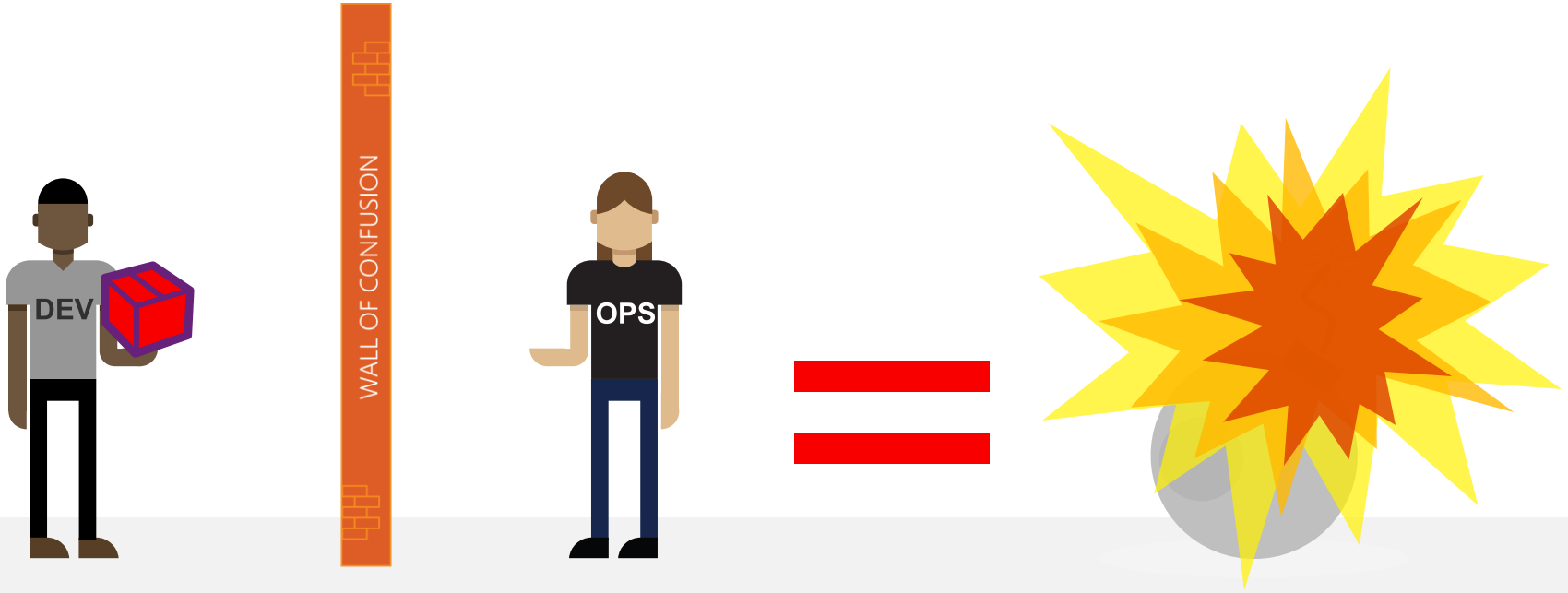
DevOPS

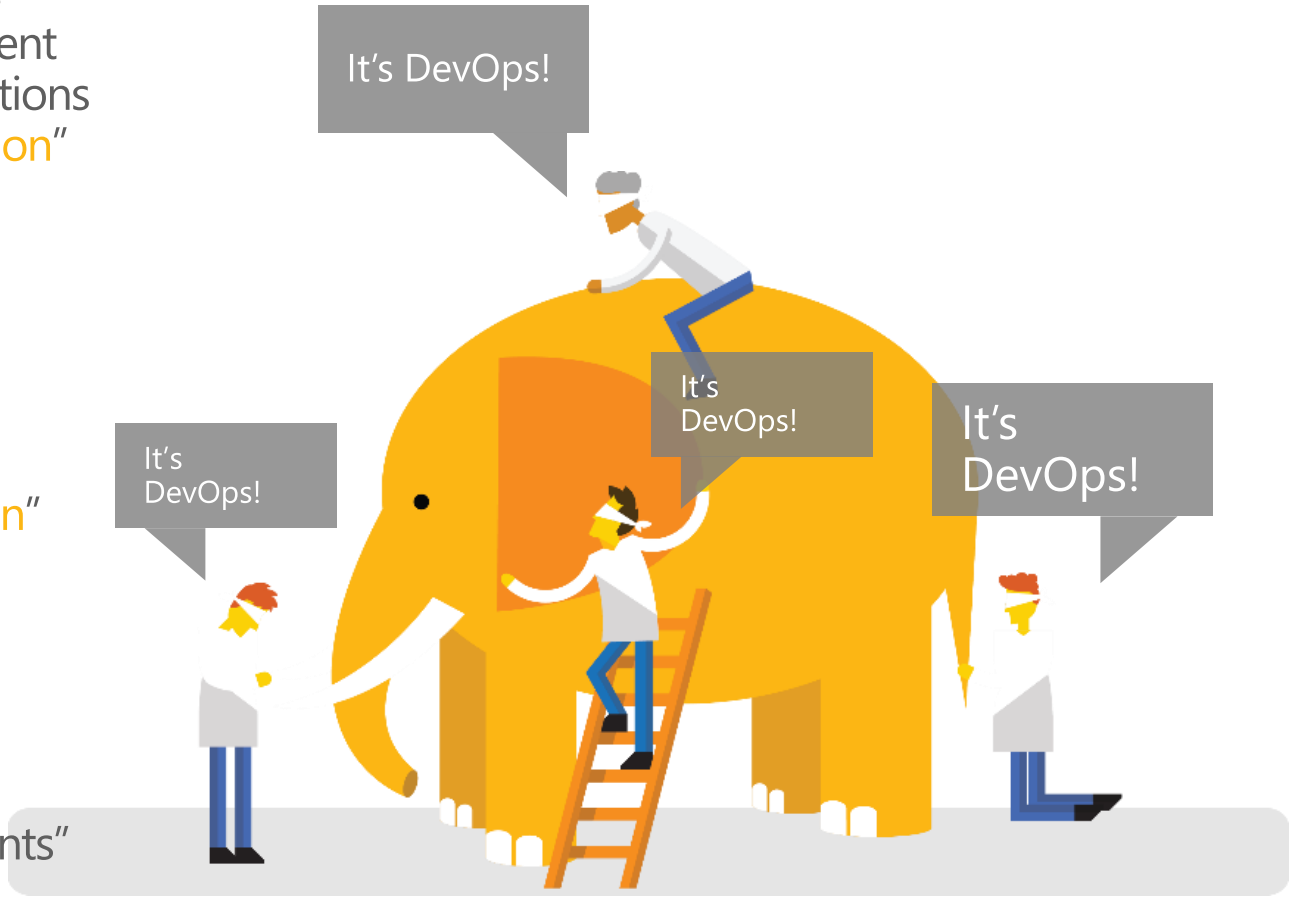
Traditional Development and Operations



"**DevOps** is development and operations collaboration"

"**DevOps** is using automation"

