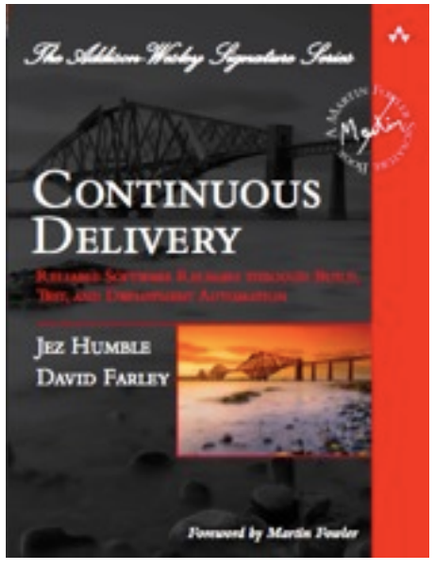
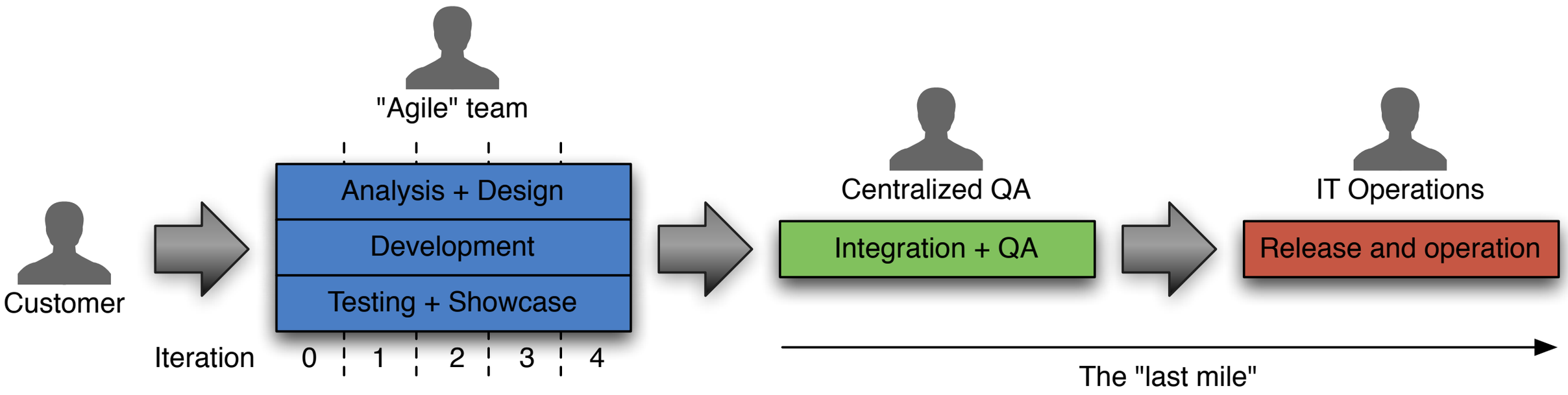
Continuous Delivery



Jez Humble, ThoughtWorks Studios
@jezhumble #continuousdelivery
JAOO 2010, Århus

Agile 101


"Agile" team
Customer
Analysis + Design
Development
Testing + Showcase
Iteration
0
1
2
3
4
Centralized QA
Integration + QA
IT Operations
Release and operation
The "last mile"

