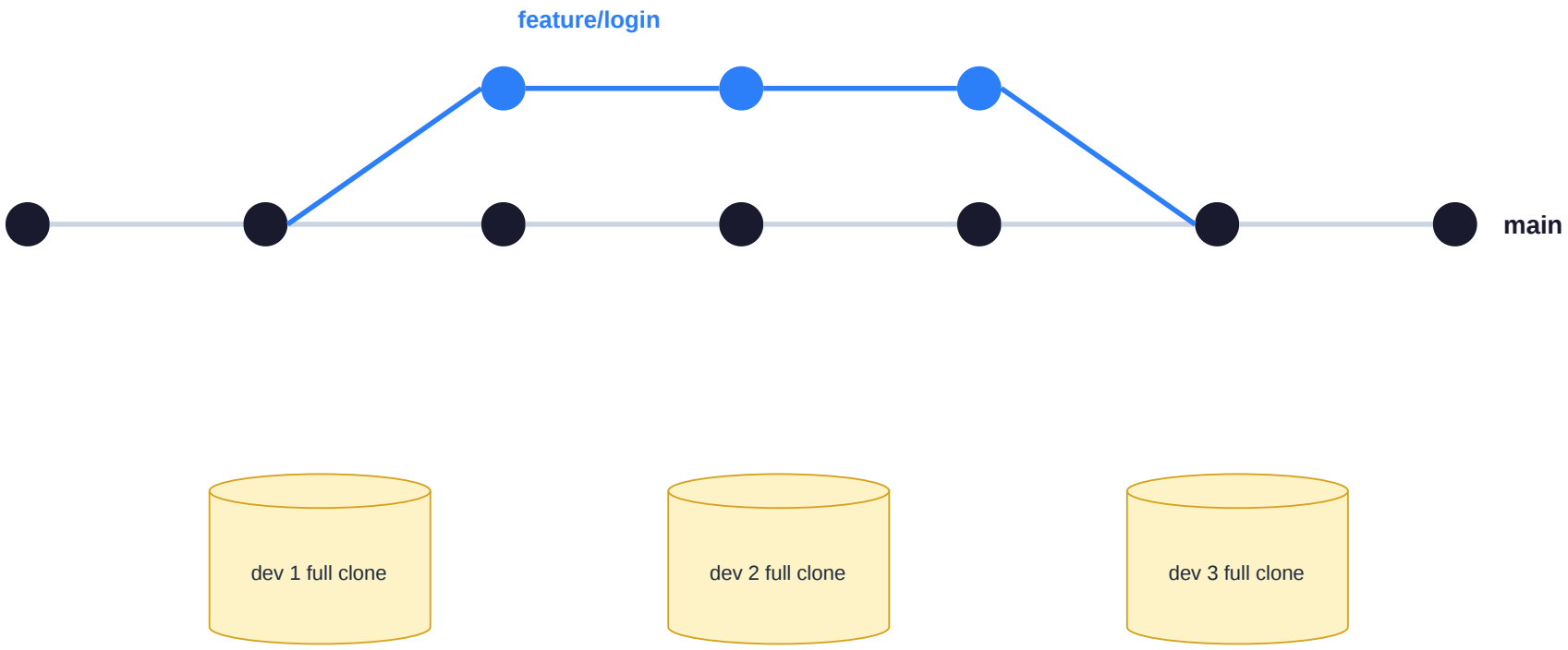
“Works on my machine”

PROBLEM



Different OS, libs, config, JVM version — the environment is invisible and unversioned.

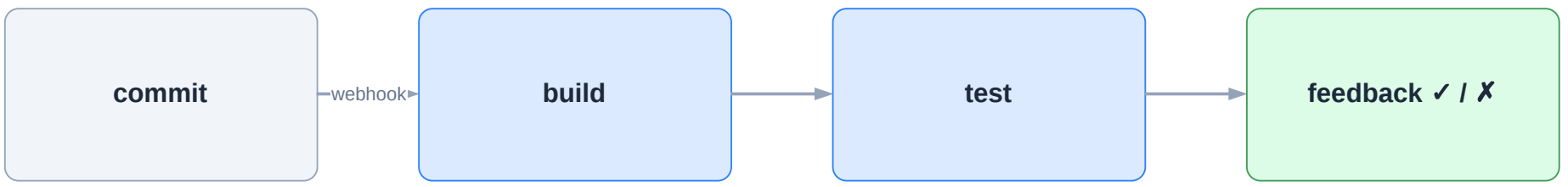
Git (2005) — distributed, branch-cheap



every clone = complete history, offline commits, cheap merges

Branching becomes free — every developer holds the whole repository.

Continuous Integration

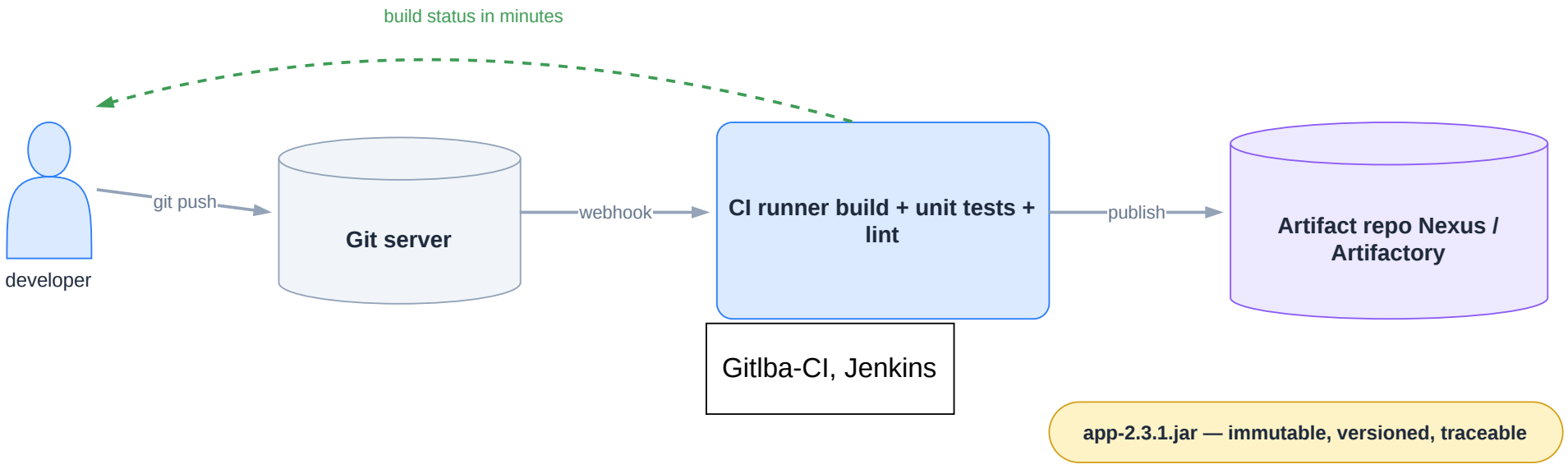


feedback loop: 9 months → 10 minutes

Jenkins | GitLab CI | GitHub Actions

Integrate every commit — find conflicts while they are still small.