"**DevOps** is small deployments"

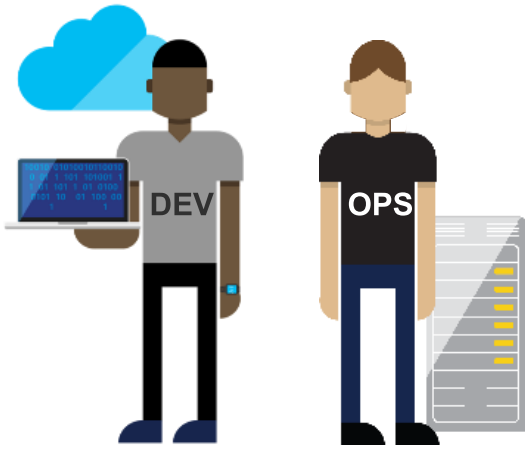


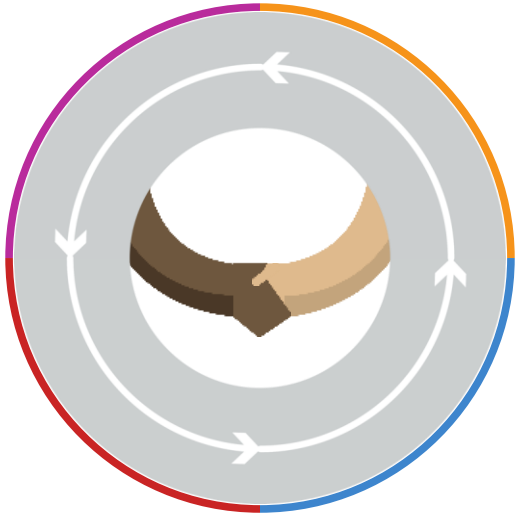
"**DevOps** is treating your infrastructure as code"

"**DevOps** is feature switches"

"Kanban for Ops?"

