IBM Software

DevOps: Development Challenges and New Approaches

Chris Sharp
STSM, Chief Architect SWG Europe DevOps
IBM Master Inventor,
Member of IBM Academy of Technology

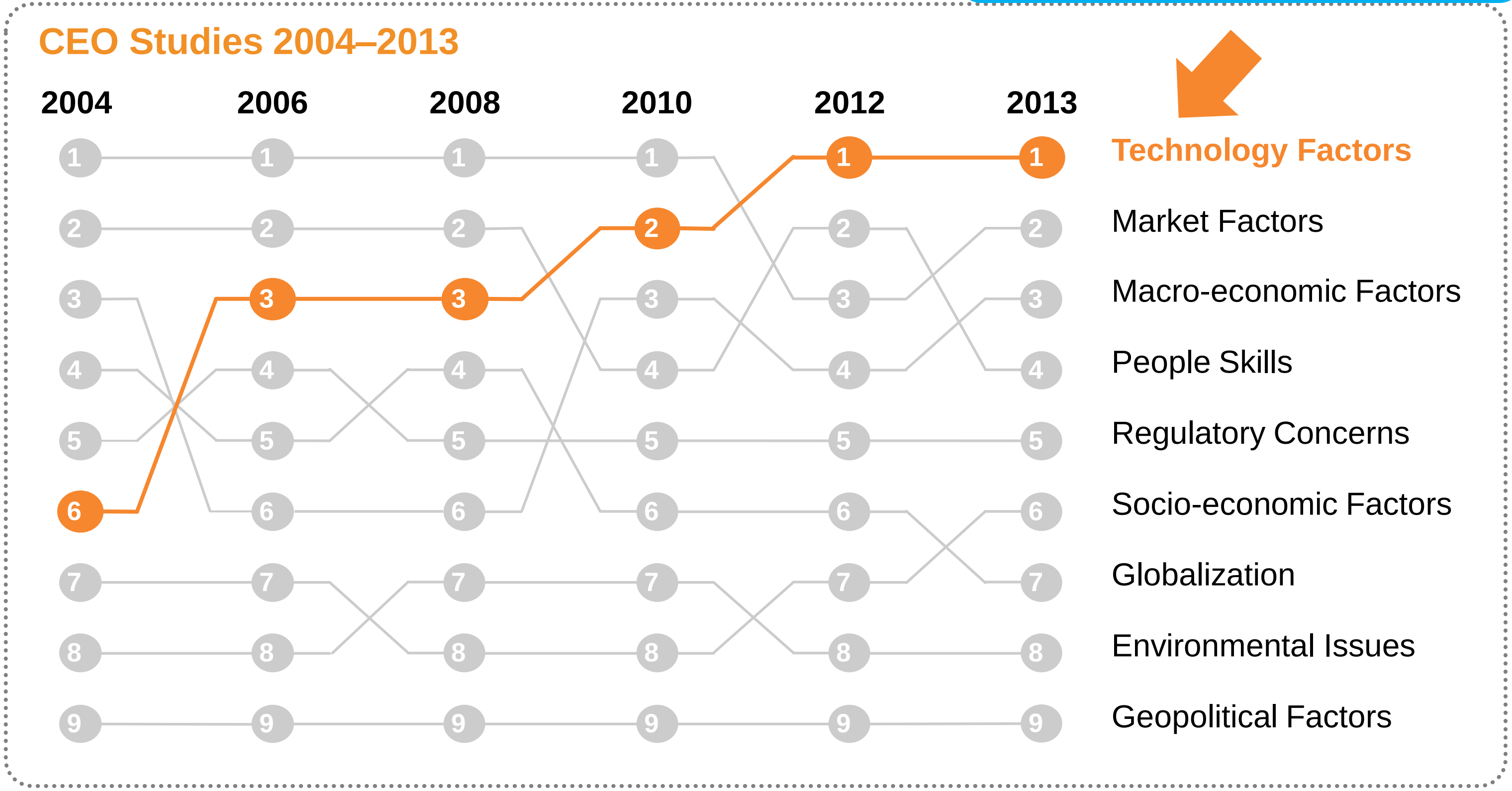
Agenda

- The Problem and the Need for Change
- What is DevOps?
  - Approaches
  - Techniques
  - Tools

Software Development Has Never Been More Critical

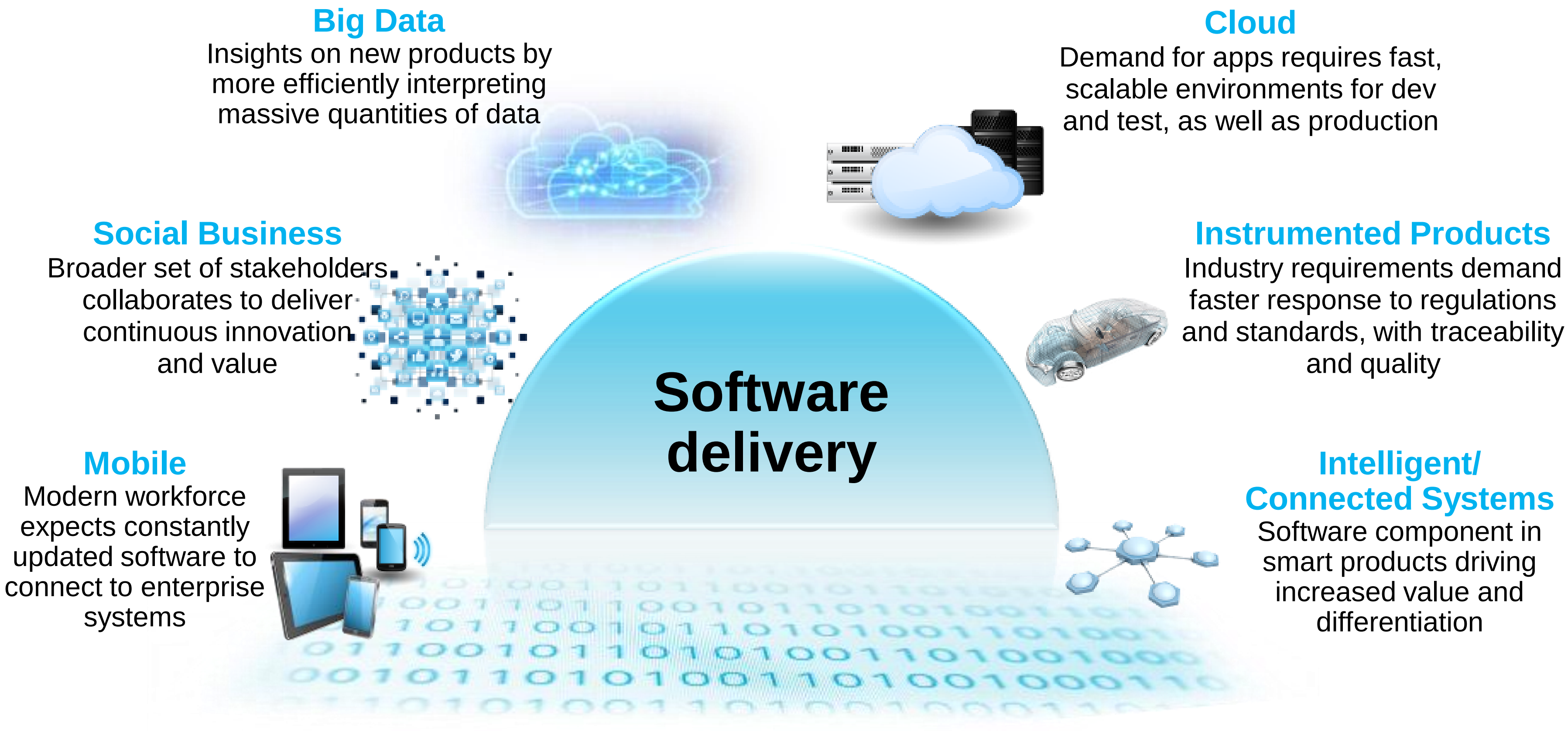
CEOs identify technology as the most important external force impacting their organizations – again

External forces that will impact the organization



Source: IBM Institute for Business Value, The Global CEO Study 2013. Question: "What are the most important external forces that will impact your organization over the next 3 to 5 years?"

Software Delivery Is Key To Exploiting Technology Trends



Big Data
Insights on new products by more efficiently interpreting massive quantities of data

Cloud
Demand for apps requires fast, scalable environments for dev and test, as well as production

Social Business
Broader set of stakeholders collaborates to deliver continuous innovation and value

Software delivery

Instrumented Products
Industry requirements demand faster response to regulations and standards, with traceability and quality

Mobile
Modern workforce expects constantly updated software to connect to enterprise systems

Intelligent/ Connected Systems
Software component in smart products driving increased value and differentiation

BUT…..

A lack of continuous delivery impacts the entire business enterprise in the new reality of “Systems Of Interaction”

Line-of-business
Takes too long to introduce or make changes to mobile apps and services

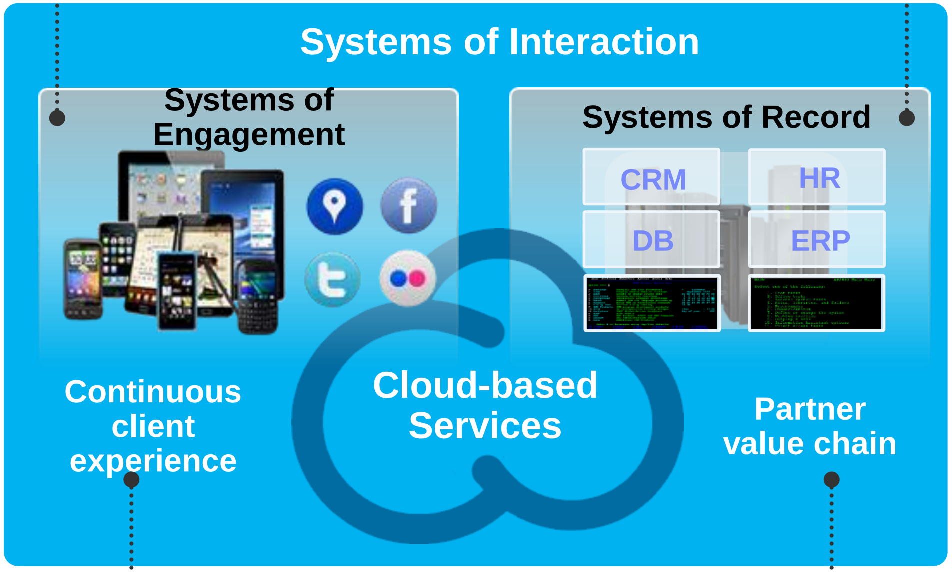
Operations
Rapid app releases impact system stability and compliance

>70%

of budgets devoted to maintenance and operations

4-6 weeks

to deliver even minor application changes to customers



Systems of Interaction

Systems of Engagement

Systems of Record

CRM

HR

DB

ERP

Continuous client experience

Cloud-based Services

Partner value chain

>45%

of customers experience production delays

>50%

of outsourced projects fail to meet objectives

Development/Test
Speed mismatch between faster moving front office and slower moving back office systems, delaying time to obtain feedback

Suppliers
Delivery in the context of agile

What is DevOps?