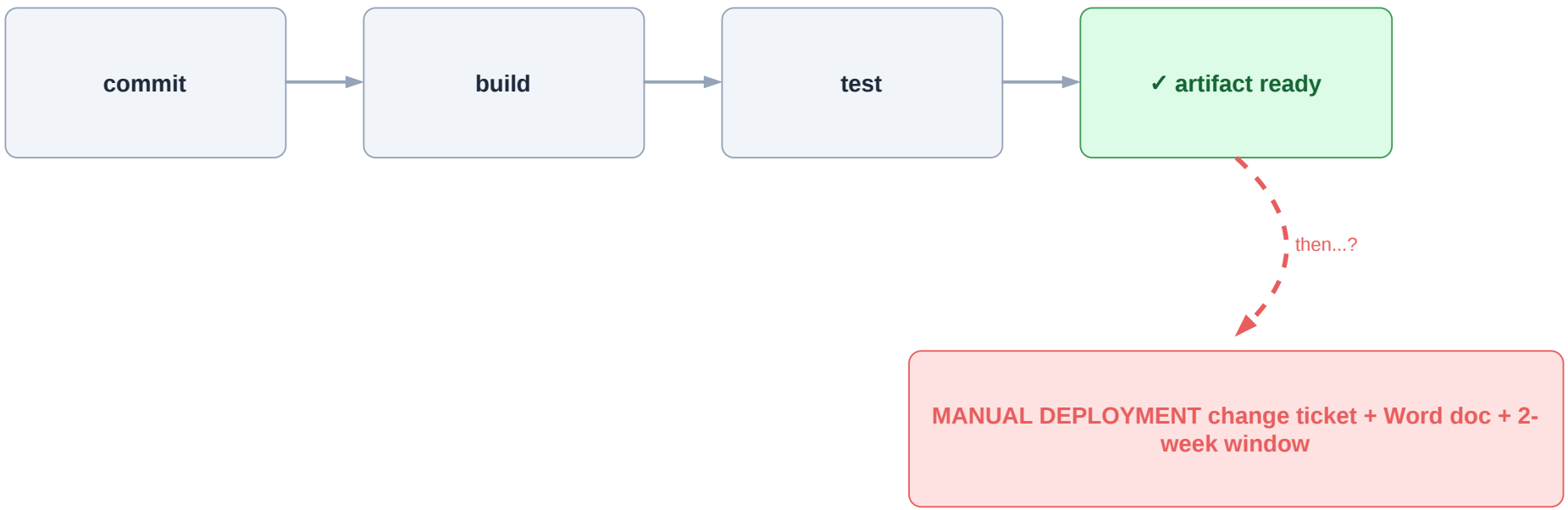
CI architecture — anatomy of a pipeline



The artifact is built once — never rebuilt per environment.

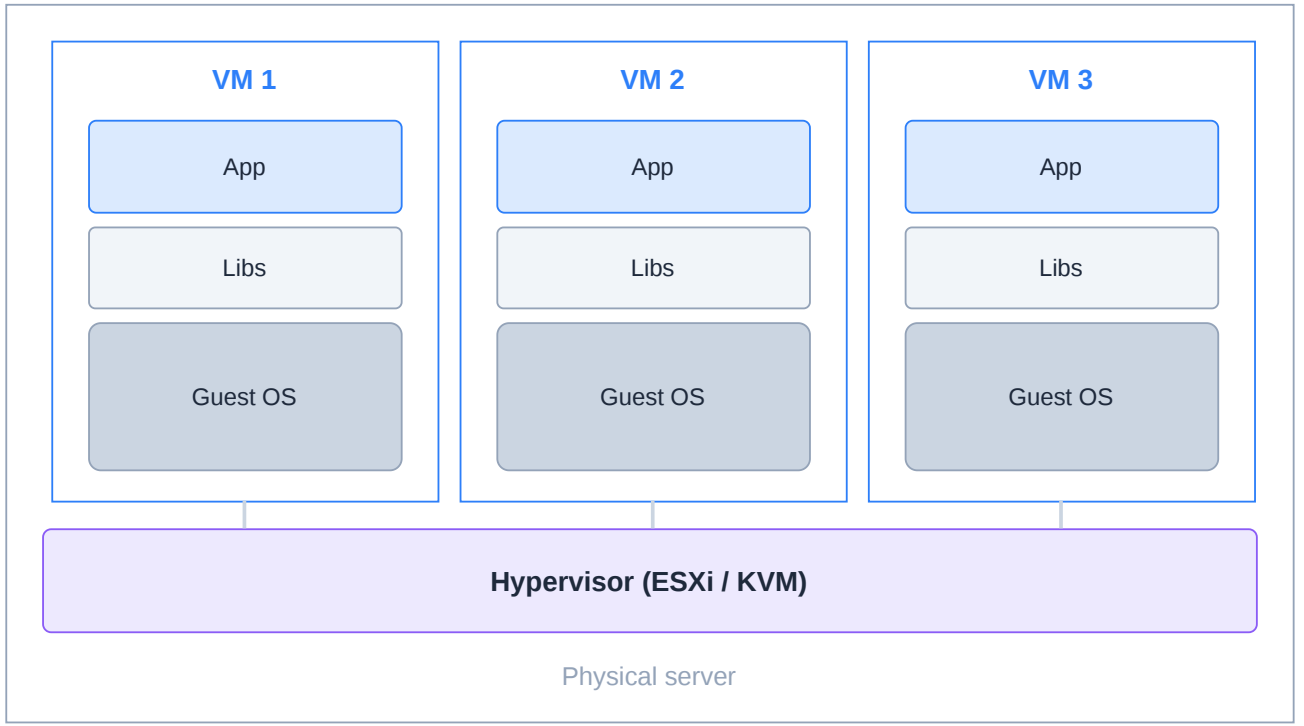
The pipeline stops at green

PROBLEM



Integration is continuous. Delivery is not.