DevOps: the three stage conversation



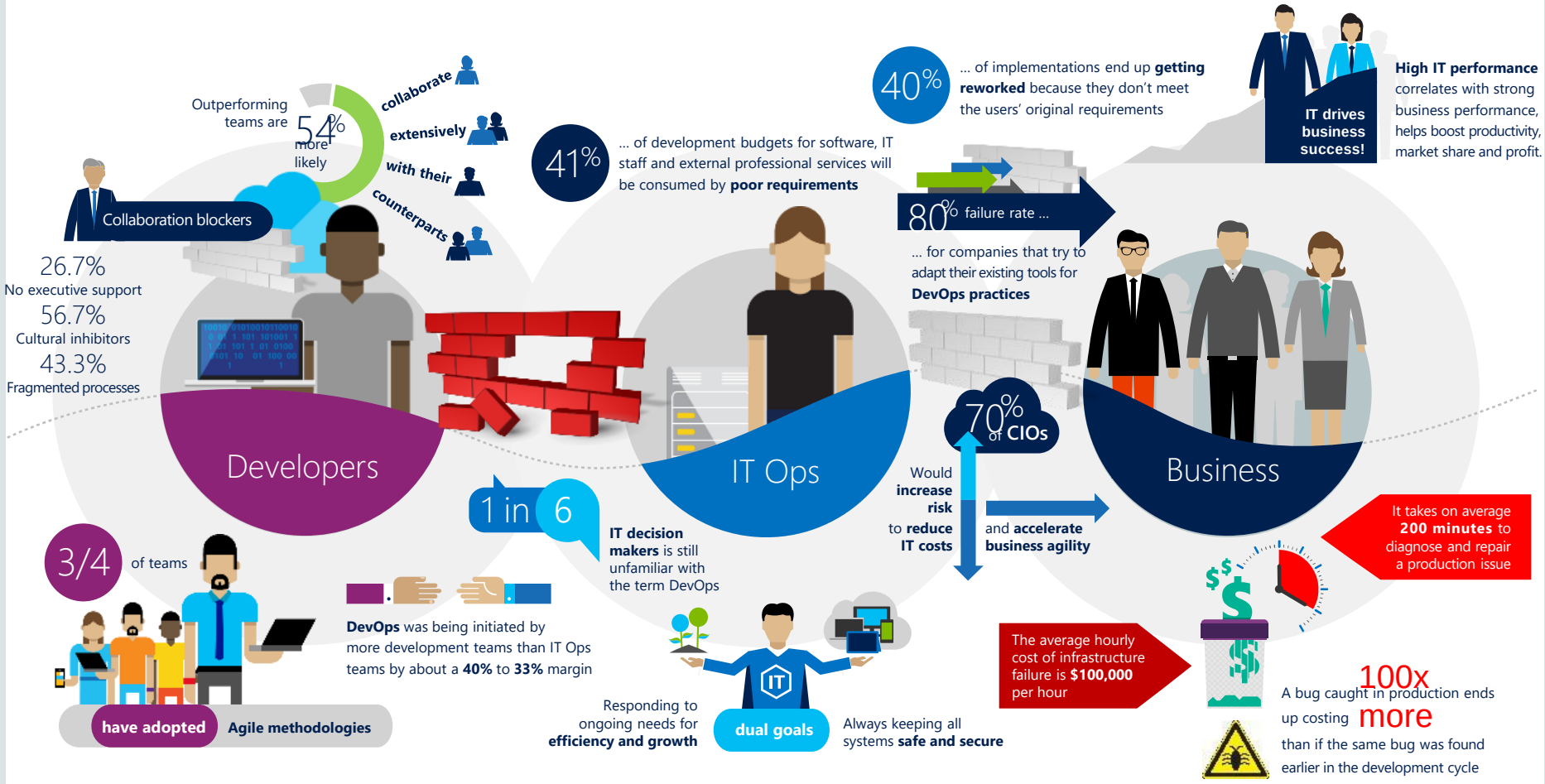


1	People

2	Process

3	Products

The consequences of inefficiency



DevOps Benefits

Strong IT Performance is a competitive advantage

Firms with high-performing IT organizations were 2x as likely to exceed their profitability, market share, and productivity goals