DevOps: a definition

DevOps (a portmanteau of development and operations) is a software development method that stresses **communication**, **collaboration** and **integration** between software developers and Information Technology(IT) professionals. DevOps is a response to the interdependence of software development and IT operations. It aims to help an organization **rapidly produce software products** and **services**.

-*Wikipedia*

Silos in the business

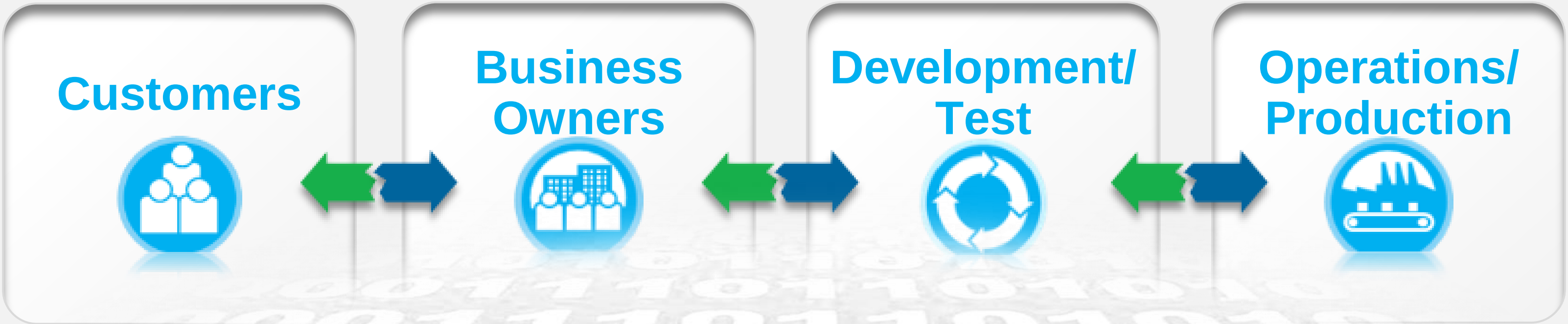
CHALLENGES

Lack of feedback between customers and the business leading to unclear requirements

Lack of visibility and **governance**, difficult to understand impact of a change

Slow deployment to test environments leave teams **waiting** and **unproductive**

Upgrade risk due to managing multiple application configurations and versions across servers



Customers

Business Owners

Development/ Test

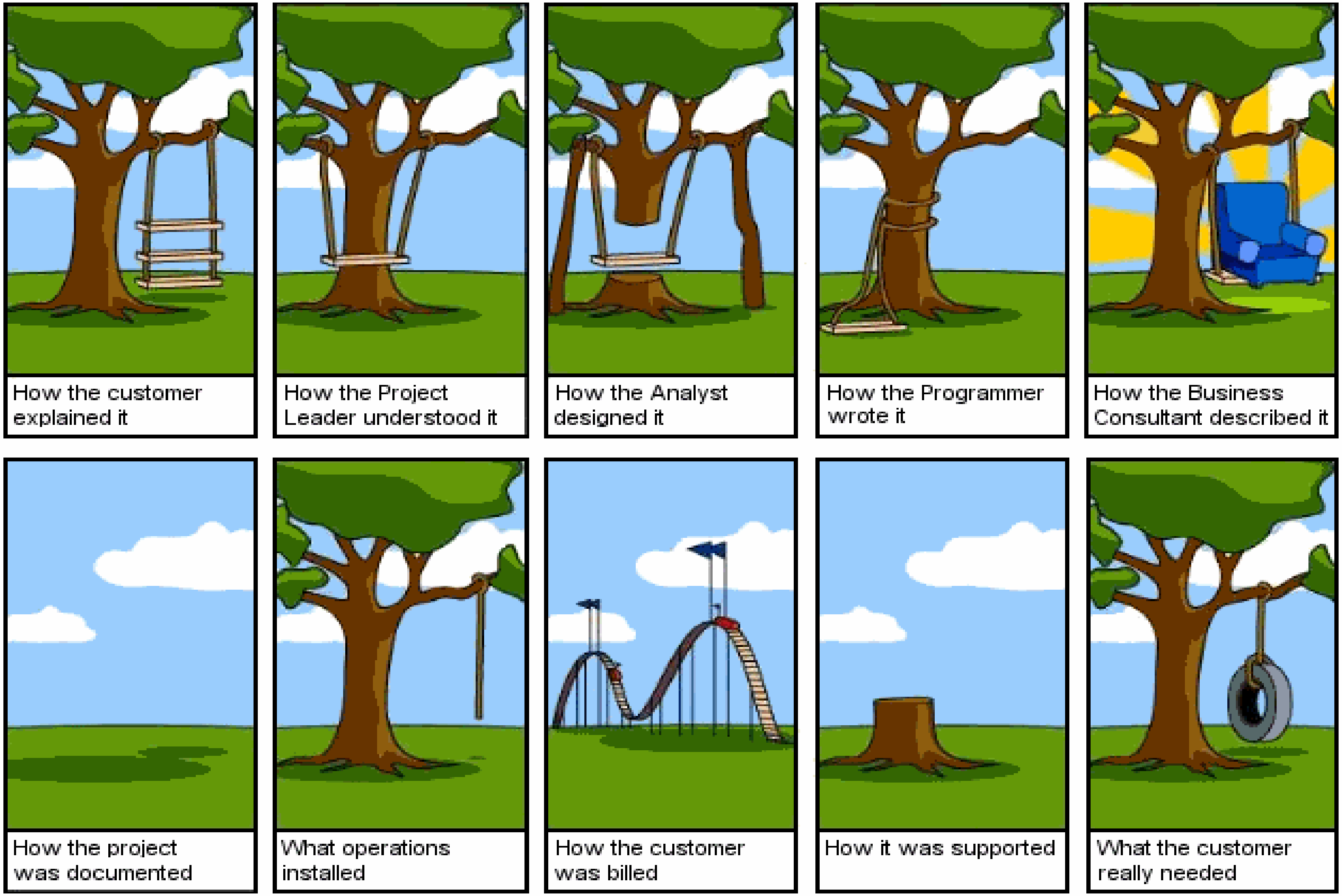
Operations/ Production

Software glitch costs trading firm Knight Capital $440 million in 45 minutes

New Zealand’s biggest phone company, Telecom paid out **$2.7 million** to some **47,000 customers who were overcharged** after a software glitch

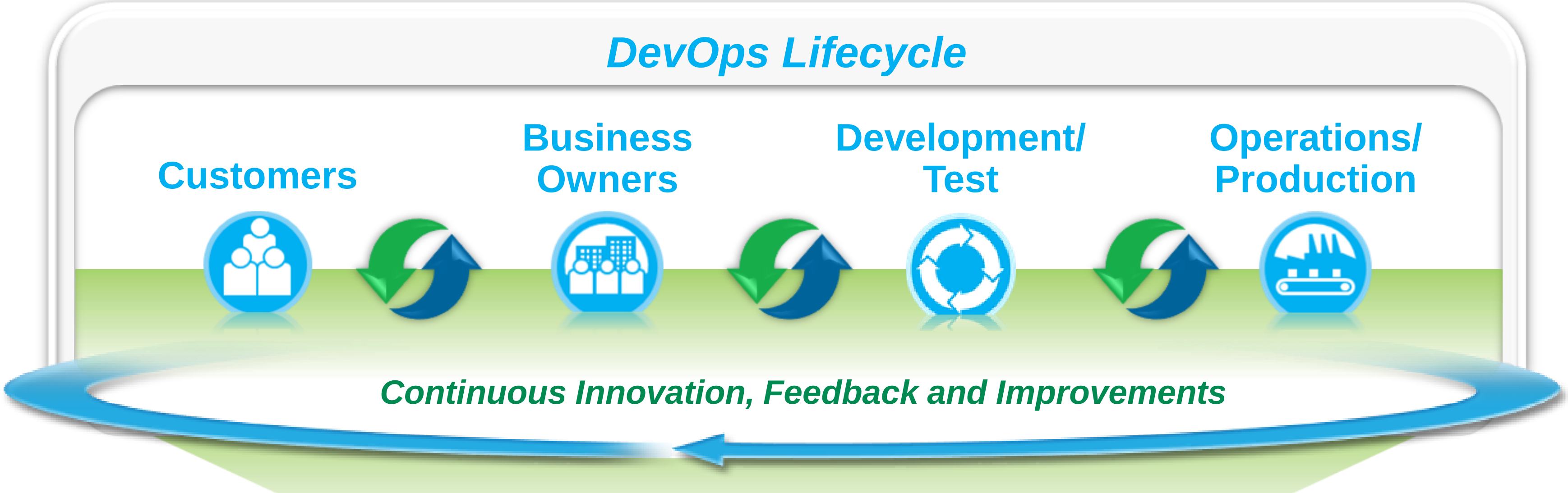
A bad software upgrade at a UK Bank left **millions unable to access money for four days**

A Classic Illustration Of The Integration Problem…


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

What is DevOps?

Capability for continuous software delivery that enables clients to seize market opportunities and reduce time to customer feedback



Accelerate Software Delivery

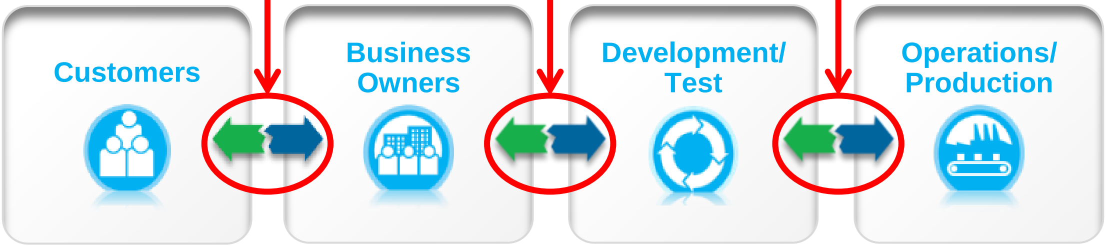
Balance speed, cost, quality and risk

Reduce time to customer feedback

Deliver better products, faster, and at lower cost

The key drivers

- Business Agility
- Agile Development
- Operational Discipline
- Cloud Computing

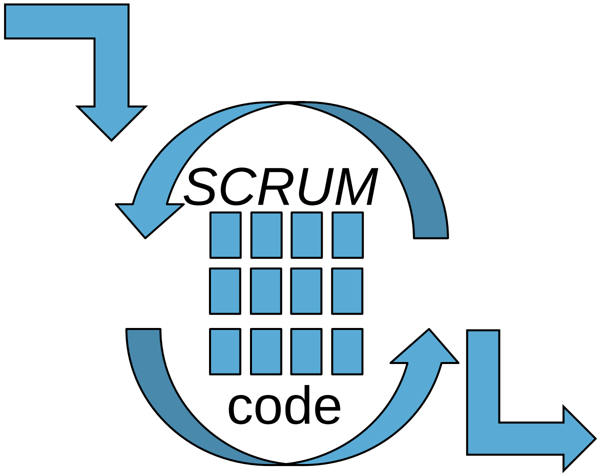


A Cautionary Tale: Water-SCRUM-fall

Many large enterprises end up implementing Agile in isolation in the engineering teams