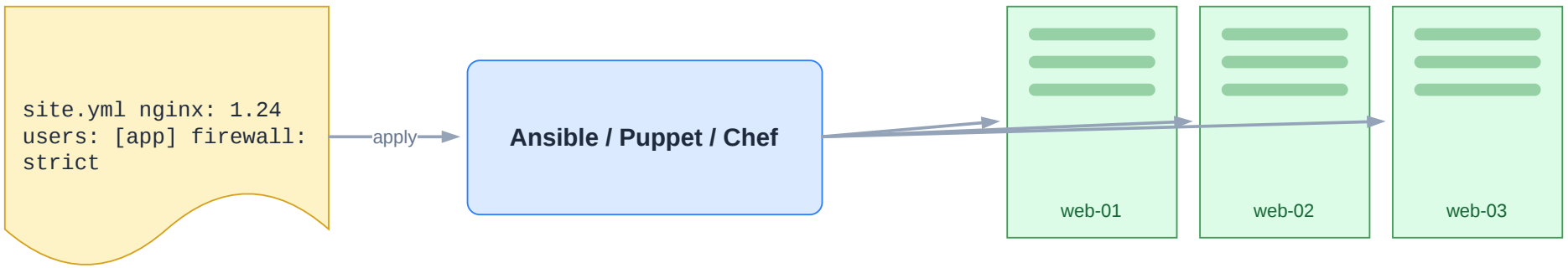
Virtualization — slicing the metal



provisioning: 6 weeks → hours

One physical box, many isolated machines.

Configuration management — declare the state

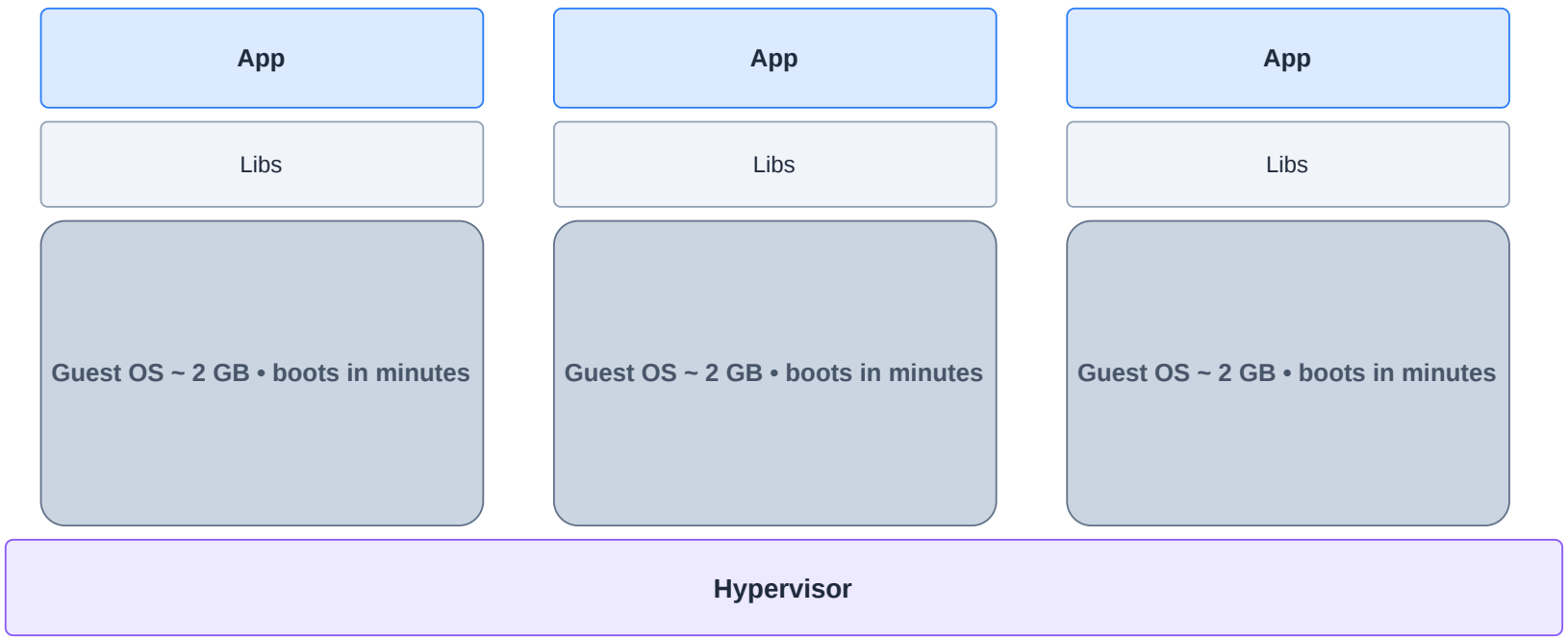


Describe the desired state, not the steps.

Idempotent: run it 100 times, same result.