DevOps Practices improve IT performance

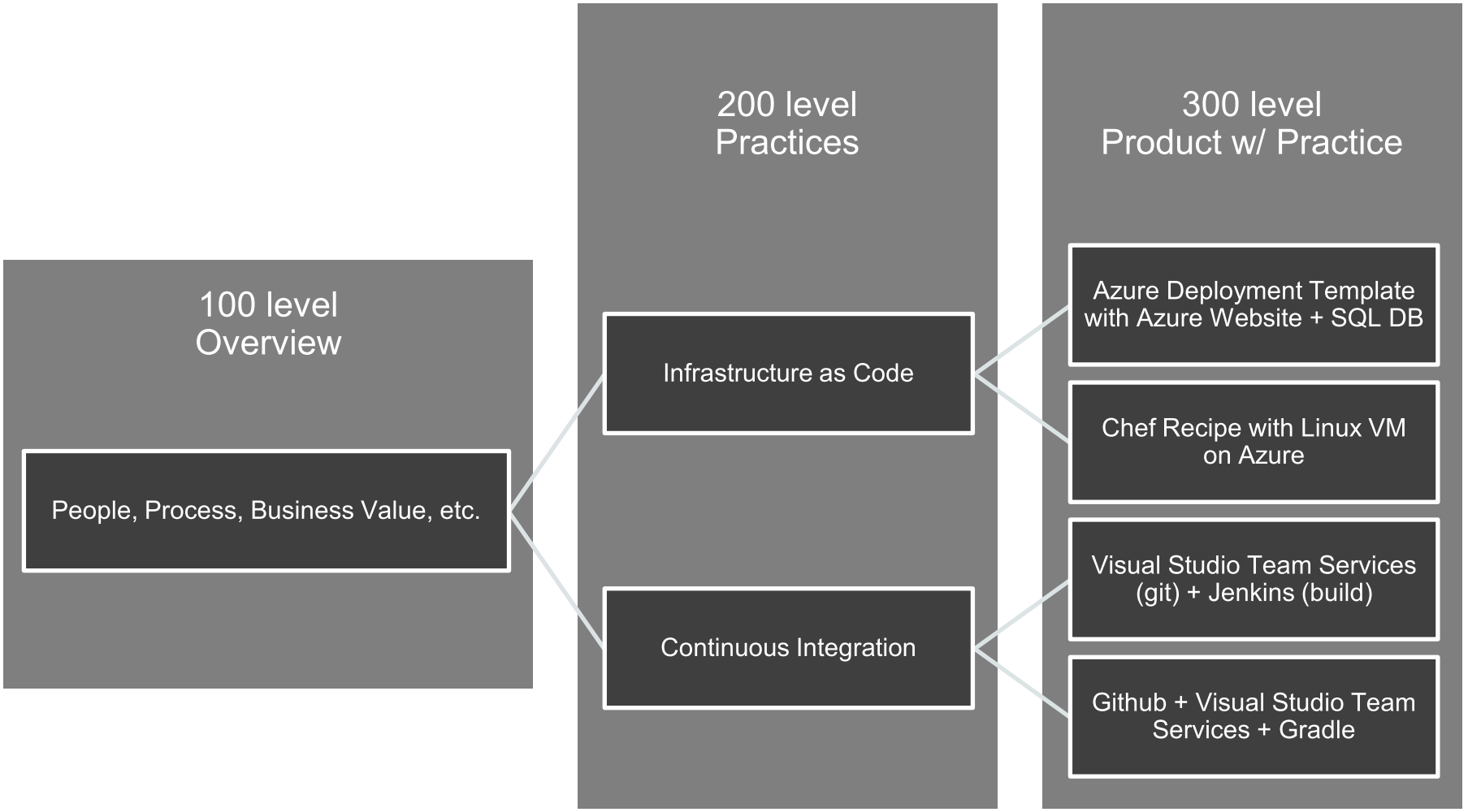
Deploy code 30x faster

and with 200x shorter lead time as compared to their lower-performing peers

Have 60x fewer failures

and recover from failure 168x faster as compared to their lower-performing peers

DevOps Frame



List of DevOps Practices

- Infrastructure as Code (IaC)
- Continuous Integration
- Automated Testing
- Continuous Deployment
- Release Management
- App Performance Monitoring
- Load Testing & Auto-Scale

- Availability Monitoring
- Change/Configuration Management
- Feature Flags
- Automated Environment De-Provisioning
- Self Service Environments
- Automated Recovery (Rollback & Roll-Forward)
- Hypothesis Driven Development
  - Testing in Production
  - Fault Injection
  - Usage Monitoring/User Telemetry