Product Vision
Requirements Analysis
Specification
Design
Approval

water



SCRUM

code

fall

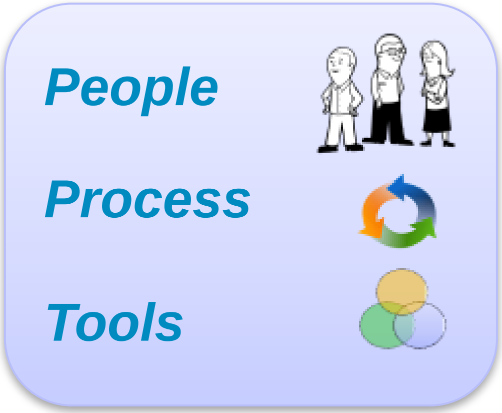
QA Testing
Acceptance
Operations

Whilst this might seem like an easy win, it misses the point

you need to optimize the whole to see the benefits

DevOps Principles and Values

- Develop and Test against production-like systems
- Iterative and frequent deployments using repeatable and reliable processes
- Continuously monitor and validate operational quality characteristics
- Amplify feedback loops everywhere
- Encourage a culture of experimentation
- Fundamentally, a spirit of flexibility, agility and automation



People

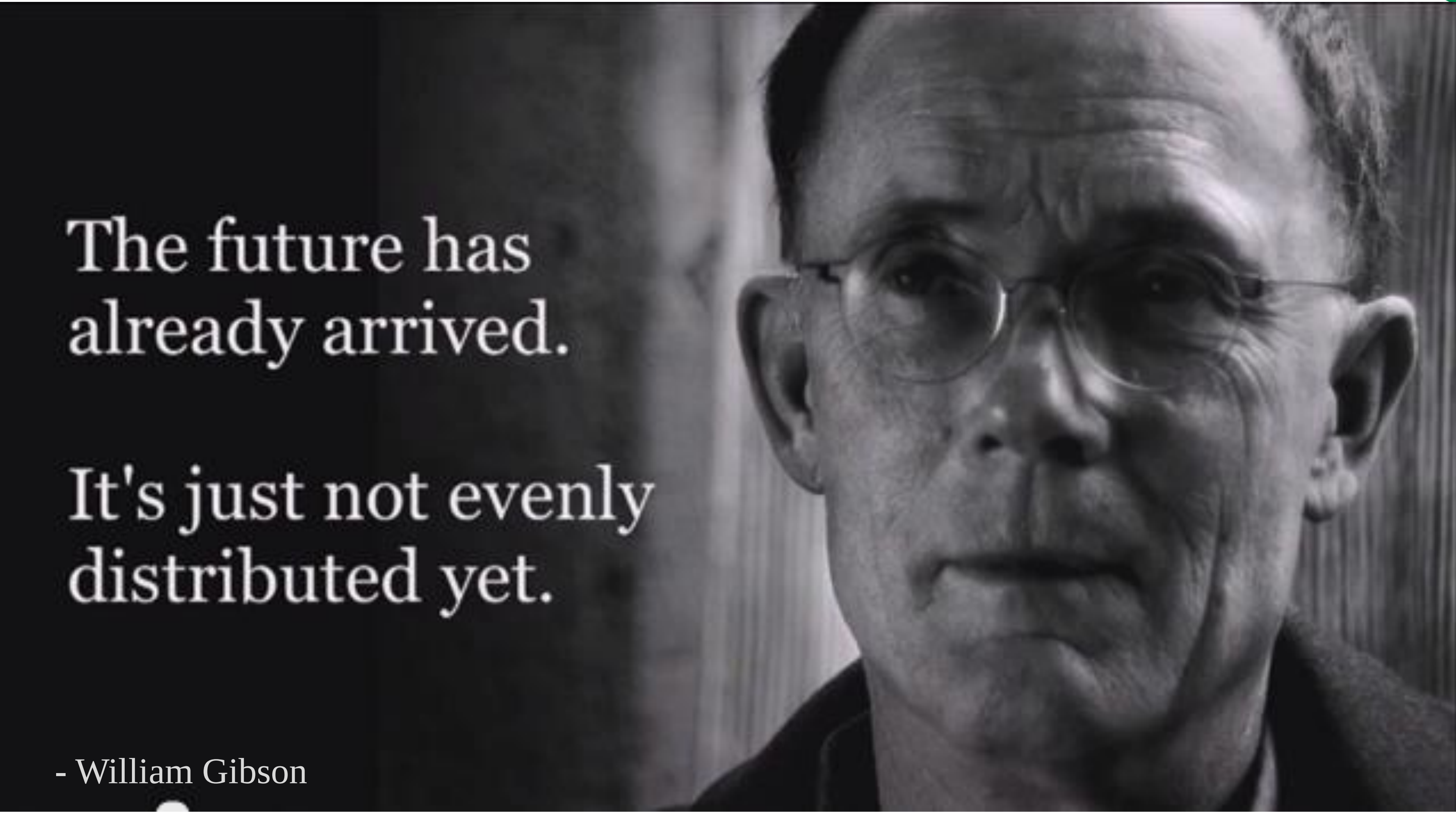
Process

Tools

How IBM Products Have Improved…

Lifecycle Measurements	2008	2010	2012 – 2014	Total Improvement
Project Initiation	30 days	10 days	2 days	**28 days**
Groomed Backlog	90 days	45 days	On-going	**89 days**
Overall Time To Development	120 days	55 days	3 days	**117 days**
Composite Build Time	36 hours	12 hours	5 hours	**700 %**
BVT Availability	N / A	18 hours	< 1hour	**17 hours**
Iteration Test Time	5 days	2 days	14 hours	**4 days**
Total Deployment Time	2 days	8 hours	4 hours -> 20 minutes	**2 days**
Overall Time To Production	9 days	3 days	2 days	**7 days**
Time Between Releases	12 Months	12 Months	3 Months	**9 Months**
Innovation / Maintenance	58% / 42%	64% / 36%	78% / 22%	**+20% / -20%**

How are other people doing this?

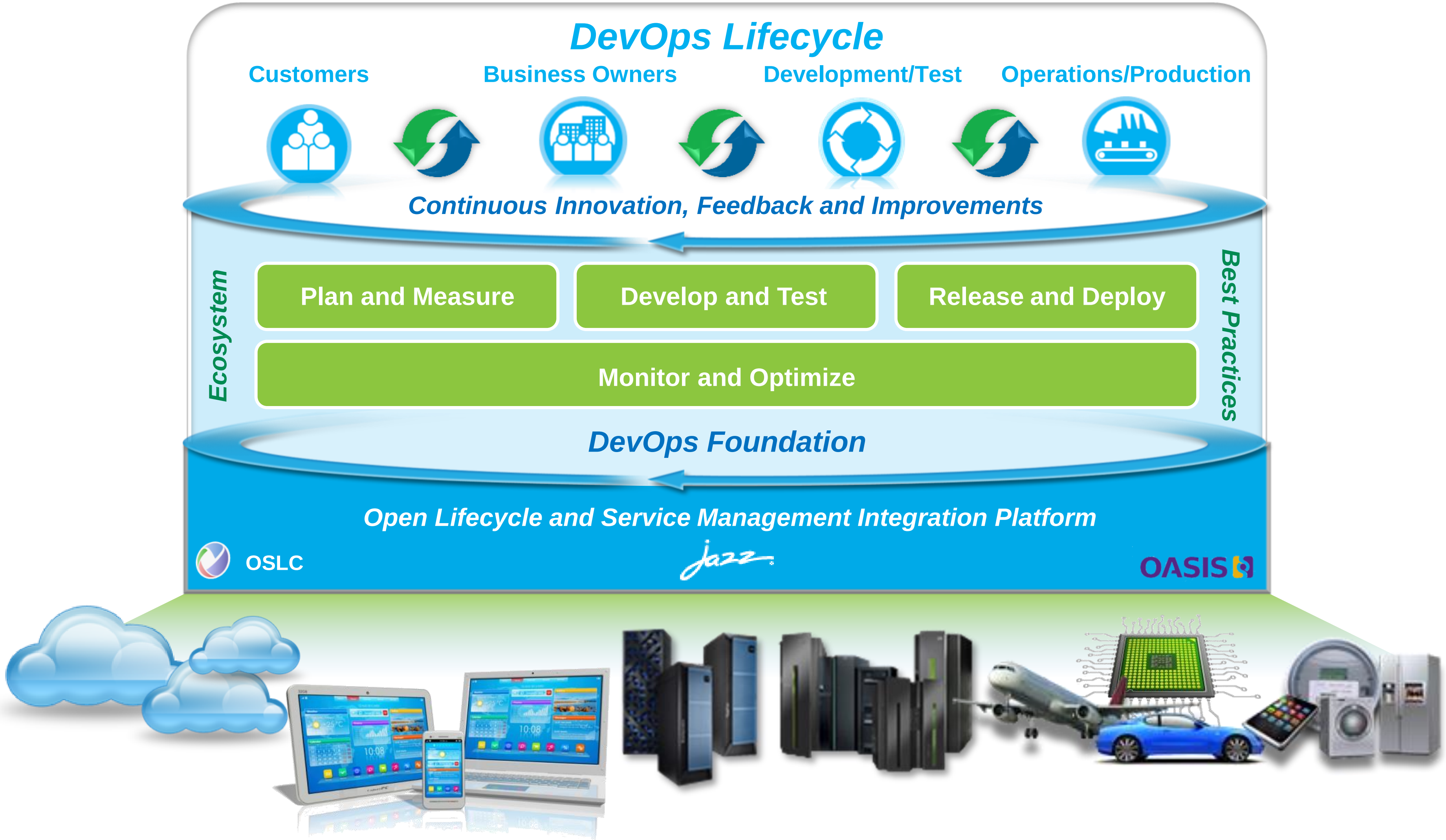


The future has already arrived.

It's just not evenly distributed yet.