The weight of VMs

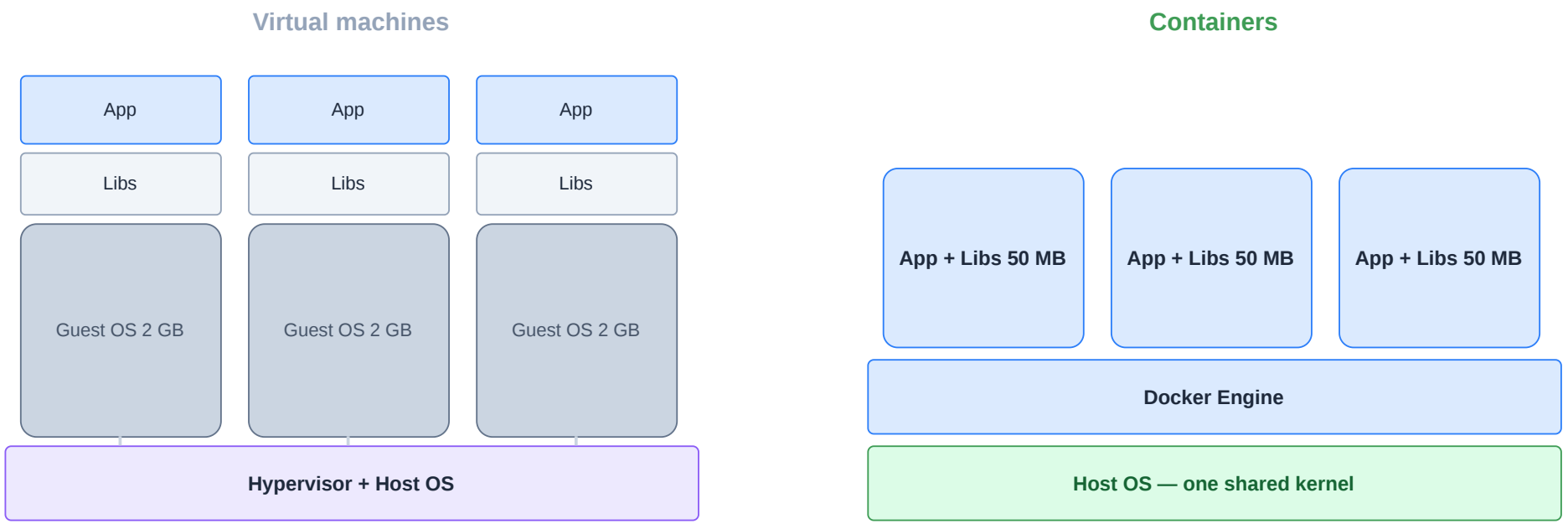
PROBLEM



3 apps → 3 operating systems → duplicated everything

Every app drags a full OS: gigabytes of disk, minutes to boot.

Docker (2013) — containers

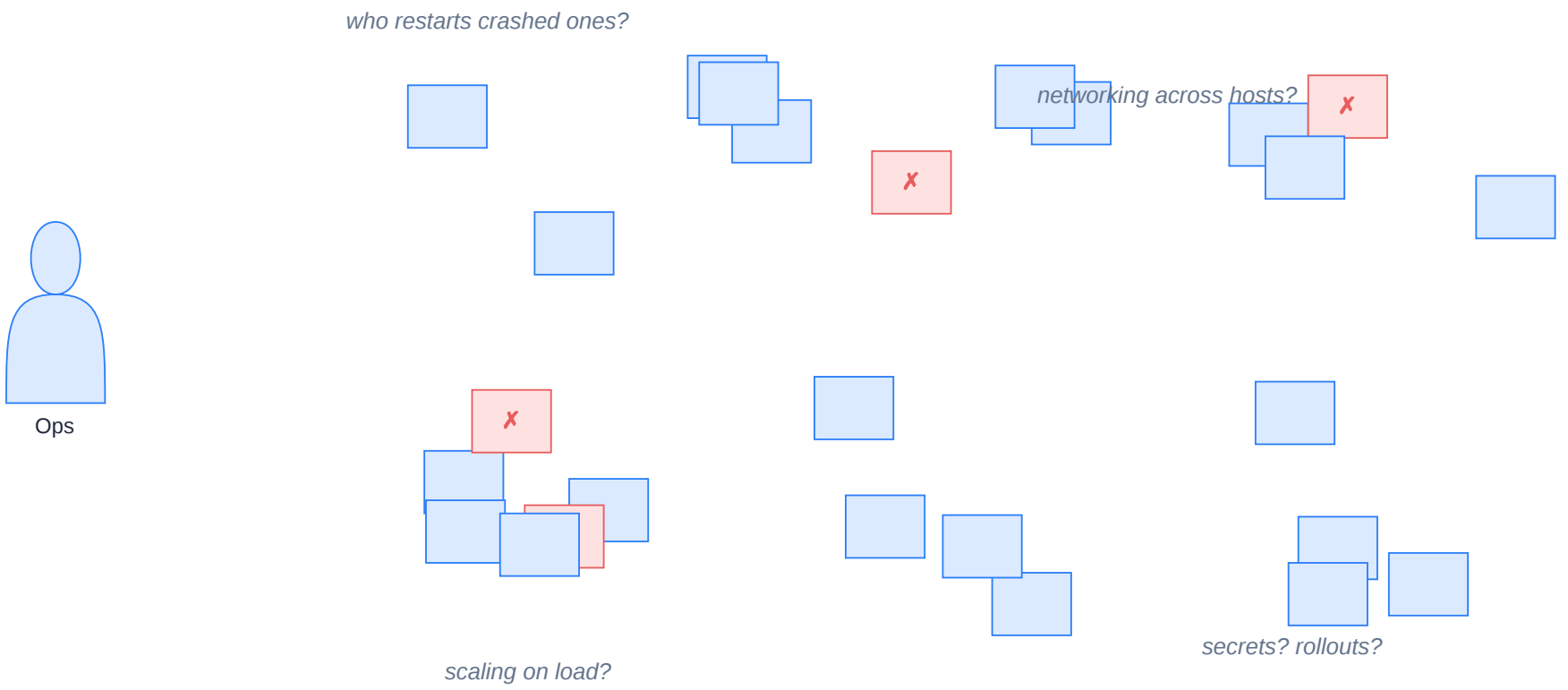


GB → MB | **minutes → seconds** | **same image: dev = test = prod ✓**

Slide 7's problem dies here: the environment ships with the app.