http://www.itproguy.com/devops-practices/

Every part of the development cycle can be automated in some way

Asset compiling

Test running

Package building

Server deployments

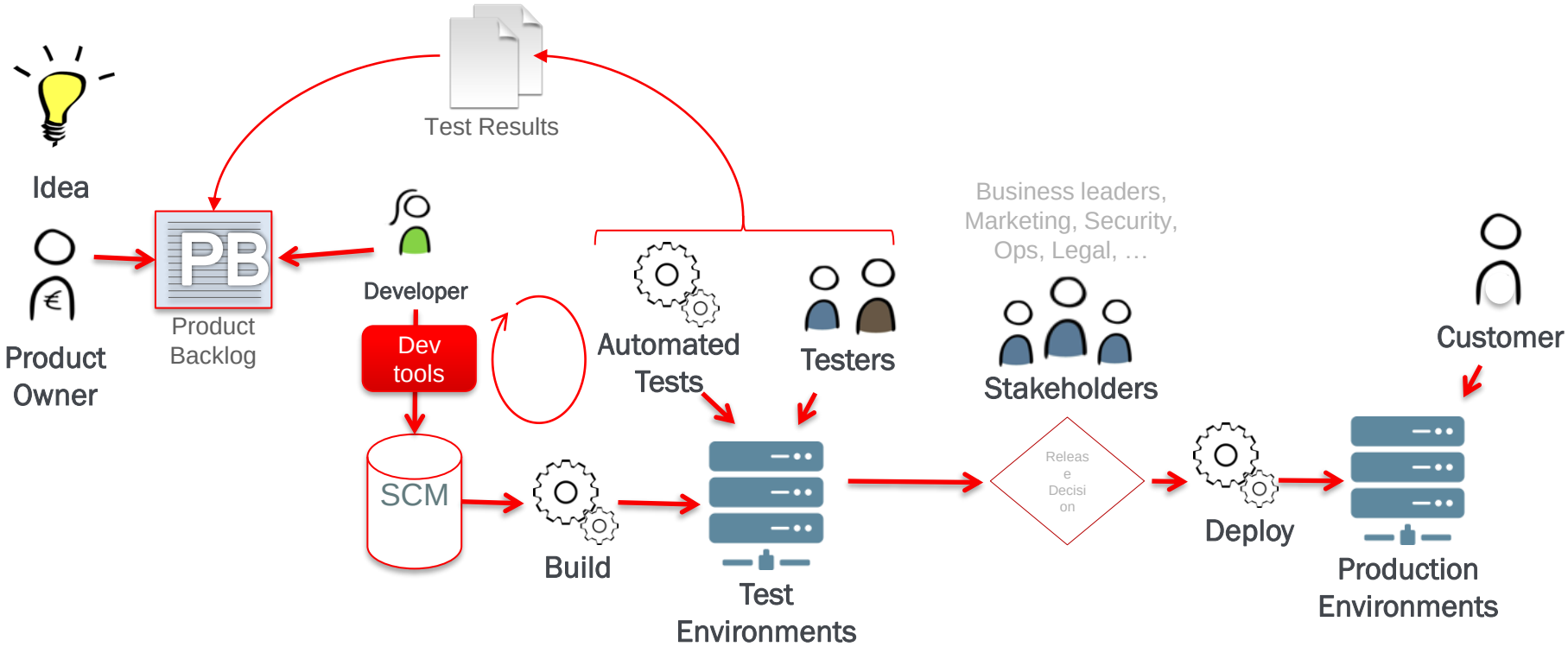
Smoke Testing

Auto-scaling

Health reporting

……

Every part of the development cycle = Processes



Automate Processes supports competency

Continuous Delivery Key Concepts

Continuous Delivery Key Concepts

Ensure a blameless culture

Create a Culture that foster

- Set metrics and measure your success

Visualize your Pipeline

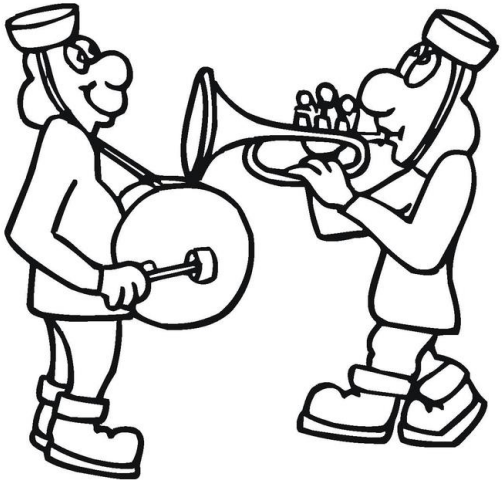
- Where you are
- Where you want to be

Ensure a blameless culture

One of the most important tools of DevOps: Failure

Getting from:

To:



Failure is not a cause for blame, it is a vehicle for change, learning, and improvement.

The Three Ways

The First Way – Flow

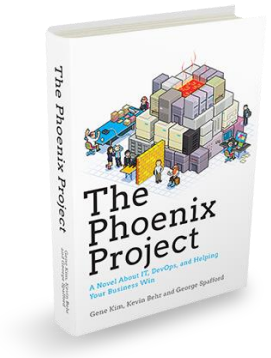
- Increase the flow of work (left to right)

The Second Way – Feedback

- Shorten feedback loops for continuous improvement (right to left)