- William Gibson

Adoption paths to a DevOps approach



DevOps Lifecycle

Customers

Business Owners

Development/Test

Operations/Production

Continuous Innovation, Feedback and Improvements

Ecosystem

Plan and Measure

Develop and Test

Release and Deploy

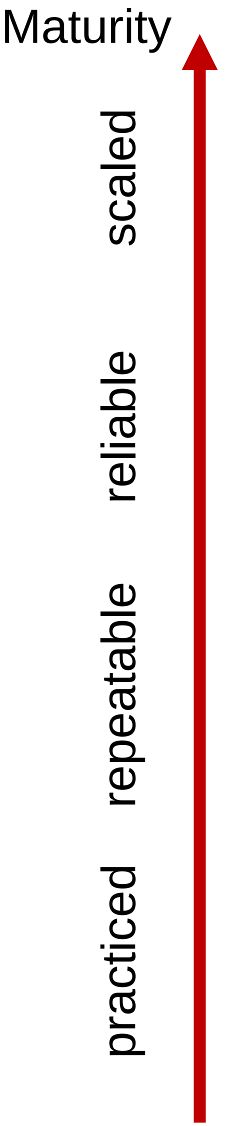
Monitor and Optimize

Best Practices

DevOps Foundation

Open Lifecycle and Service Management Integration Platform

Maturity of adoption



Maturity

scaled

Continuous improvement the norm

reliable

Efficient

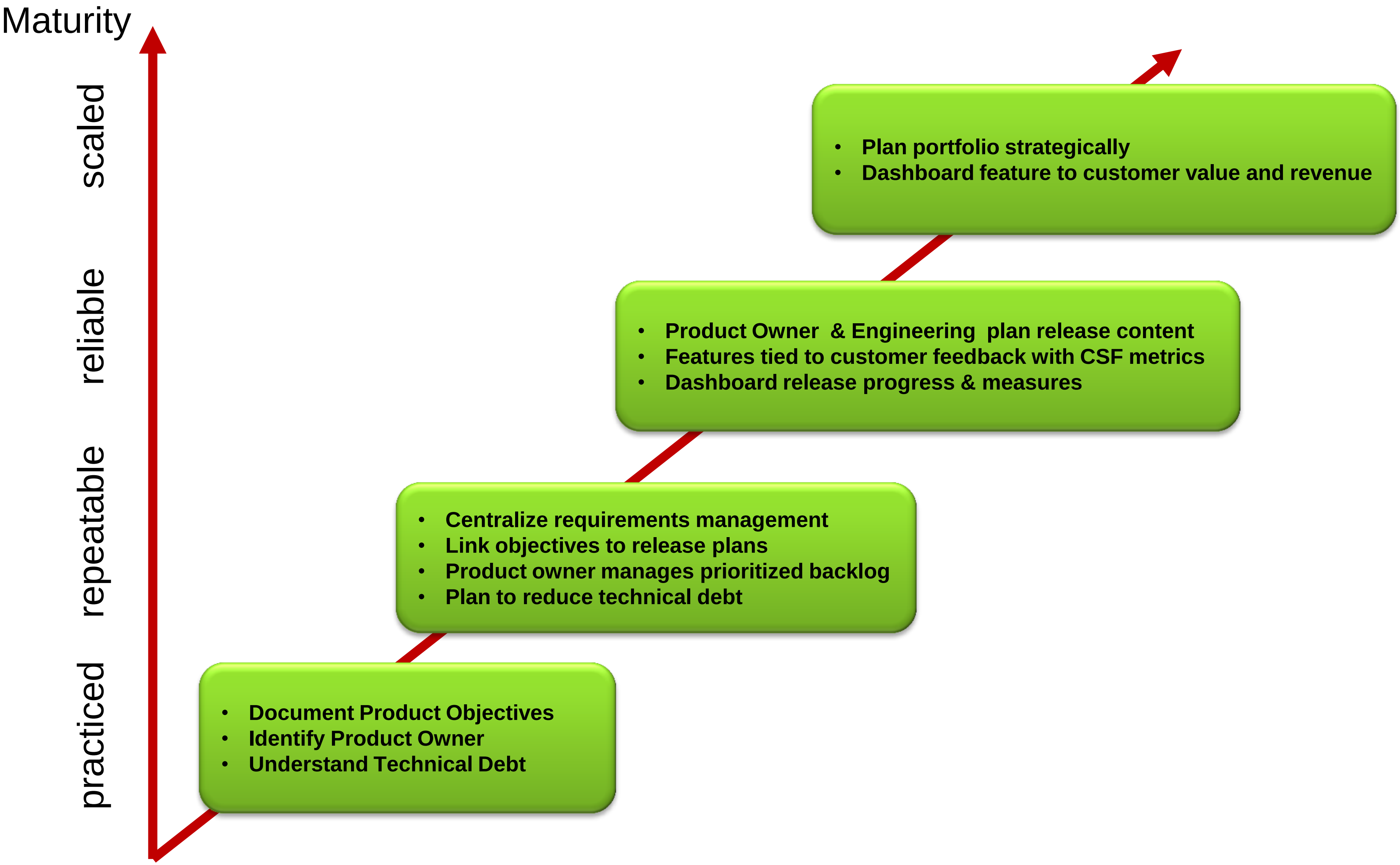
repeatable

The fundamentals are in place

practiced

Able to do it, but takes “heroics”

Plan & Measure



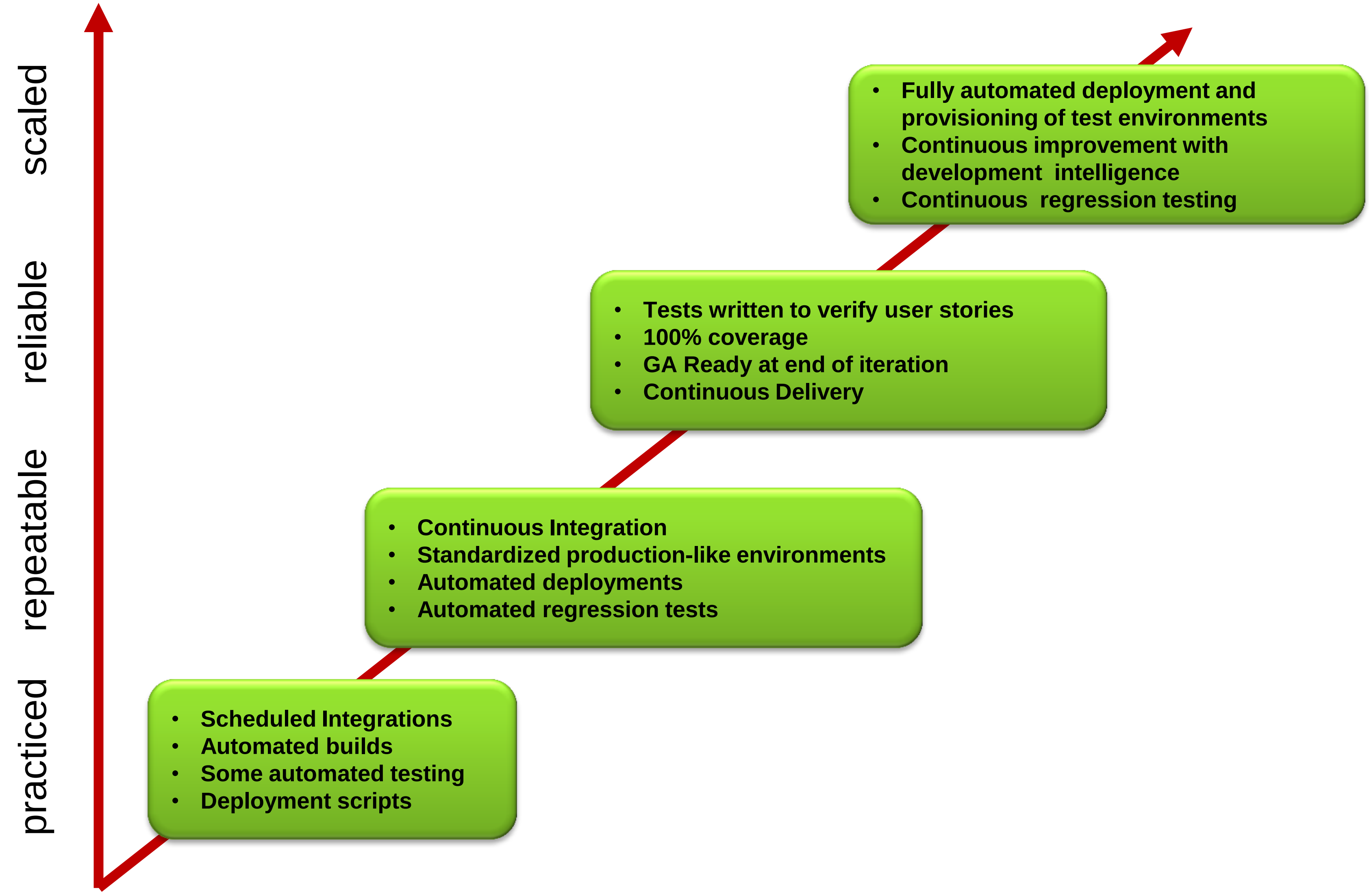
- Plan portfolio strategically
- Dashboard feature to customer value and revenue

- Product Owner & Engineering plan release content
- Features tied to customer feedback with CSF metrics
- Dashboard release progress & measures

- Centralize requirements management
- Link objectives to release plans
- Product owner manages prioritized backlog
- Plan to reduce technical debt

- Document Product Objectives
- Identify Product Owner
- Understand Technical Debt

Develop & Test



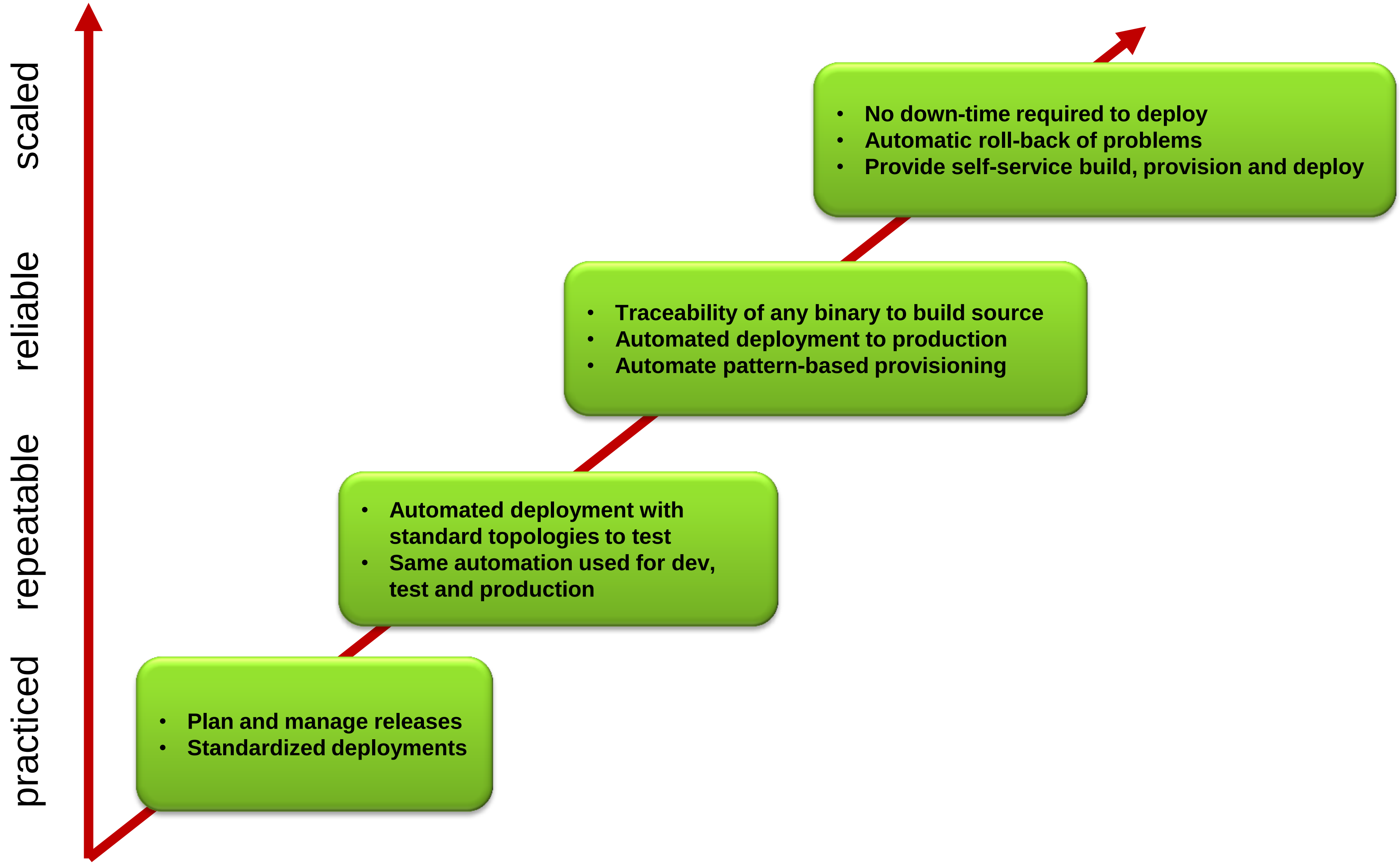
- Scheduled Integrations
- Automated builds
- Some automated testing
- Deployment scripts