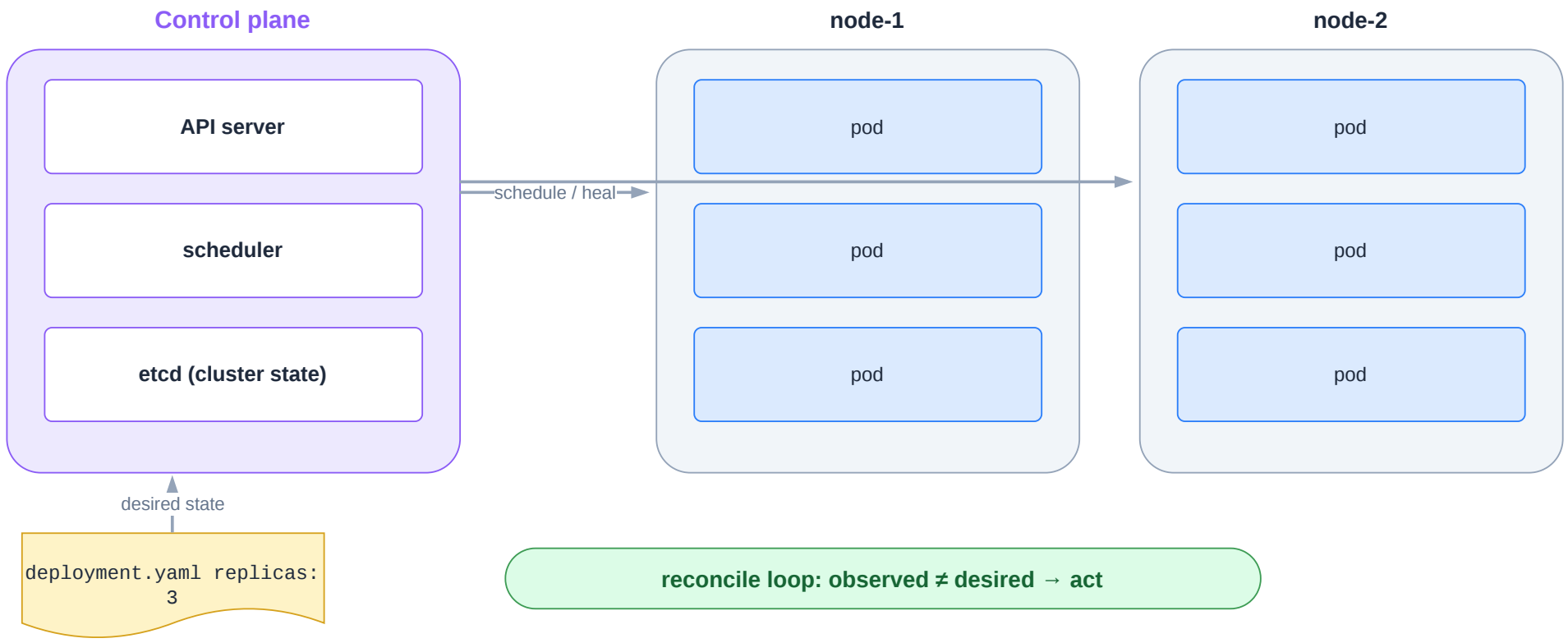
Container sprawl

PROBLEM



200 containers across 20 hosts — humans can't schedule this.

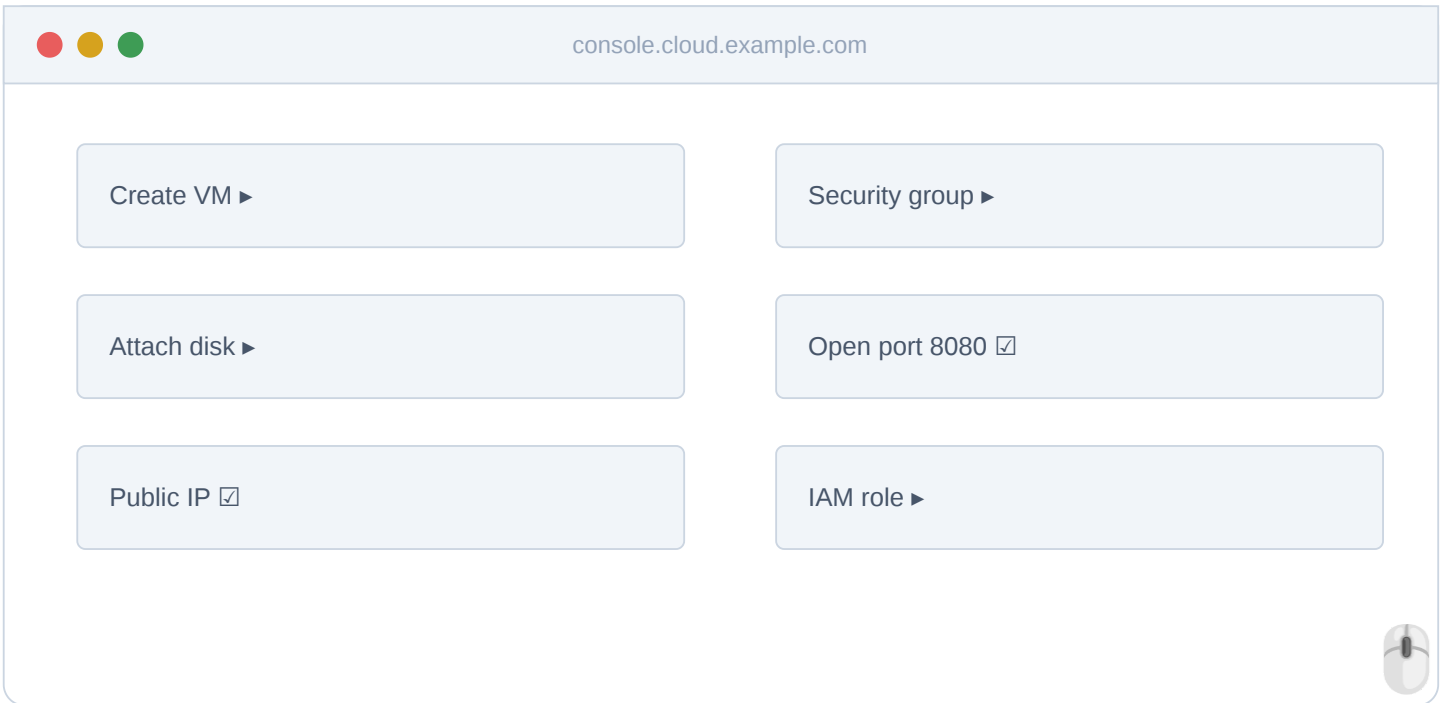
Kubernetes (2014) — desired state



You declare, the cluster converges. Self-healing by design.