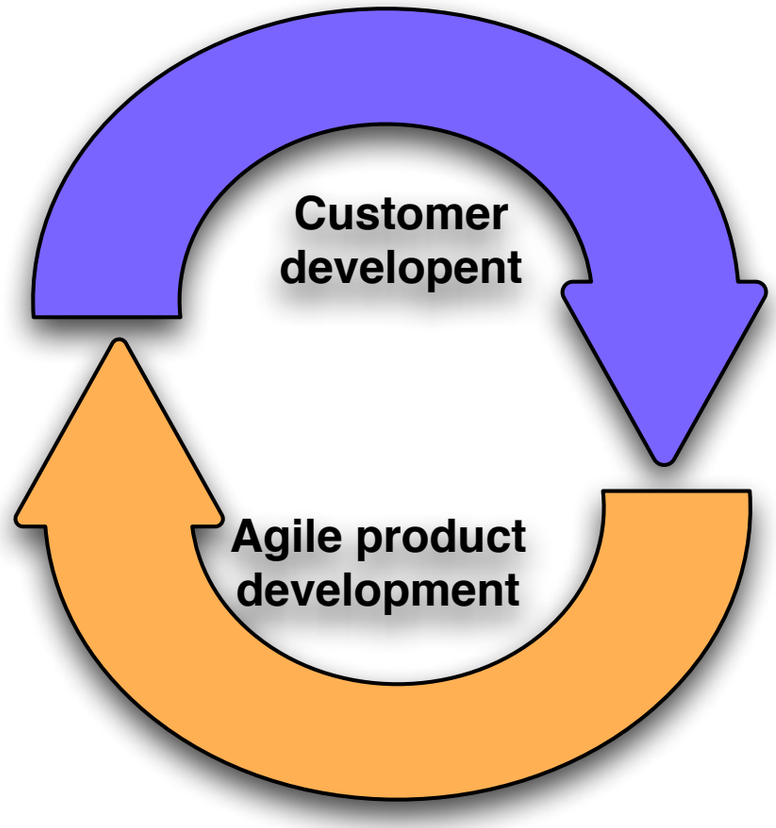
web 2.0

disrupting traditional businesses

http://code.flickr.com/

releasing frequently

feedback from users

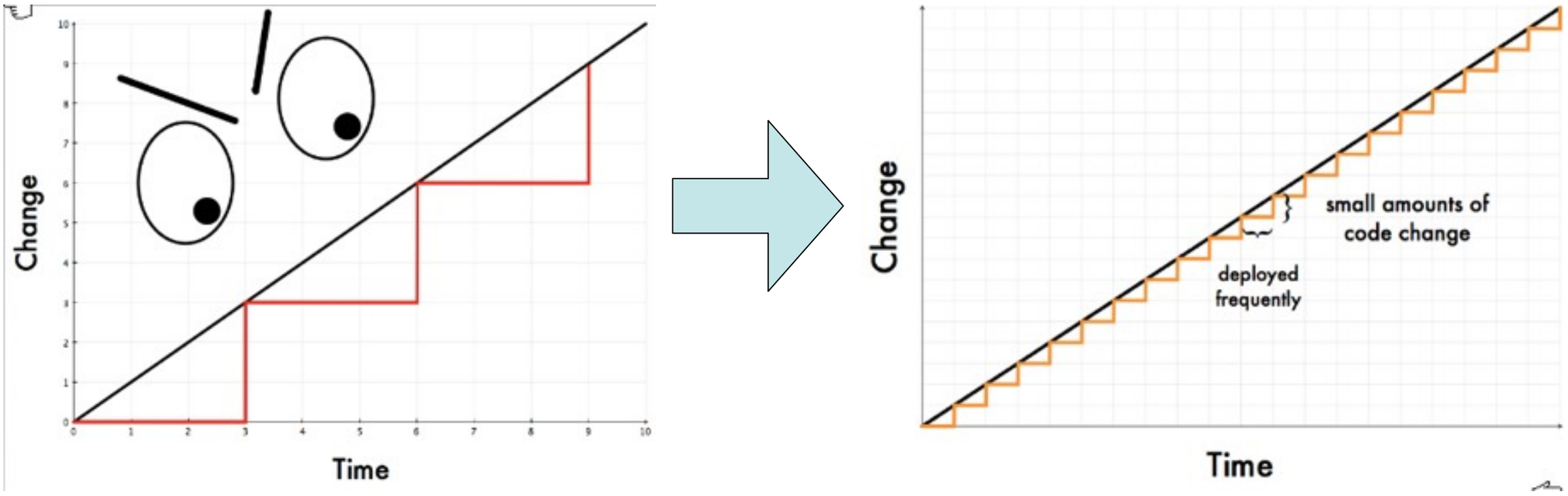


Eric Ries, “The Lean Startup” http://bit.ly/8ZoX5F

releasing frequently

feedback from users

reduce risk of release



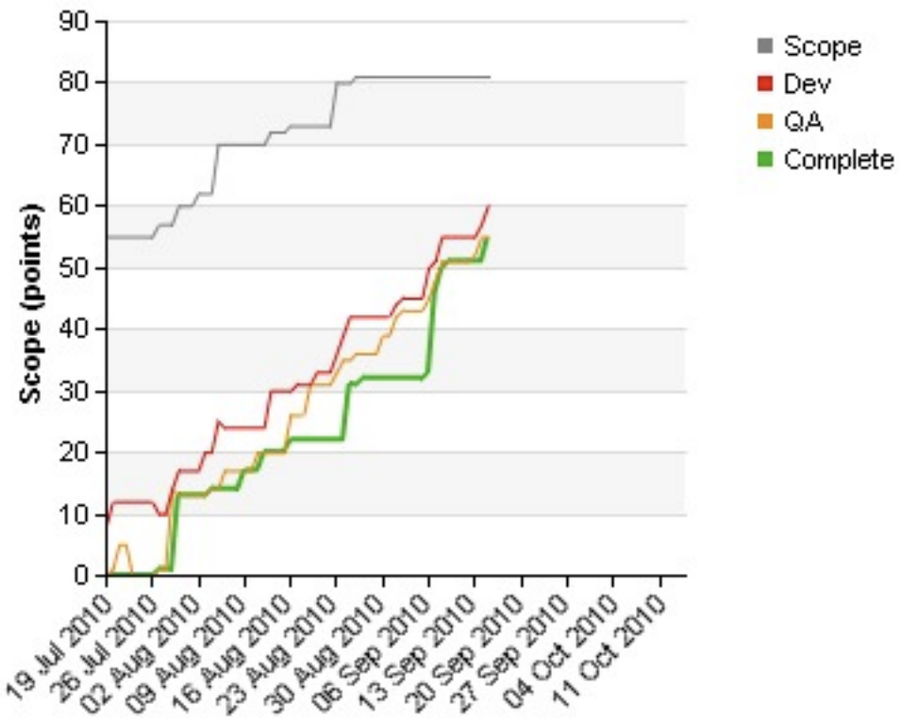
John Allspaw: “Ops Metametrics” http://slidesha.re/dsSZIr