- Continuous Integration
- Standardized production-like environments
- Automated deployments
- Automated regression tests

- Tests written to verify user stories
- 100% coverage
- GA Ready at end of iteration
- Continuous Delivery

- Fully automated deployment and provisioning of test environments
- Continuous improvement with development intelligence
- Continuous regression testing

Release & Deploy



practiced

- Plan and manage releases
- Standardized deployments

repeatable

- Automated deployment with standard topologies to test
- Same automation used for dev, test and production

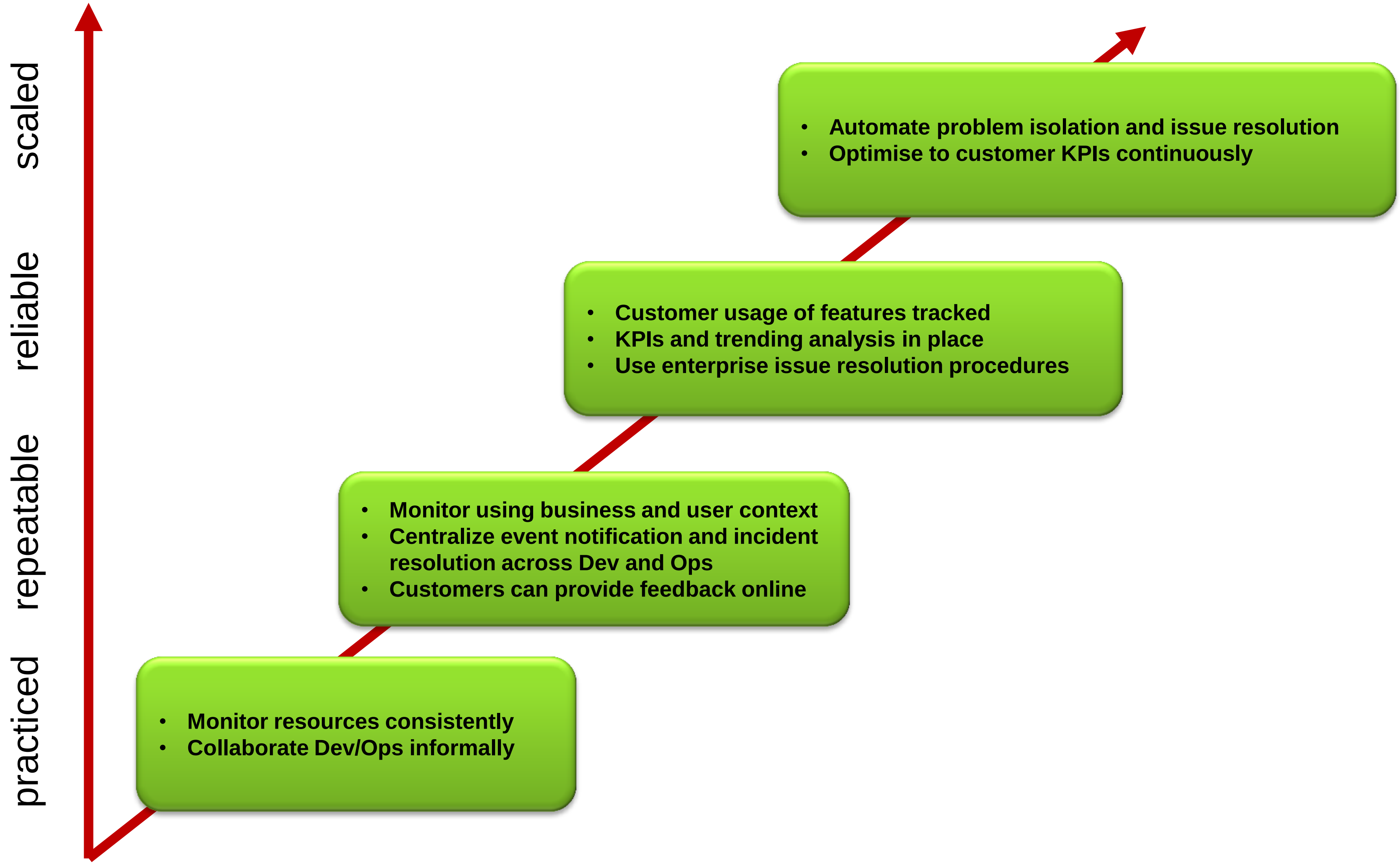
reliable

- Traceability of any binary to build source
- Automated deployment to production
- Automate pattern-based provisioning

scaled

- No down-time required to deploy
- Automatic roll-back of problems
- Provide self-service build, provision and deploy

Monitor & Optimize



- Automate problem isolation and issue resolution
- Optimise to customer KPIs continuously

- Customer usage of features tracked
- KPIs and trending analysis in place
- Use enterprise issue resolution procedures

- Monitor using business and user context
- Centralize event notification and incident resolution across Dev and Ops
- Customers can provide feedback online

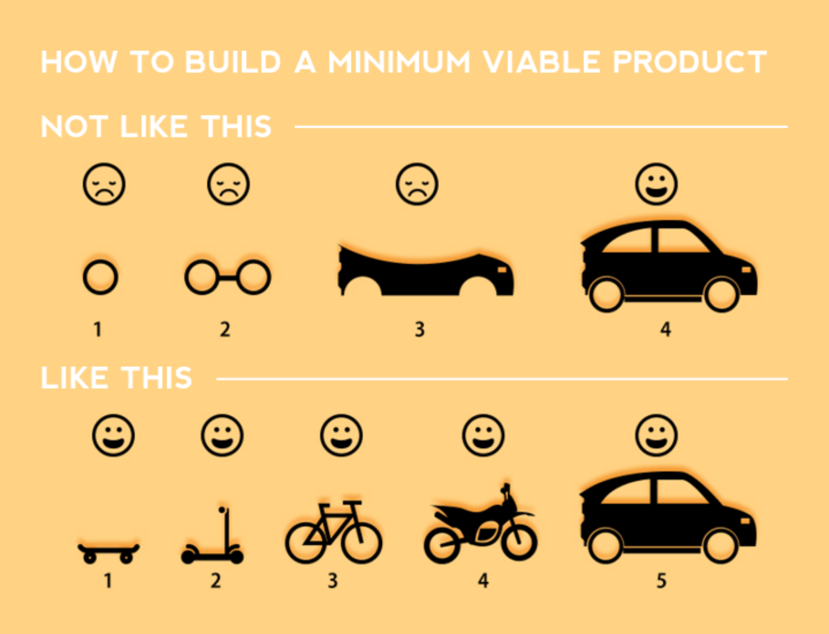
- Monitor resources consistently
- Collaborate Dev/Ops informally

Culture

- Shared goals and success measures
- End-to-end alignment and ownership of goals
- Continuous alignment across disciplines empowers a team

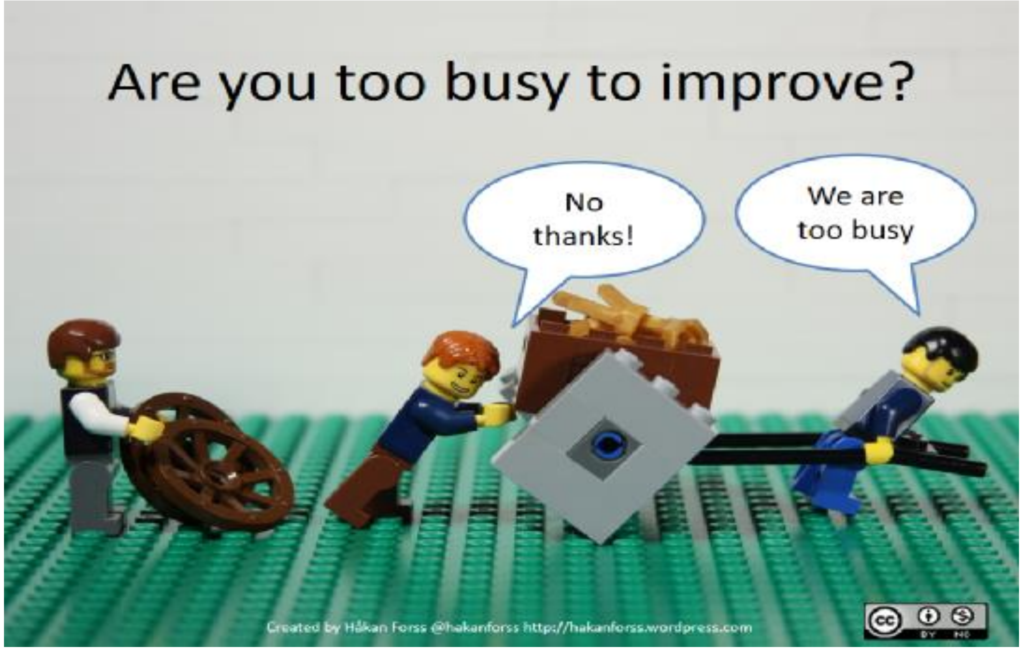
Culture

- Minimal Viable Product
  - – Focus on value at each step
  - – Keep focus on incremental progress
- Fail fast
  - – Encourage experimentation
  - – Blameless post-mortems
- Shift-Left
  - – Defects are cheaper and easier to fix the closer they are to the developer
- Culture of continuous improvement
  - – The *right* people are needed



Change

- Understand why you want to change
- Get measurable change fast, even if your goal takes years
- Start with continuous integration
- Create culture of continuous improvement



Created by Håkan Forss @hakanforss http://hakanforss.wordpress.com

DevOps Techniques

Continuous Integration…

- ….is an attitude, not a tool.
  (James Shore, 2005)

- It’s a commitment from the team that the latest code in the repo will build and pass all tests, and they will check their code in *frequently*.

- Tools can help to automate this, but the principals must be well understood and upheld by all.