ClickOps

PROBLEM



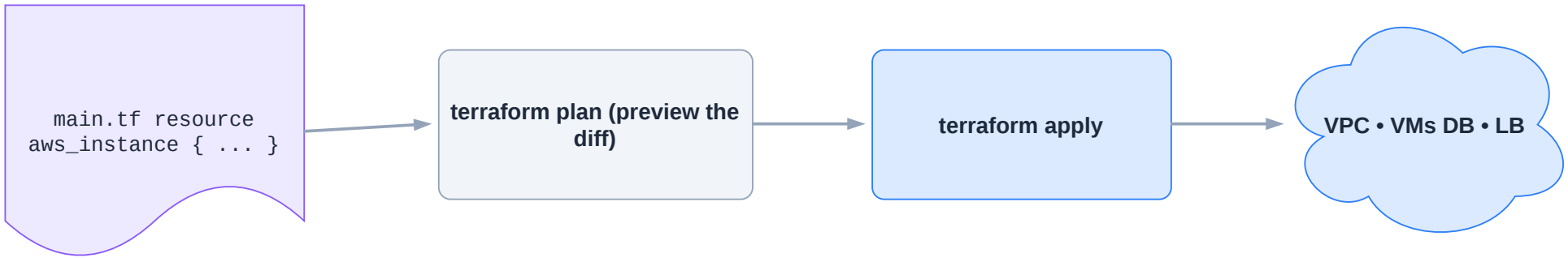
no history

no review

not reproducible

50 clicks in a cloud console — undocumented, unrepeatable, unreviewable.

Infrastructure as Code

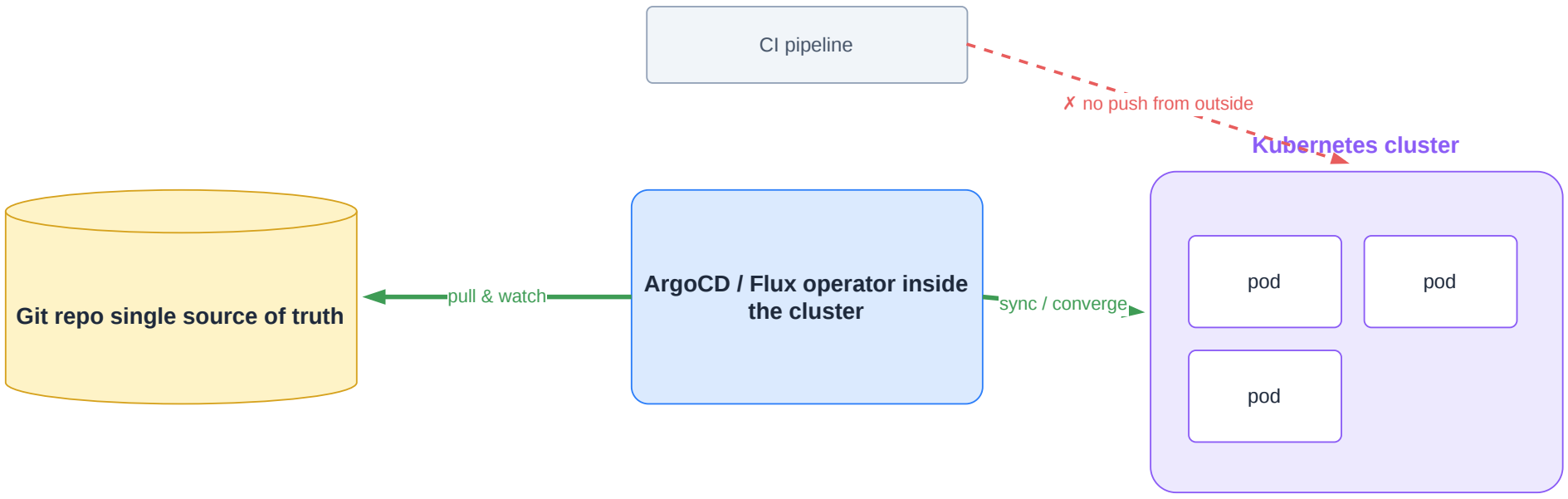



```
- instance_type = "t3.small"
+ instance_type = "t3.large"
```

git diff = infra changelog

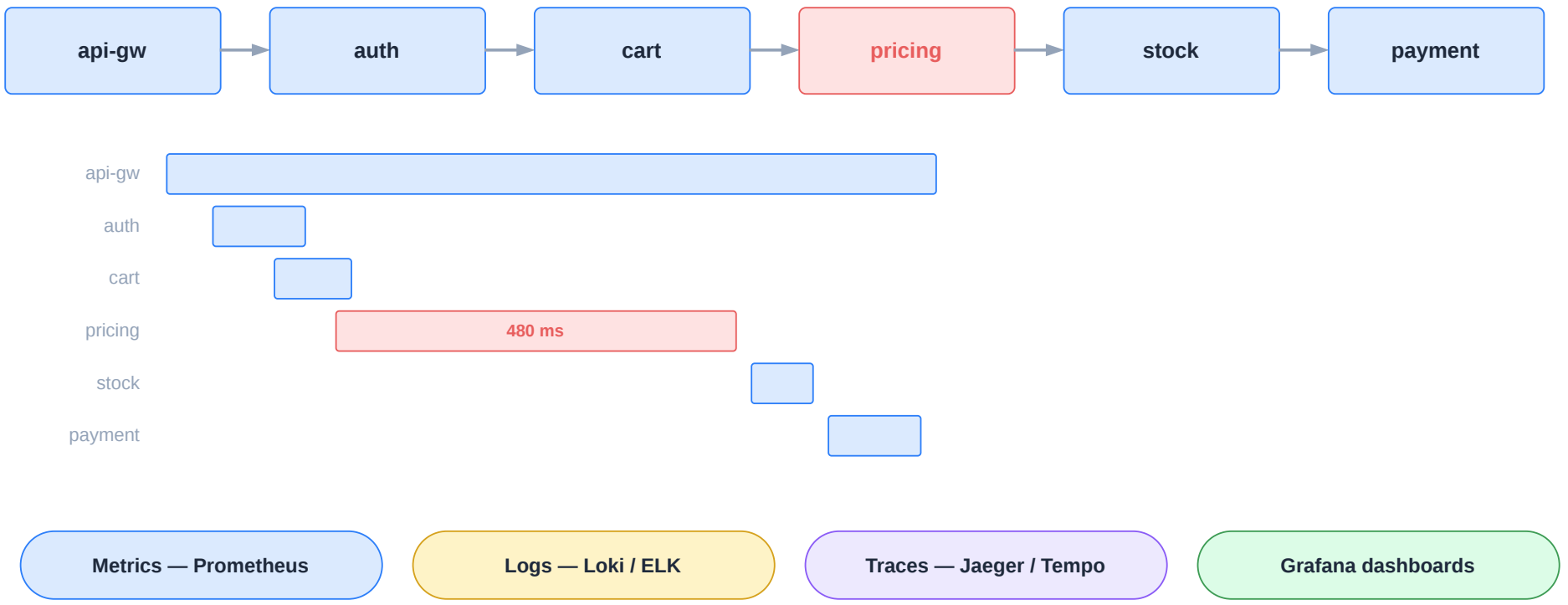
Your infrastructure has a changelog, code reviews, and rollbacks.