releasing frequently

feedback from users

reduce risk of release

real project progress



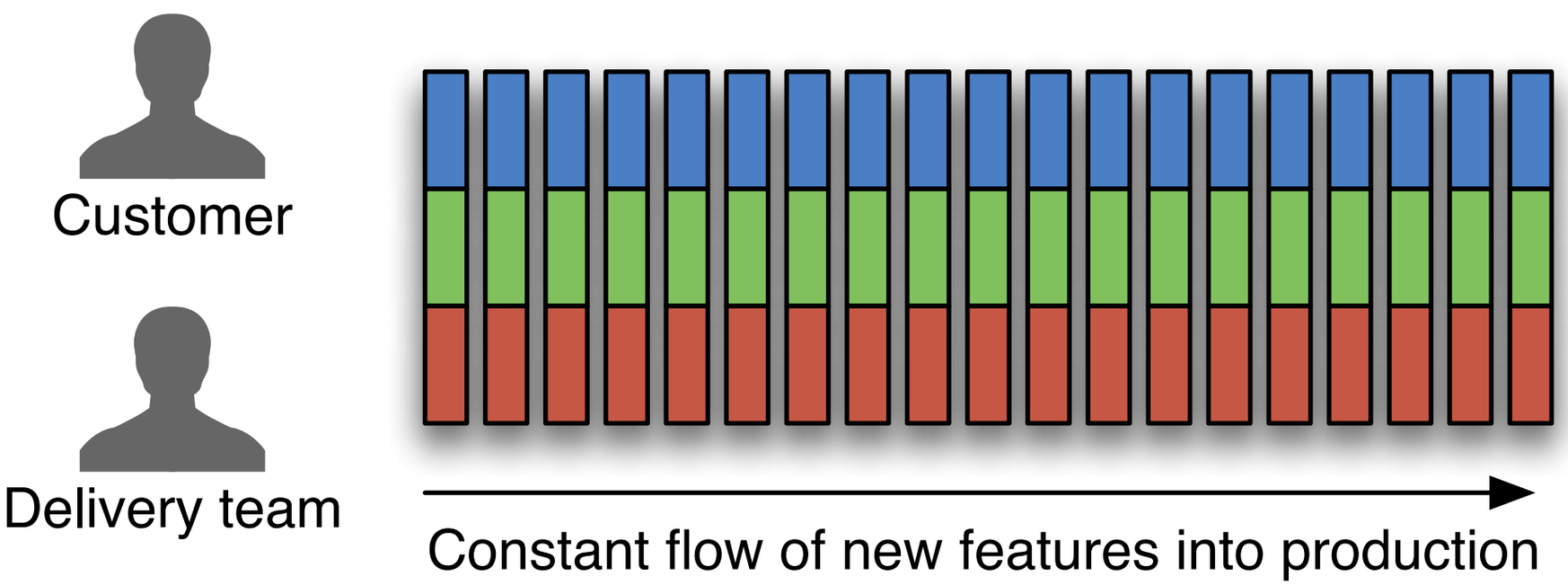
agile manifesto

Our highest priority is to satisfy the customer through early and **continuous delivery** of **valuable software**

production-ready software

Fast, automated feedback on the **production readiness** of your applications **every** time there is a **change** – to code, infrastructure, or configuration