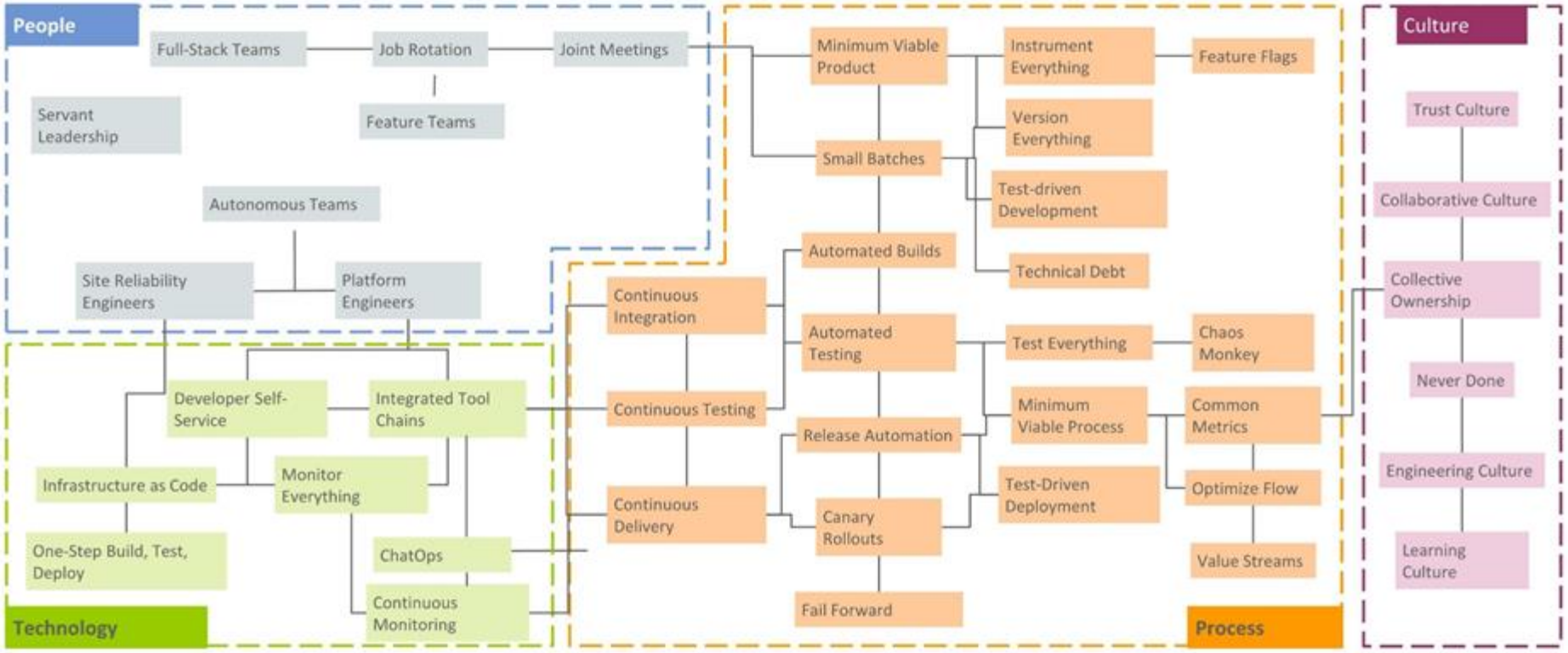
The Third Way – Continuous experimentation and learning

- Create a culture that fosters
  - Experimentation, taking risks and learning from failure
  - Understanding that repetition and practice leads to mastery



The Theory of Constraints is an important element of the Three Ways.

Gartner DevOps Model



Starting point of DevOps could be from any of the categories: *People*, *Process*, *Culture* or *Technology*.

Continuous Delivery Automation Concepts

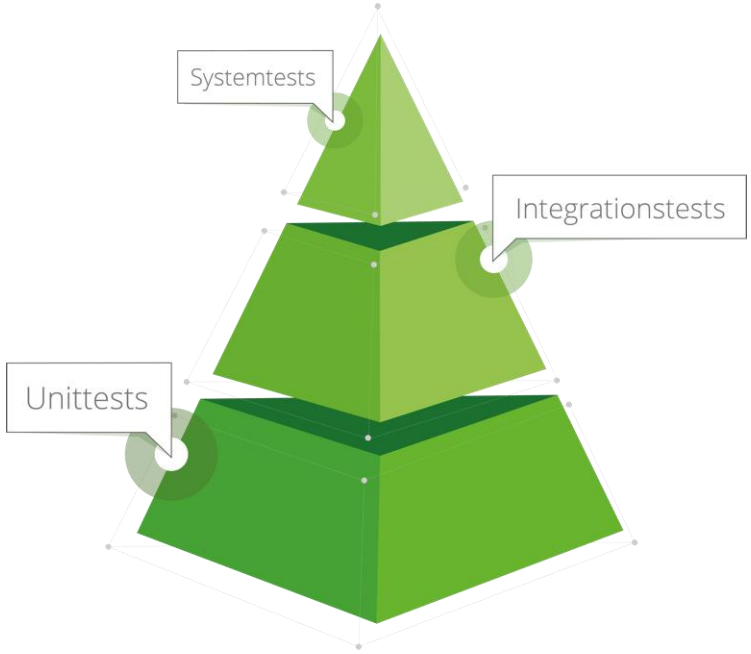
Increase Automation

Automate as much as possible

Use test automation to enable quicker, cheaper releases

Use model driven development to ensure consistent, faster releases

If you have to do the same thing over and over again, step back
and automate



Sharpening the Axe

- Employ toolsets
- Automate processes
- Increase confidence
- Improve visibility
- Utilize retrospectives
- Prepare for change