GitOps — pull, don't push



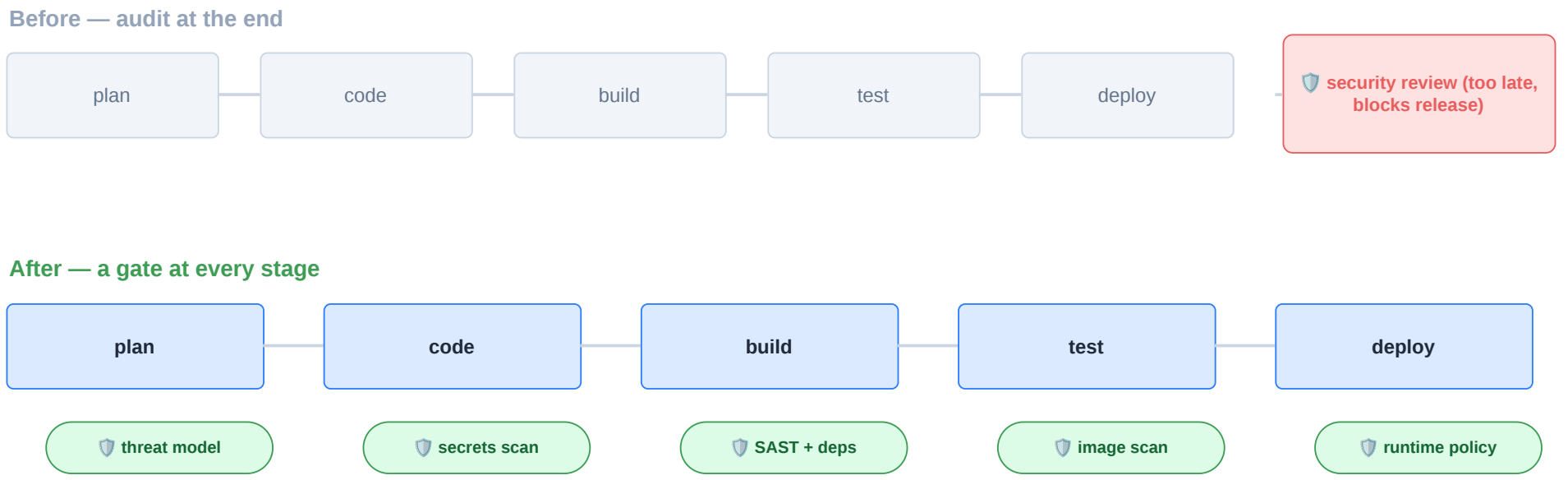
The cluster pulls its own desired state. Drift is auto-corrected.

Observability — one request, six services



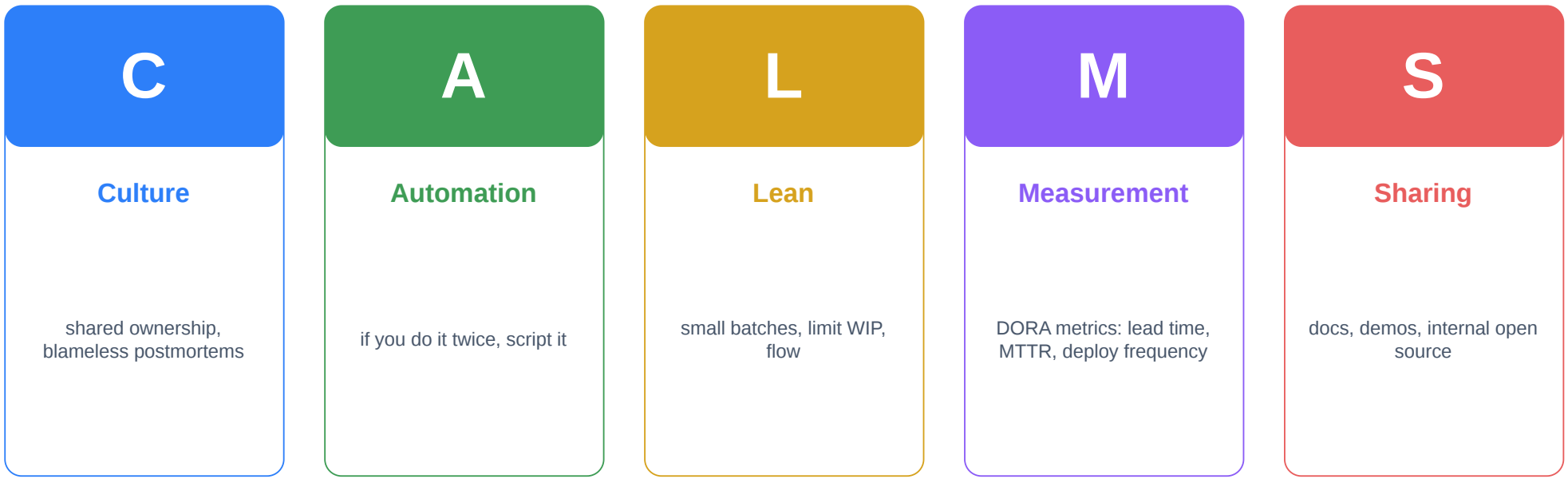
You can't debug a distributed system with SSH and grep.