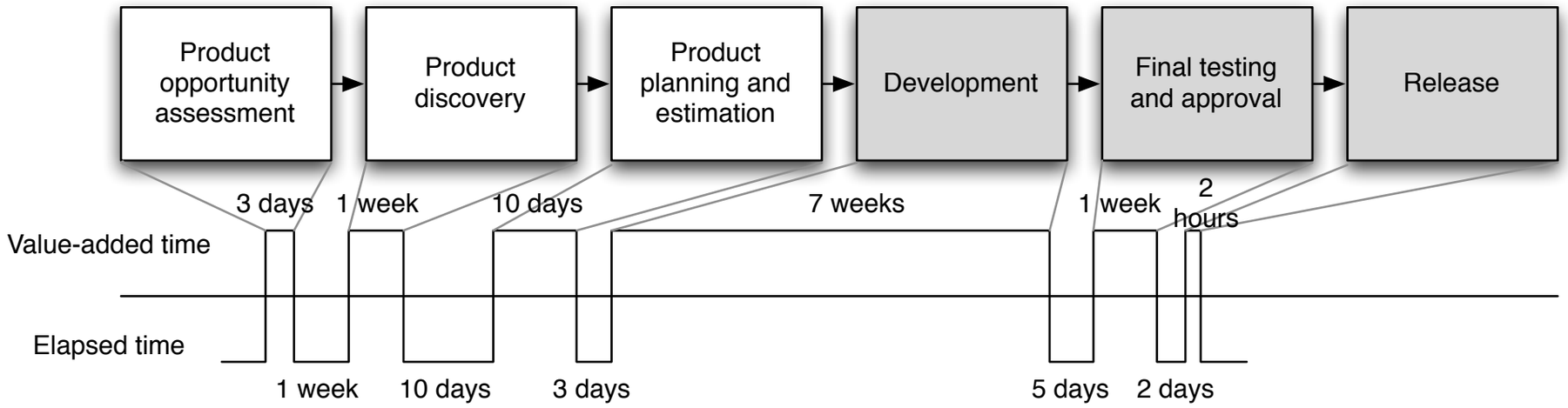
continuous delivery



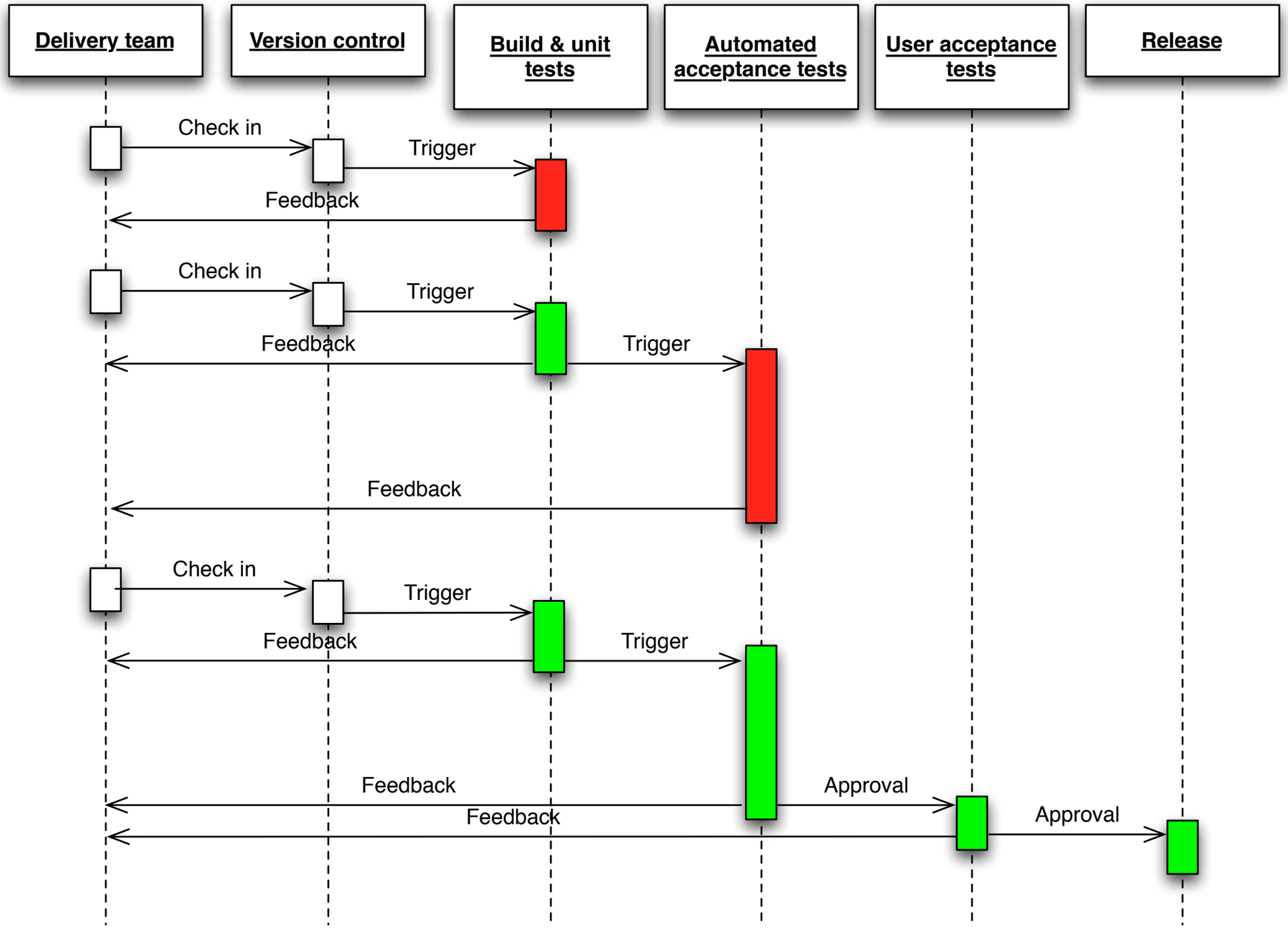
Software always production ready

Releases tied to business needs, not operational constraints

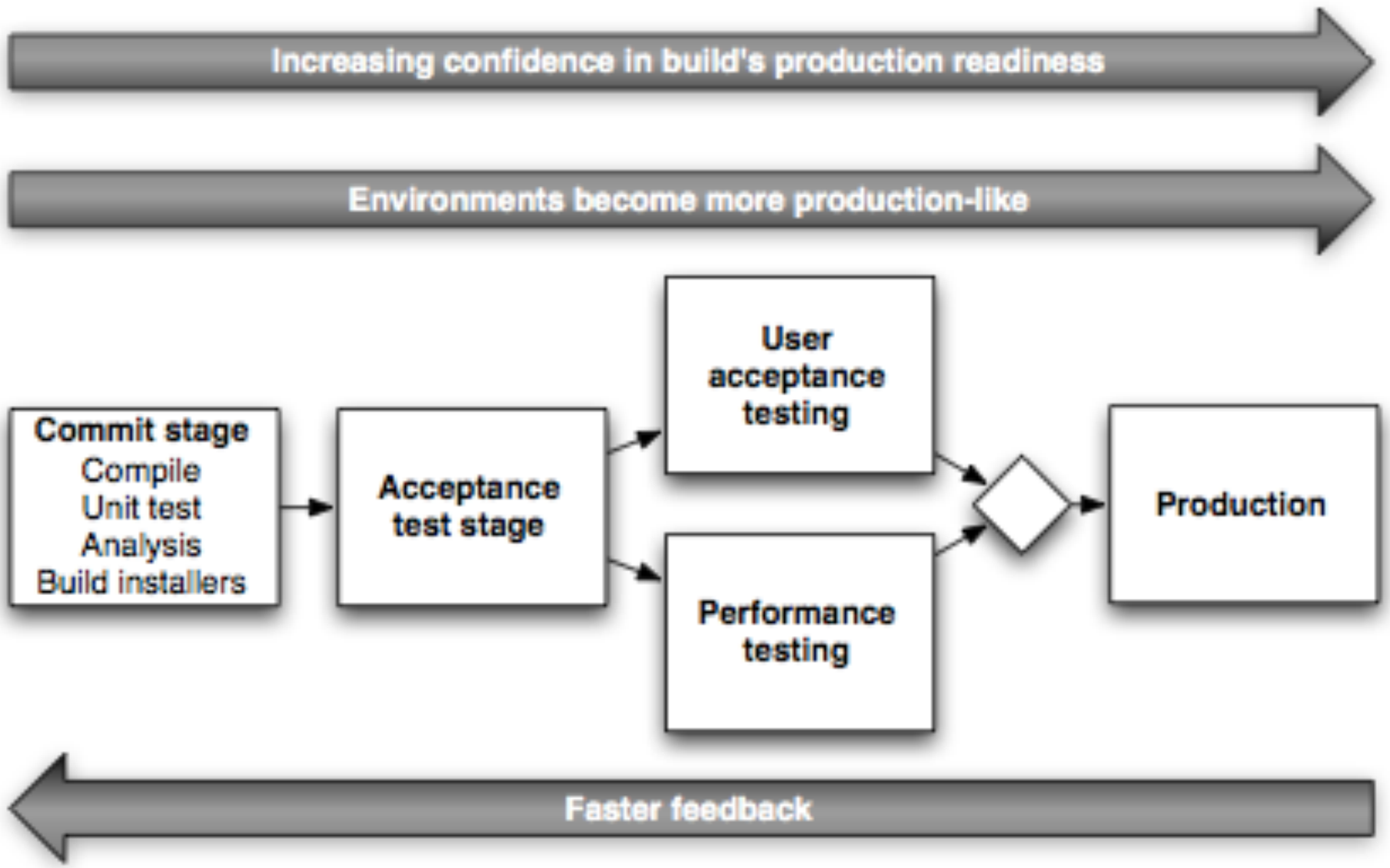
value stream mapping



deployment pipeline



deployment pipeline



deployment pipeline

principles

- create a repeatable, reliable process for releasing software
- automate almost everything
- keep everything in version control
- if it hurts, do it more often, and bring the pain forward
- build quality in
- done means released
- everybody is responsible for delivery
- continuous improvement

ask this question

- "How long would it take your organization to deploy a change that involved just one single line of code? Do you do this on a repeatable, reliable basis?"
- What gets in the way of getting software out of the door?