Continuous Integration – the process

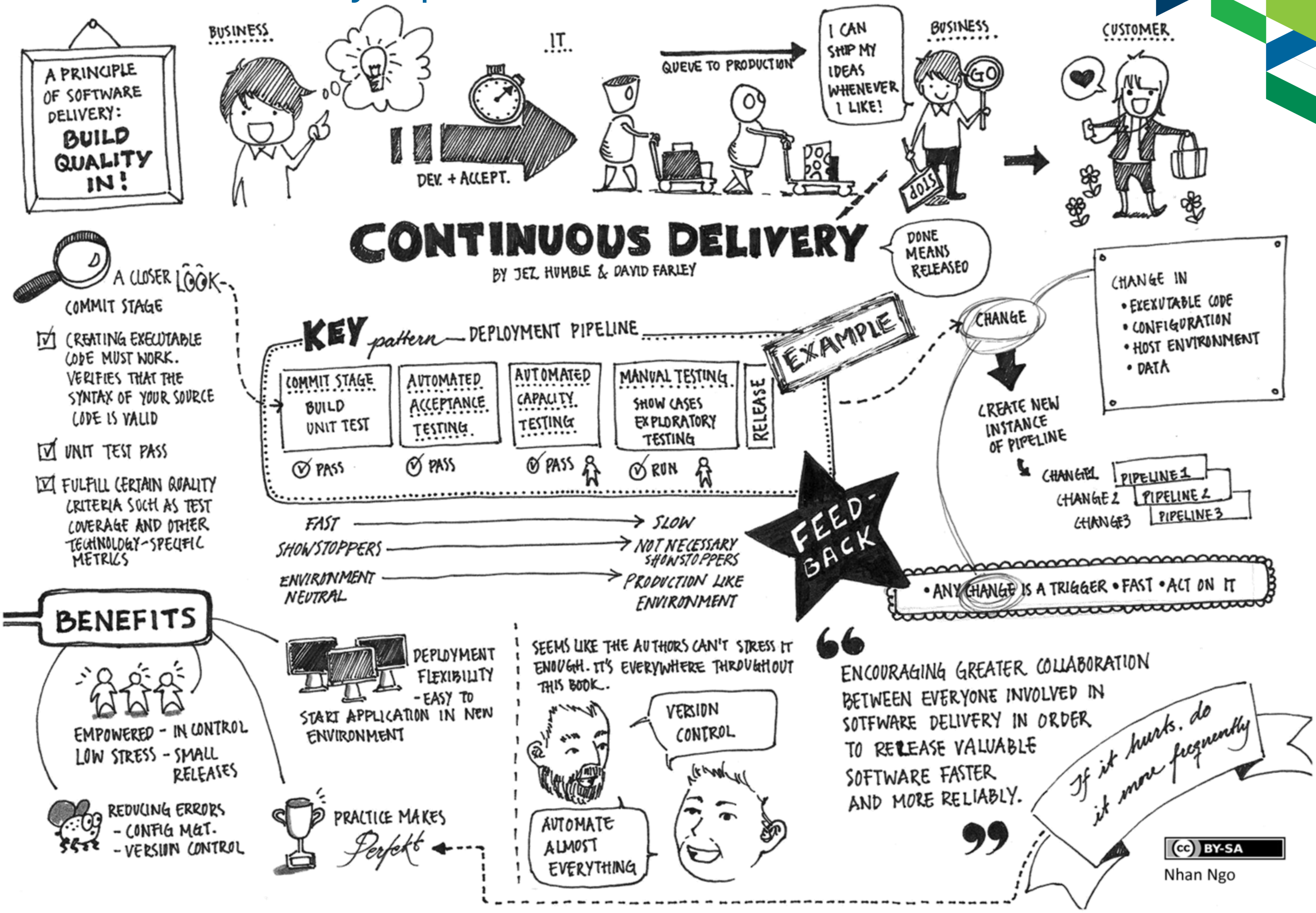
1. Developer checks out from repo and works on a change
2. Developer runs a local build and full test suite locally
3. If success, then the developer pulls the latest source from the repo and rebuilds/tests
4. If success, then developer checks in their change and a CI build run
5. If failure, **STOP THE LINE** until fixed,
   a) If a quick fix, fix immediately and re-run build/test
   b) If not fixed within 10 mins, rollback change
6. If success, done!

Continuous Delivery and Continuous Deployment

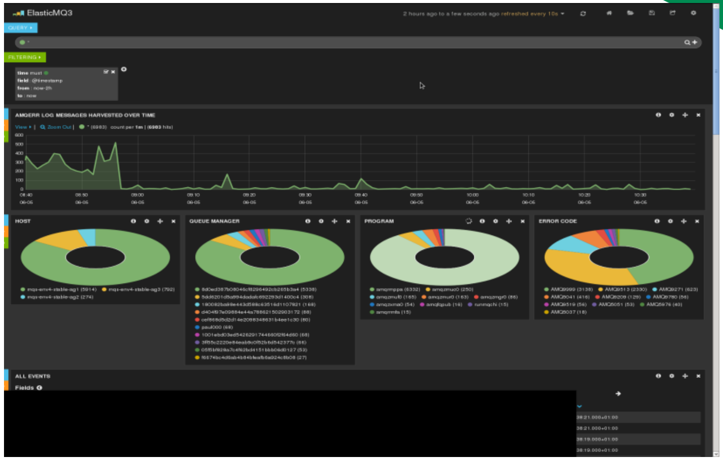
- **Continuous Delivery**
  - the practice of an Agile team continuously having real, GA-Ready code that could be -- *if desired* -- shipped or deployed at any time.
  - Note that it does not *have* to be shipped or deployed, but it could be if desired.
  - Whole purpose is to eliminate risk

- **Continuous Deployment**
  - the practice of actually shipping or deploying GA-Ready code into **production** on a continuous basis.
  - Dev and Ops have to work together, address each other's needs (i.e., high confidence in the quality, minimal downtime, automated rollback in case of errors, etc.), and trust each other to do the right thing.
  - This obvious need for Development and Operations to work together as one team is what gave rise to the initial idea of "DevOps."

Continuous Delivery Pipeline


A PRINCIPLE OF SOFTWARE DELIVERY: BUILD QUALITY IN!
BUSINESS
IT
DEV. + ACCEPT.
QUEUE TO PRODUCTION
I CAN SHIP MY IDEAS WHENEVER I LIKE!
BUSINESS
GO
STOP
CUSTOMER
CONTINUOUS DELIVERY
BY JEZ HUMBLE & DAVID FARLEY
DONE MEANS RELEASED
A CLOSER LOOK
COMMIT STAGE
☑ CREATING EXECUTABLE CODE MUST WORK. VERIFIES THAT THE SYNTAX OF YOUR SOURCE CODE IS VALID
☑ UNIT TEST PASS
☑ FULFILL CERTAIN QUALITY CRITERIA SUCH AS TEST COVERAGE AND OTHER TECHNOLOGY-SPECIFIC METRICS
KEY pattern — DEPLOYMENT PIPELINE
EXAMPLE
COMMIT STAGE
BUILD UNIT TEST
AUTOMATED ACCEPTANCE TESTING
AUTOMATED CAPACITY TESTING
MANUAL TESTING
SHOW CASES EXPLORATORY TESTING
RELEASE
✓ PASS
✓ PASS
✓ PASS
✓ RUN
FAST → SLOW
SHOWSTOPPERS → NOT NECESSARY SHOWSTOPPERS
ENVIRONMENT NEUTRAL → PRODUCTION LIKE ENVIRONMENT
FEED-BACK
CHANGE
CREATE NEW INSTANCE OF PIPELINE
CHANGE IN
• EXECUTABLE CODE
• CONFIGURATION
• HOST ENVIRONMENT
• DATA
CHANGE1 PIPELINE 1
CHANGE 2 PIPELINE 2
CHANGE3 PIPELINE 3
• ANY CHANGE IS A TRIGGER • FAST • ACT ON IT
BENEFITS
EMPOWERED - IN CONTROL
LOW STRESS - SMALL RELEASES
REDUCING ERRORS
- CONFIG MGT.
- VERSION CONTROL
DEPLOYMENT FLEXIBILITY
- EASY TO START APPLICATION IN NEW ENVIRONMENT
PRACTICE MAKES Perfekt
SEEMS LIKE THE AUTHORS CAN'T STRESS IT ENOUGH. IT'S EVERYWHERE THROUGHOUT THIS BOOK.
VERSION CONTROL
AUTOMATE ALMOST EVERYTHING
“ENCOURAGING GREATER COLLABORATION BETWEEN EVERYONE INVOLVED IN SOTFWARE DELIVERY IN ORDER TO RELEASE VALUABLE SOFTWARE FASTER AND MORE RELIABLY.”
If it hurts, do it more frequently
CC BY-SA
Nhan Ngo