DevSecOps — shift security left



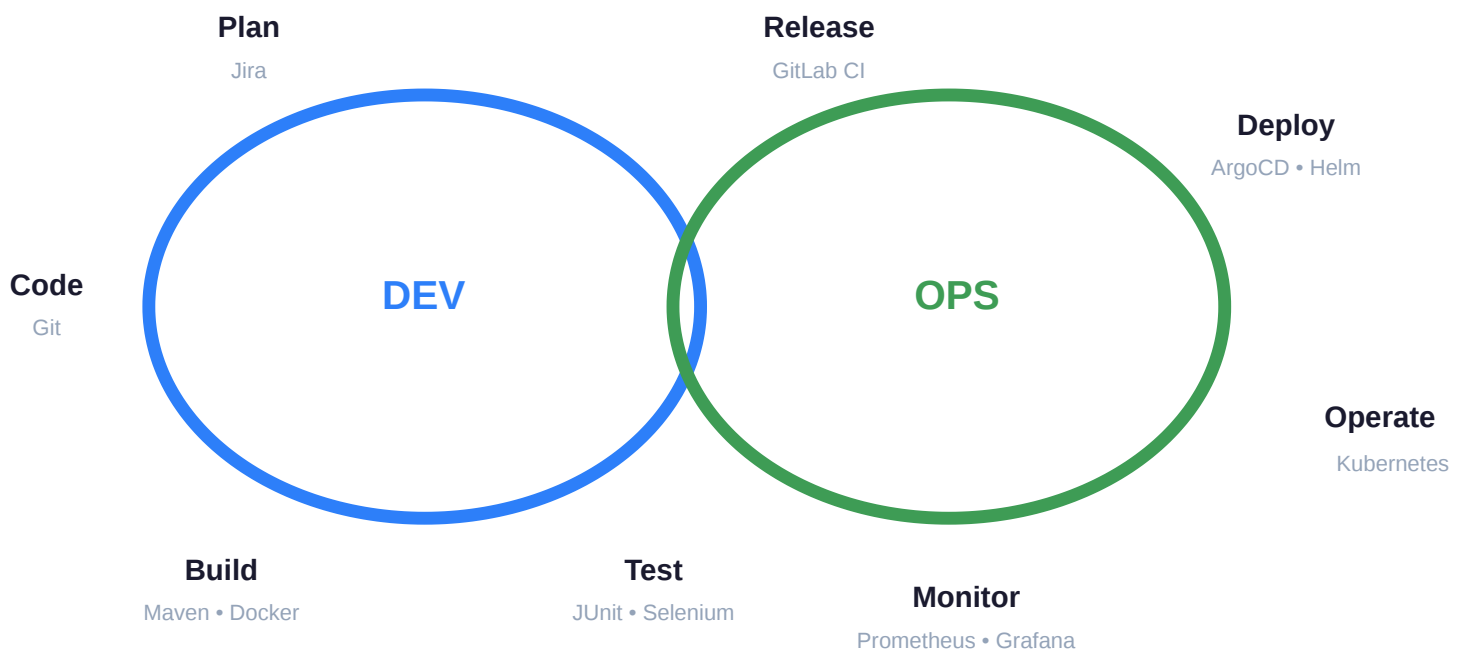
Security becomes a property of the pipeline, not an afterthought.

It was never about the tools — CALMS

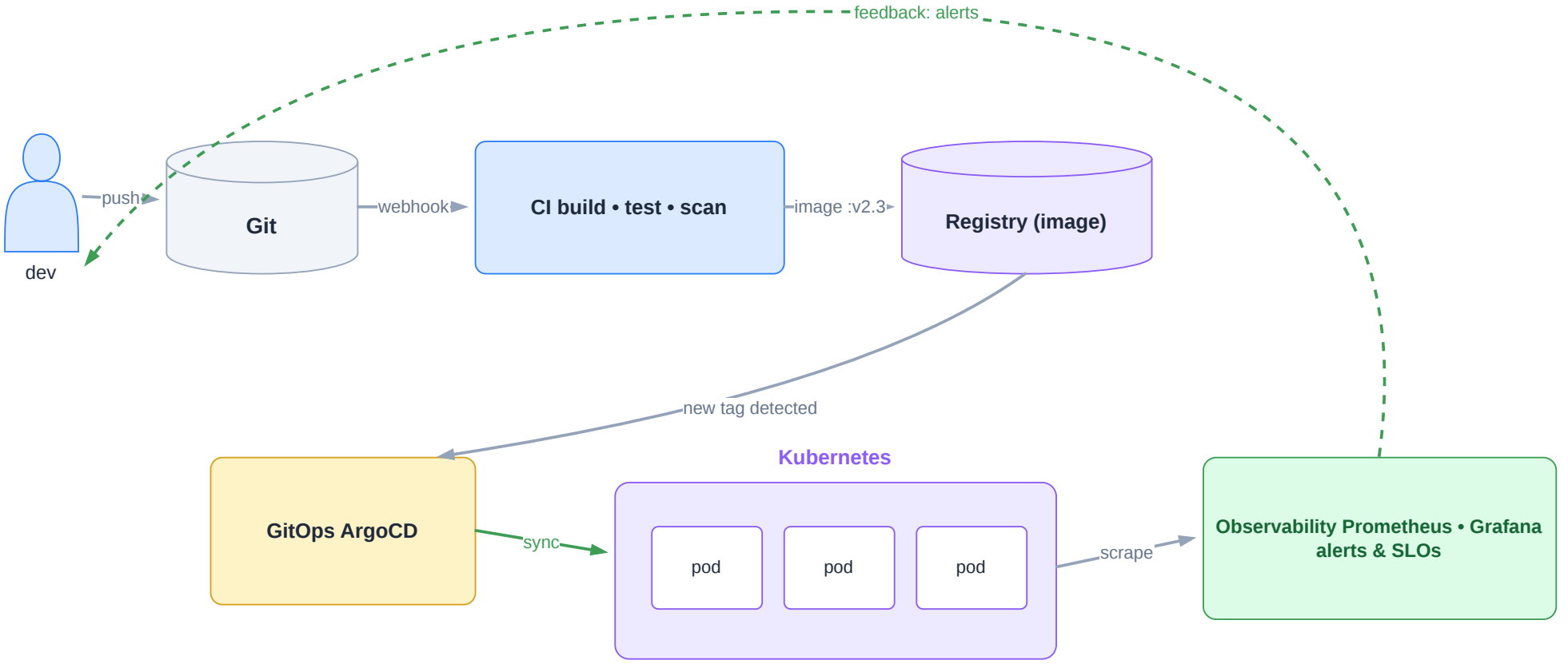


Tools enable the culture. They never replace it.

The infinite loop — tools mapped on each phase



Modern reference architecture



commit → production in minutes, every step audited.