Mary and Tom Poppendieck, *Implementing Lean Software Development,* p59.

practices

only build your binaries once

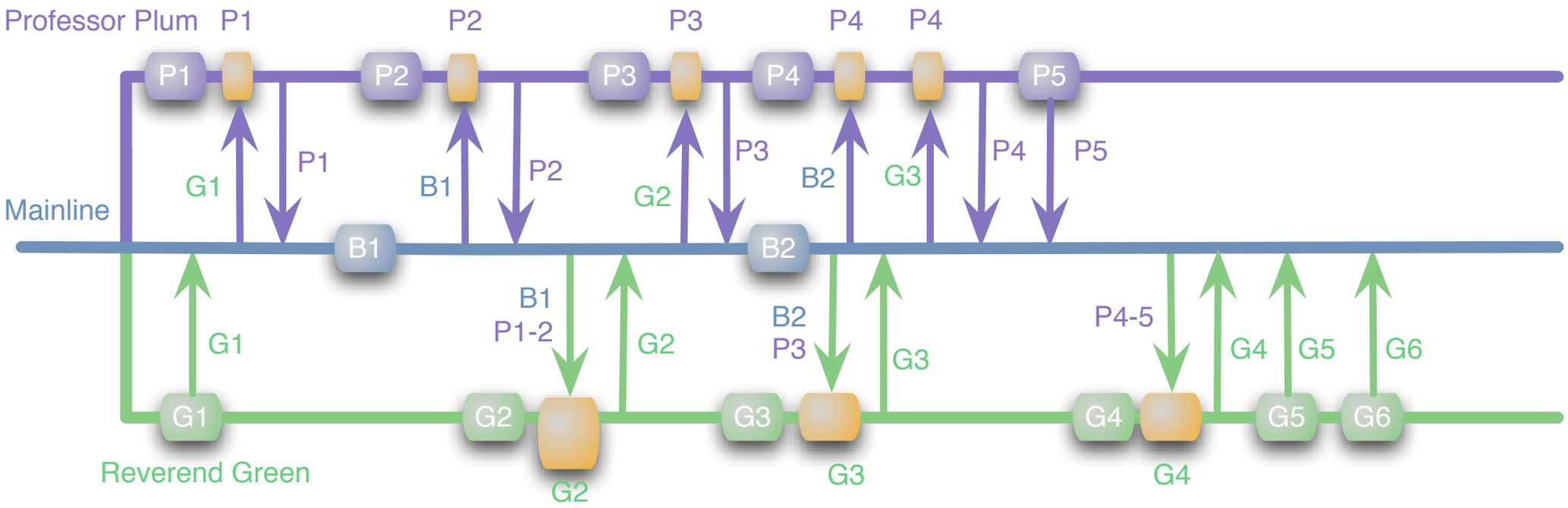
deploy the same way to every environment