Continuous Monitoring

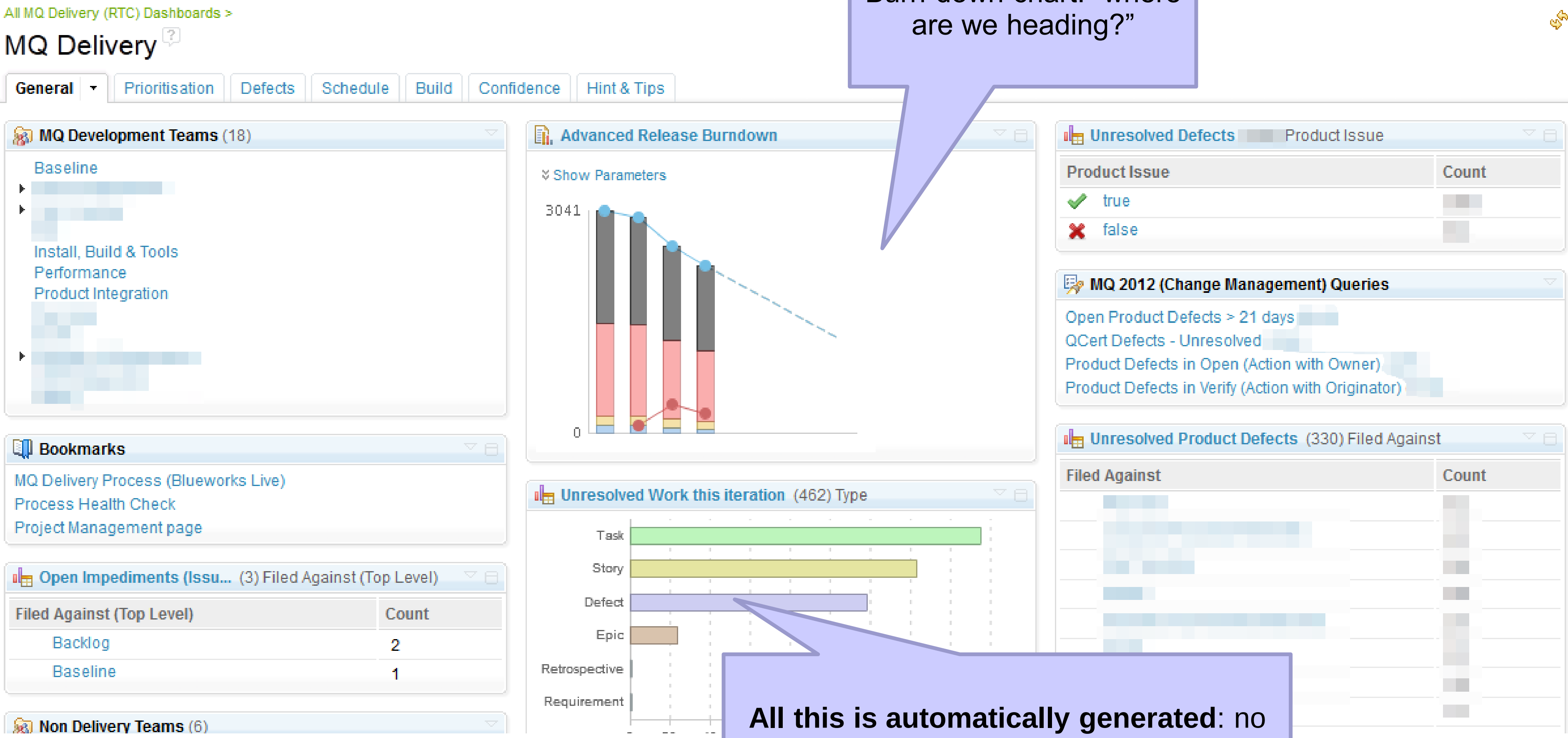

ElasticMQ3
2 hours ago to a few seconds ago refreshed every 10s
QUERY
FILTERING
time must
field : @timestamp
from : now-2h
to : now
AMQERR LOG MESSAGES HARVESTED OVER TIME
View | Zoom Out | * (6983) count per 1m | (6983 hits)
600
500
400
300
200
100
0
08:40 08-05
08:50 08-05
09:00 08-05
09:10 08-05
09:20 08-05
09:30 08-05
09:40 08-05
09:50 08-05
10:00 08-05
10:10 08-05
10:20 08-05
10:30 08-05
HOST
mqs-env4-stable-ag1 (5914)
mqs-env4-stable-ag3 (792)
mqs-env4-stable-ag2 (274)
QUEUE MANAGER
8d0ed387b08046cf8296492cb265b3a4 (5338)
5dd6201d8a994dadafc692293d1400c4 (306)
180082ba98e443d589c63516d1107821 (168)
d404f97e09884e44a788621502903172 (88)
cef868d5d2d14e2098348631b4ee1c30 (80)
paul000 (68)
1001ebd03ed54262917446602f64d60 (66)
3ff55c2220e84eab9c0f52b6d542377c (66)
05f5bf828a7c4f62bd4151bbb06d0127 (53)
f6674bc4d6ab4b84bfeafb6a924c8b08 (27)
PROGRAM
amqrmppa (6332)
amqzmuc0 (250)
amqzmuf0 (165)
amqzmur0 (163)
amqzmgr0 (86)
amqzxma0 (54)
amqfqpub (16)
runmqchi (15)
amqrrmfa (15)
ERROR CODE
AMQ9999 (3136)
AMQ9513 (2330)
AMQ9271 (623)
AMQ6041 (416)
AMQ9208 (129)
AMQ9780 (56)
AMQ9519 (56)
AMQ6051 (53)
AMQ5976 (40)
AMQ6037 (18)
ALL EVENTS
Fields

[redacted]

38:21.000+01:00
38:21.000+01:00
38:19.000+01:00
38:19.000+01:00

- Real-time Dashboards are your friend
- Real-time monitoring of your CD pipeline performance
- Real-time monitoring of your test and production environments
- Real-time monitoring of your customers experience and feedback
- Automate data collection…don’t ask your team to collect it by hand

Project Dashboard example



All MQ Delivery (RTC) Dashboards >

MQ Delivery

General | Prioritisation | Defects | Schedule | Build | Confidence | Hint & Tips

MQ Development Teams (18)

- Baseline
- [redacted]
- Install, Build & Tools
- Performance
- Product Integration
- [redacted]

Bookmarks

- MQ Delivery Process (Blueworks Live)
- Process Health Check
- Project Management page

Open Impediments (Issu... (3) Filed Against (Top Level)

Filed Against (Top Level)	Count
Backlog	2
Baseline	1

Non Delivery Teams (6)

Advanced Release Burndown

Show Parameters

Unresolved Work this iteration (462) Type

Unresolved Defects [redacted] Product Issue

Product Issue	Count
✓ true	[redacted]
✗ false	[redacted]

MQ 2012 (Change Management) Queries

- Open Product Defects > 21 days [redacted]
- QCert Defects - Unresolved [redacted]
- Product Defects in Open (Action with Owner) [redacted]
- Product Defects in Verify (Action with Originator) [redacted]

Unresolved Product Defects (330) Filed Against

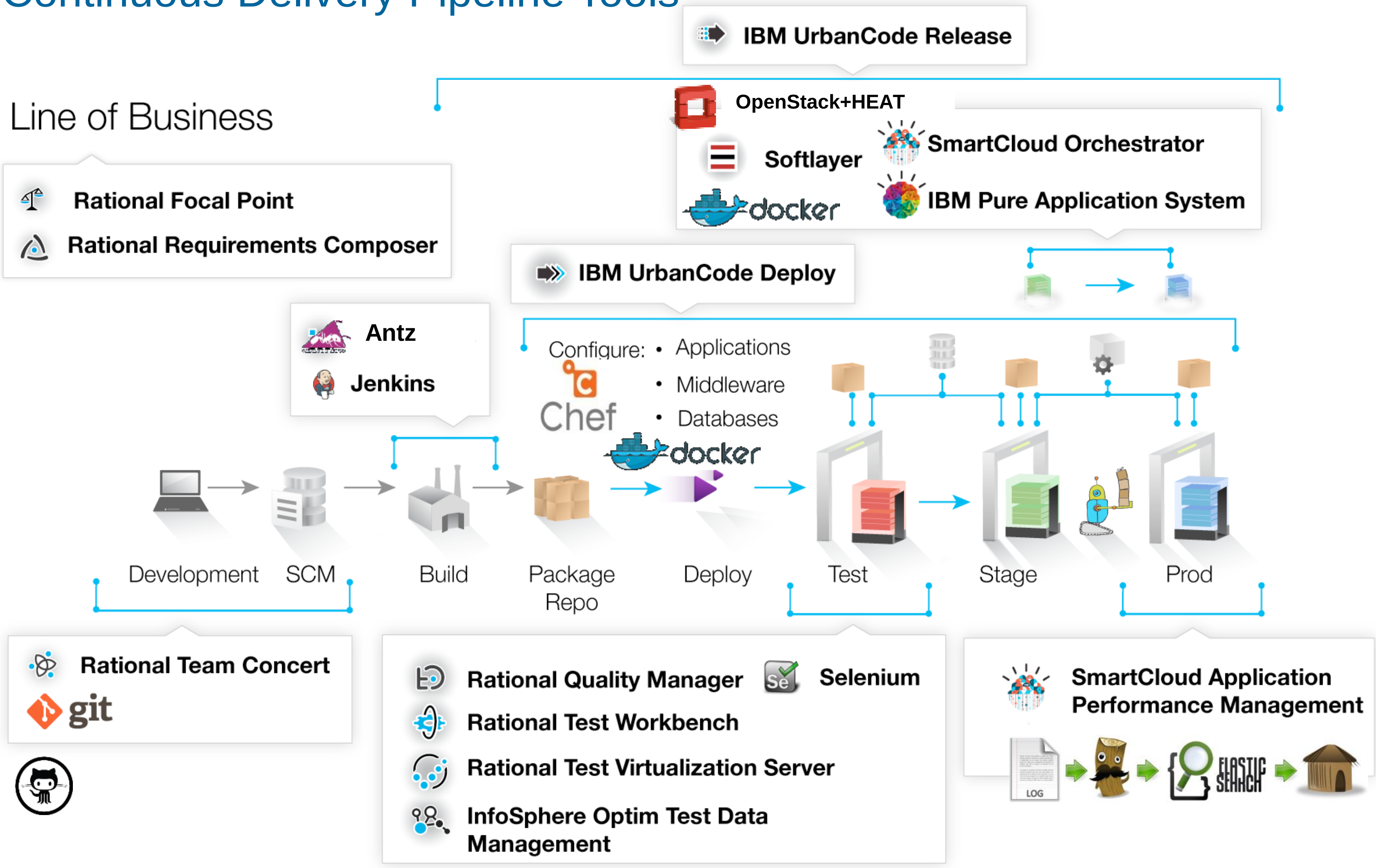
Filed Against	Count
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]

Burn-down chart: “where are we heading?”

All this is automatically generated: no overhead creating charts!

DevOps Tools

Continuous Delivery Pipeline Tools



IBM UrbanCode Release

OpenStack+HEAT

Softlayer

SmartCloud Orchestrator

docker

IBM Pure Application System

Line of Business

Rational Focal Point

Rational Requirements Composer

IBM UrbanCode Deploy

Antz

Jenkins

Configure:

- Applications
- Middleware
- Databases

Chef

docker

Development

SCM

Build

Package Repo

Deploy

Test

Stage

Prod

Rational Team Concert

git

Rational Quality Manager

Selenium

Rational Test Workbench

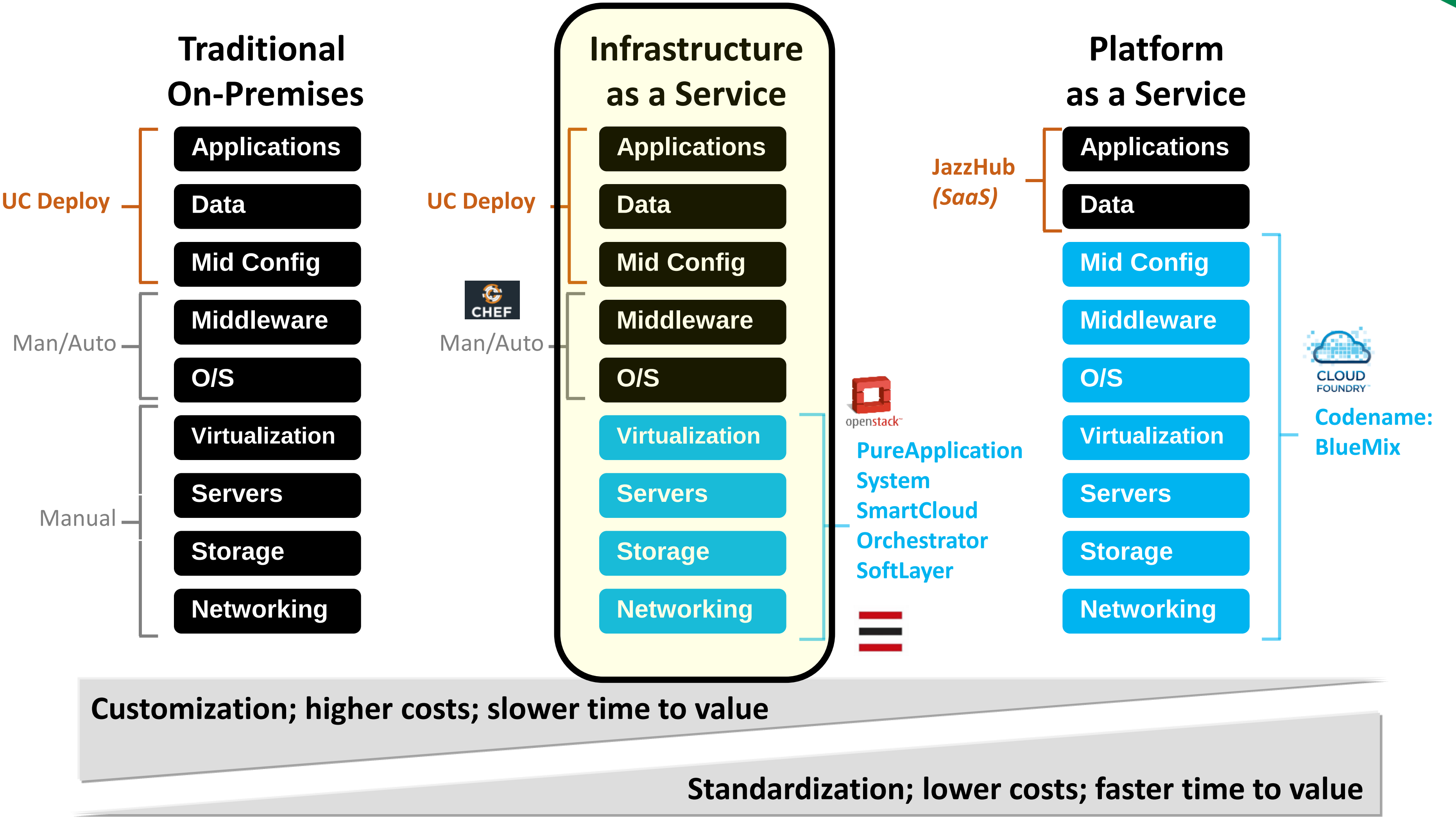
Rational Test Virtualization Server

InfoSphere Optim Test Data Management

SmartCloud Application Performance Management

DevOps and Cloud adoption

Automating for faster delivery with DevOps and cloud



Traditional On-Premises

UC Deploy: Applications, Data, Mid Config

Man/Auto: Middleware, O/S

Manual: Virtualization, Servers, Storage, Networking

Infrastructure as a Service

UC Deploy: Applications, Data, Mid Config

CHEF

Man/Auto: Middleware, O/S

Virtualization, Servers, Storage, Networking

openstack

PureApplication System
SmartCloud Orchestrator
SoftLayer

Platform as a Service

JazzHub *(SaaS)*: Applications, Data

Mid Config, Middleware, O/S, Virtualization, Servers, Storage, Networking

CLOUD FOUNDRY

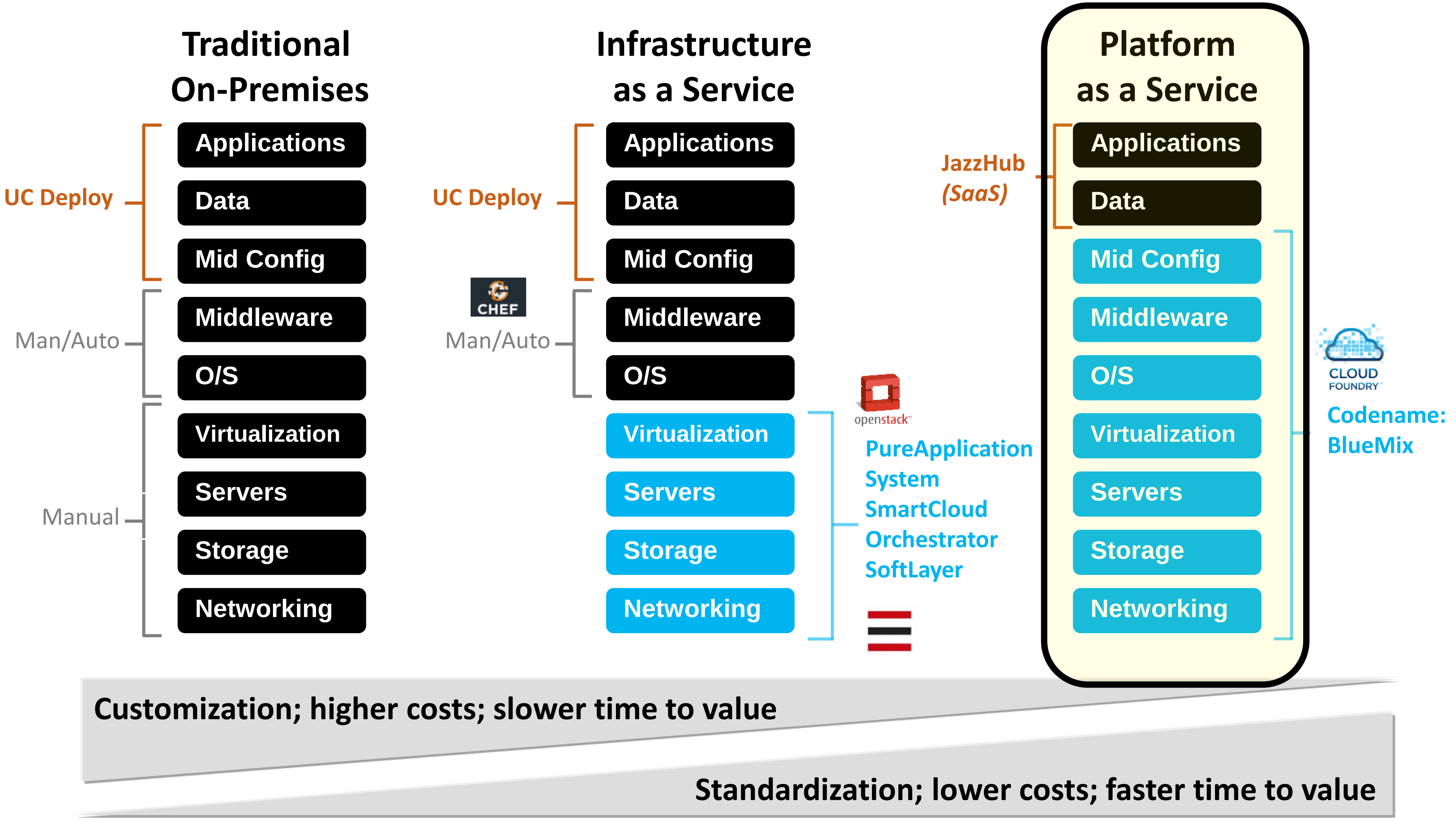
Codename: BlueMix

Customization; higher costs; slower time to value

Standardization; lower costs; faster time to value

DevOps and Cloud adoption

Automating for faster delivery with DevOps and cloud



Traditional On-Premises

UC Deploy – Applications, Data, Mid Config

Man/Auto – Middleware, O/S

Manual – Virtualization, Servers, Storage, Networking

Infrastructure as a Service

UC Deploy – Applications, Data, Mid Config

CHEF

Man/Auto – Middleware, O/S

Virtualization, Servers, Storage, Networking

openstack

PureApplication System

SmartCloud Orchestrator

SoftLayer

Platform as a Service

JazzHub *(SaaS)* – Applications, Data

Mid Config, Middleware, O/S, Virtualization, Servers, Storage, Networking

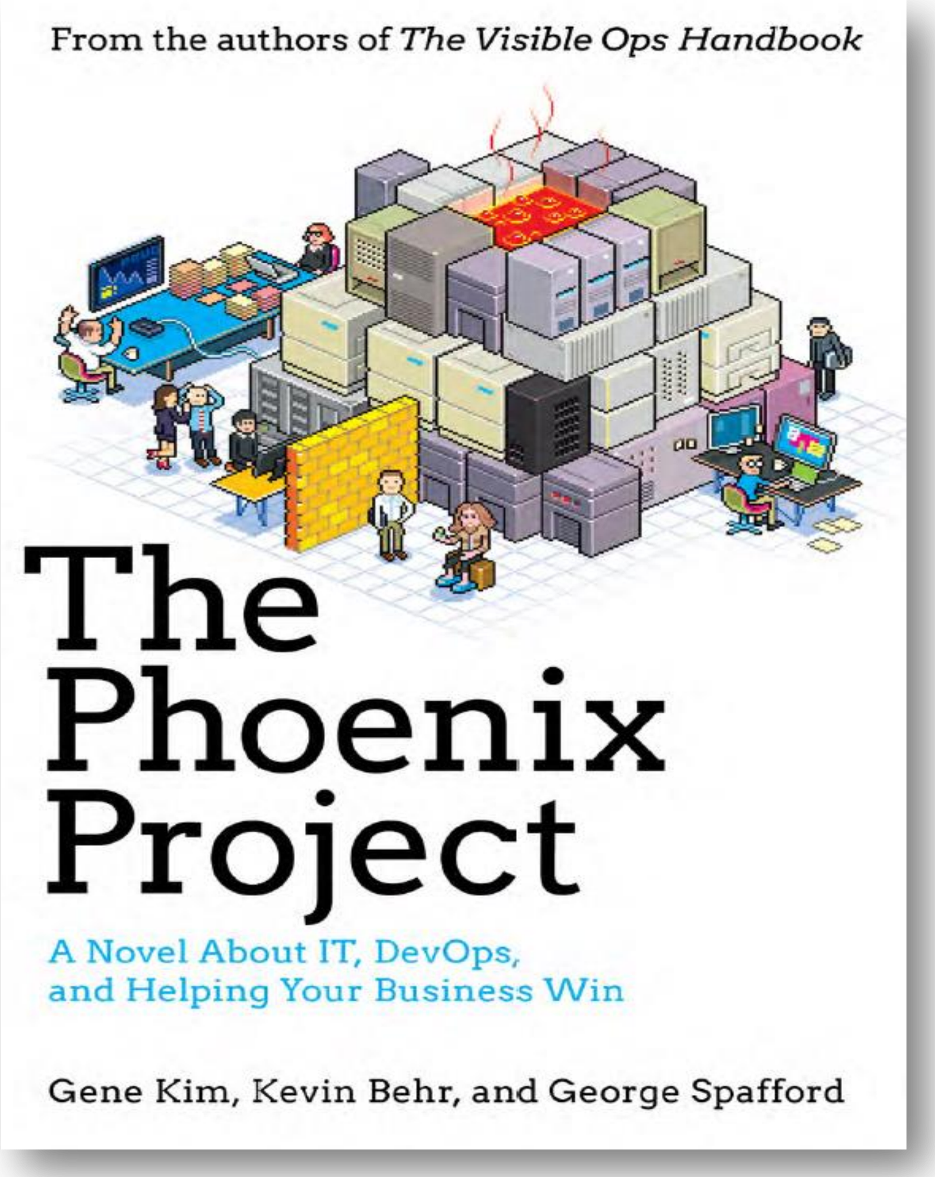
CLOUD FOUNDRY

Codename: BlueMix

Customization; higher costs; slower time to value

Standardization; lower costs; faster time to value

Recommended Reading



From the authors of *The Visible Ops Handbook*

The Phoenix Project

A Novel About IT, DevOps, and Helping Your Business Win

Gene Kim, Kevin Behr, and George Spafford

A parable about how a fictional auto-parts manufacturer manages to avoid corporate disaster by applying DevOps principles to regain the agility to achieve market leadership.

Thank you!