smoke test your deployments

keep your environments similar

if anything fails, stop the line

continuous integration



everybody checks in to mainline

use branch by abstraction for architectural change

use feature bits to switch off incomplete features

Diagram invented by Martin Fowler

different kinds of testing

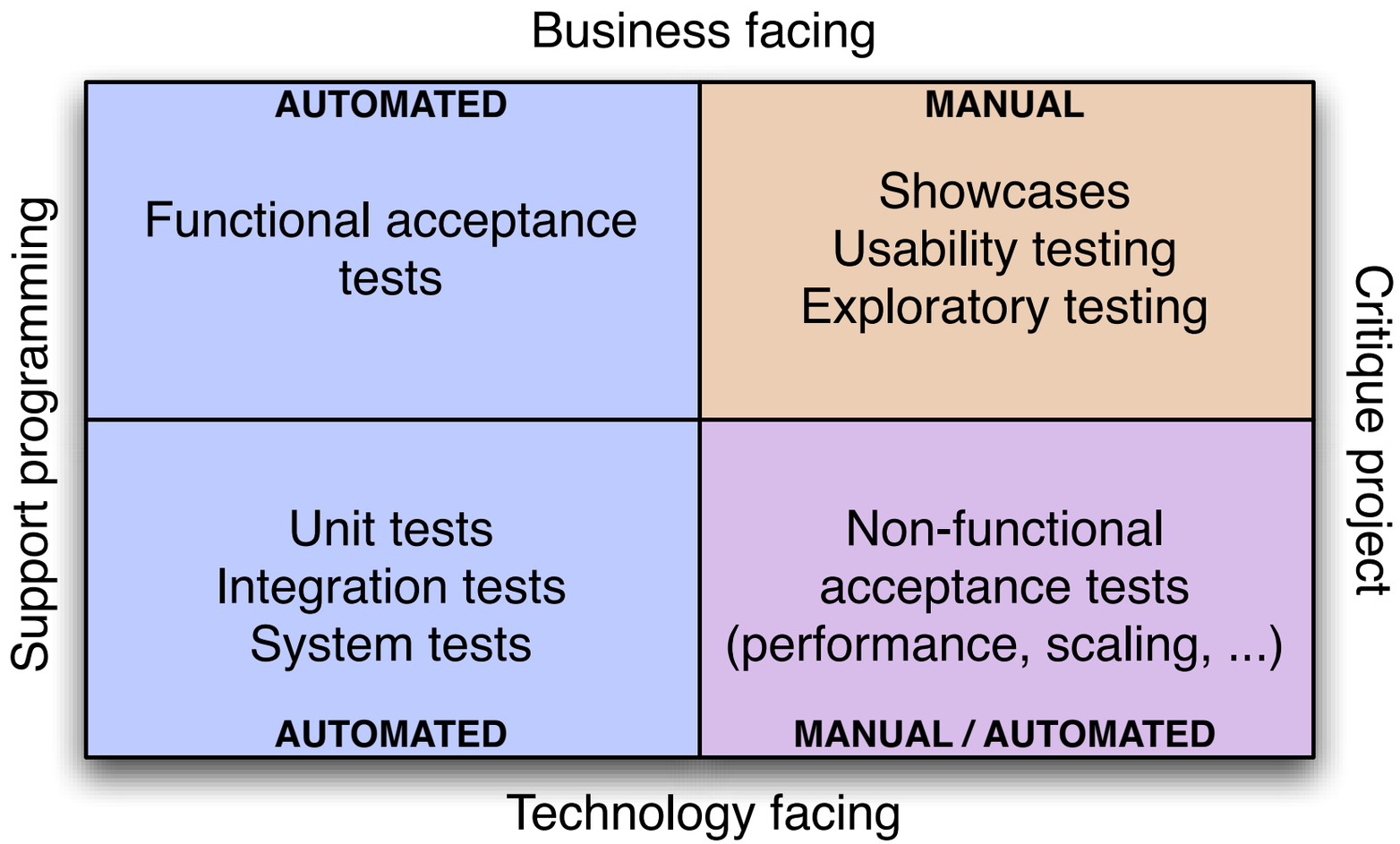
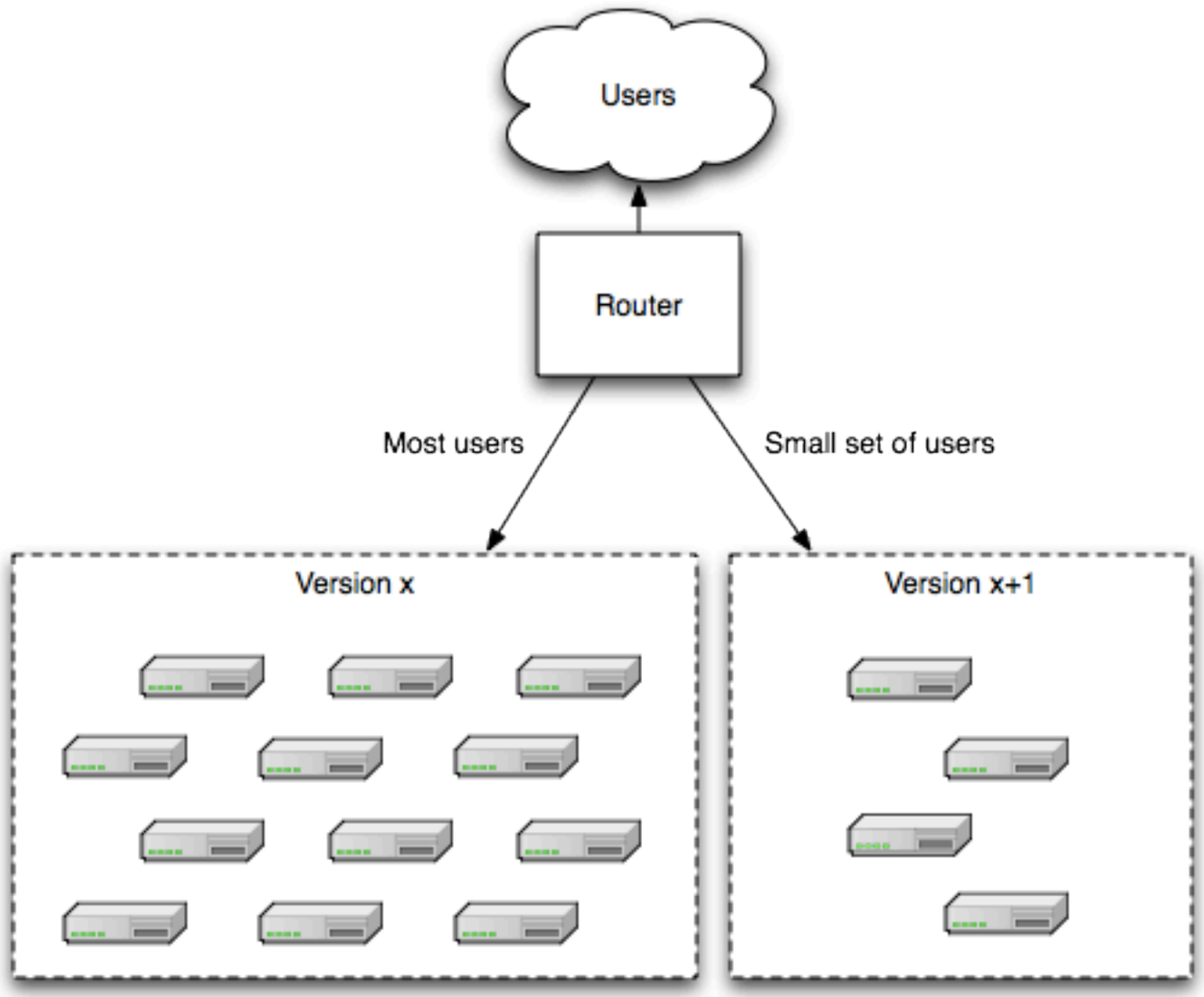
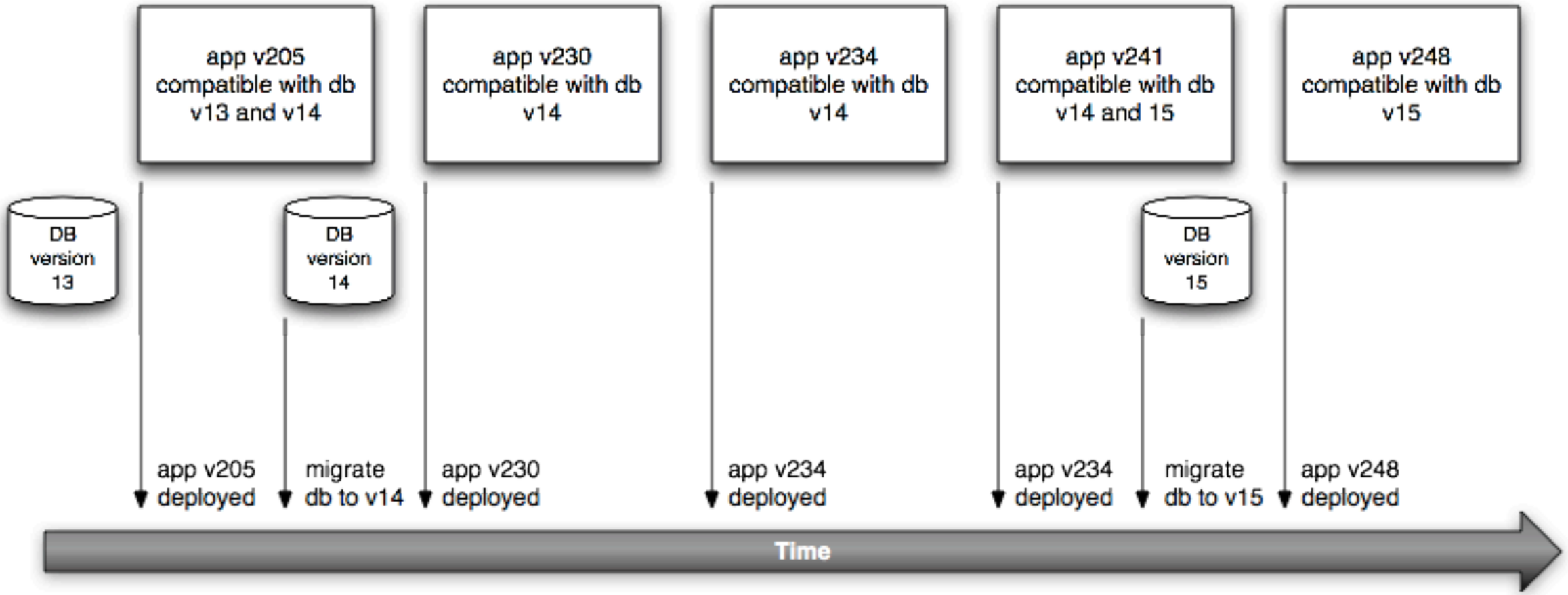


Diagram invented by Brian Marick

canary releasing



data migration



objections

Visibility and control over locking down

Compliance - automation over documentation

Auditing - see who does what

Make it easy to remediate outages

people are the key

Get everyone together at the beginning

Keep meeting

Make it easy for everyone to see what's happening

Continuous improvement (kaizen)

go™
Agile Release Management

ThoughtWorks studios

thank you!

http://continuousdelivery.com/
http://studios.thoughtworks.com/go
http://thoughtworks.com/