Employ Toolsets

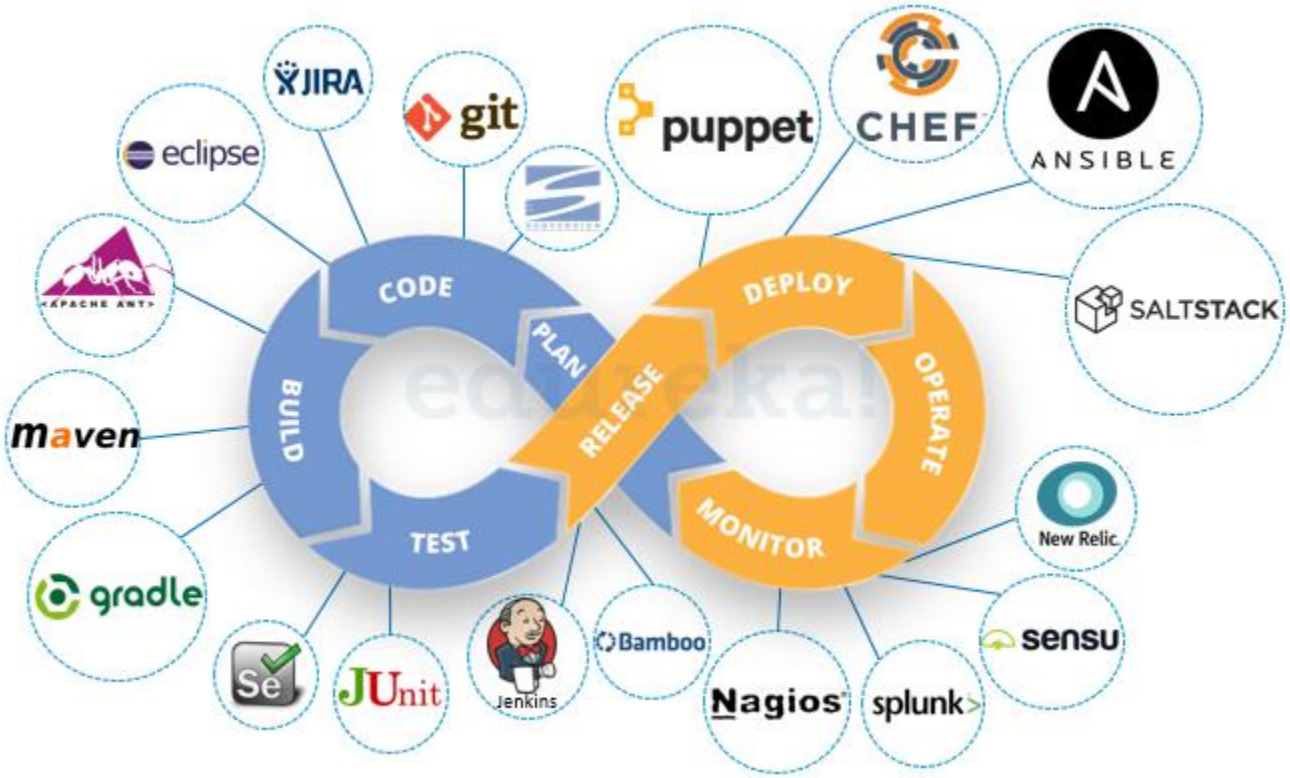
- Tools are a huge part of successful DevOps
  - **Dev Tools:** Linters, Virtual Environments
  - **QA Tools:** Static/Dynamic Code Analysis
  - **Ops Tools:** Job Runners, Notifications

- Every tool can provide some benefit to all aspects of of the development cycle

Automation in software delivery

Tools

Tools – think, analyze and investment

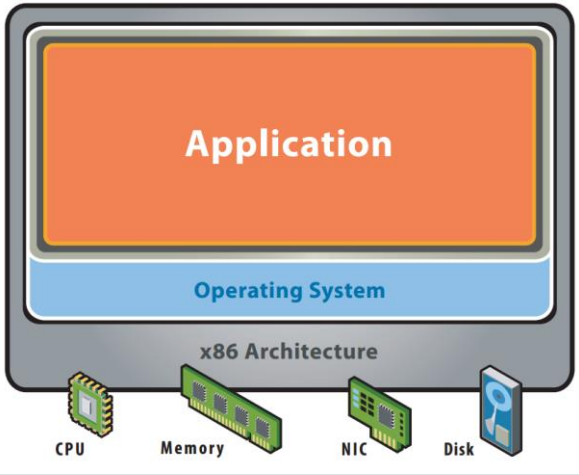


Data Center Automation

Virtualization

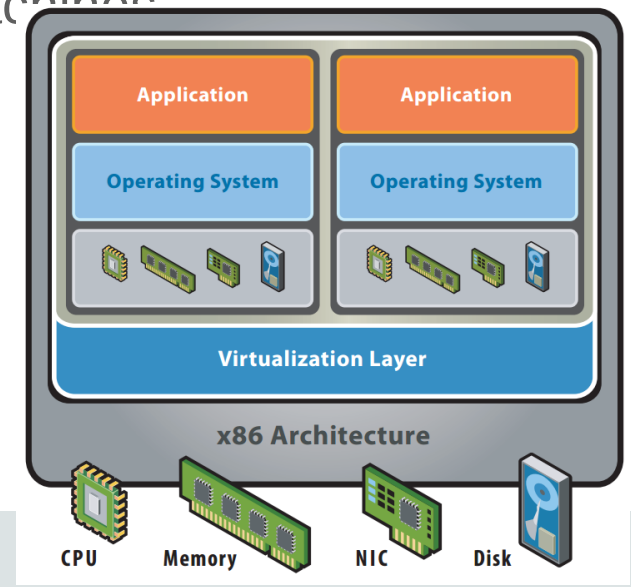
Before Virtualization:

- Single OS image per machine
- Software and hardware tightly coupled
- Running multiple applications on same machine often creates conflict
- Underutilized resources
- Inflexible and costly infrastructure



After Virtualization:

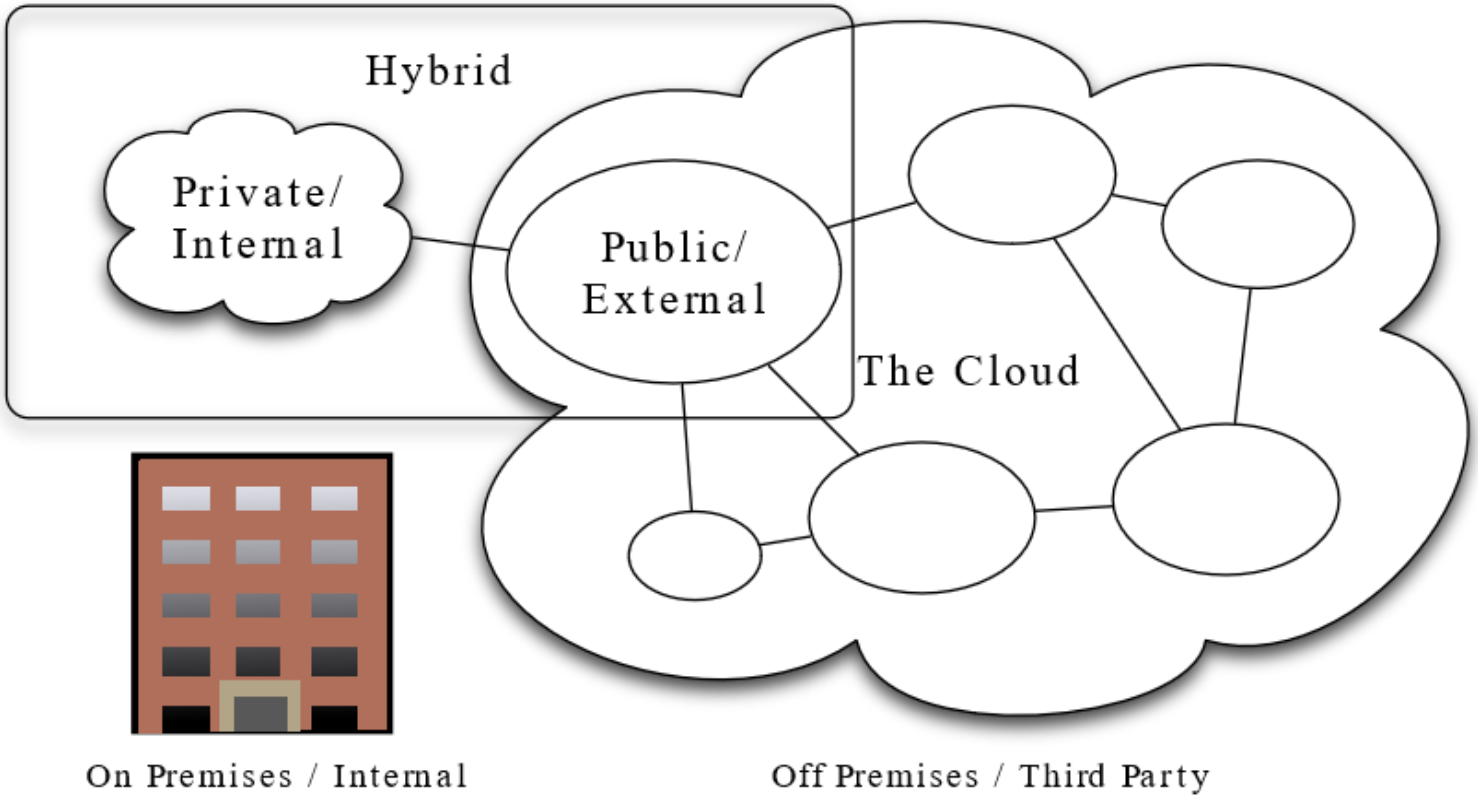
- Hardware-independence of operating system and applications
- Virtual machines can be provisioned to any system
- Can manage OS and application as a single unit by encapsulating them into virtual machines



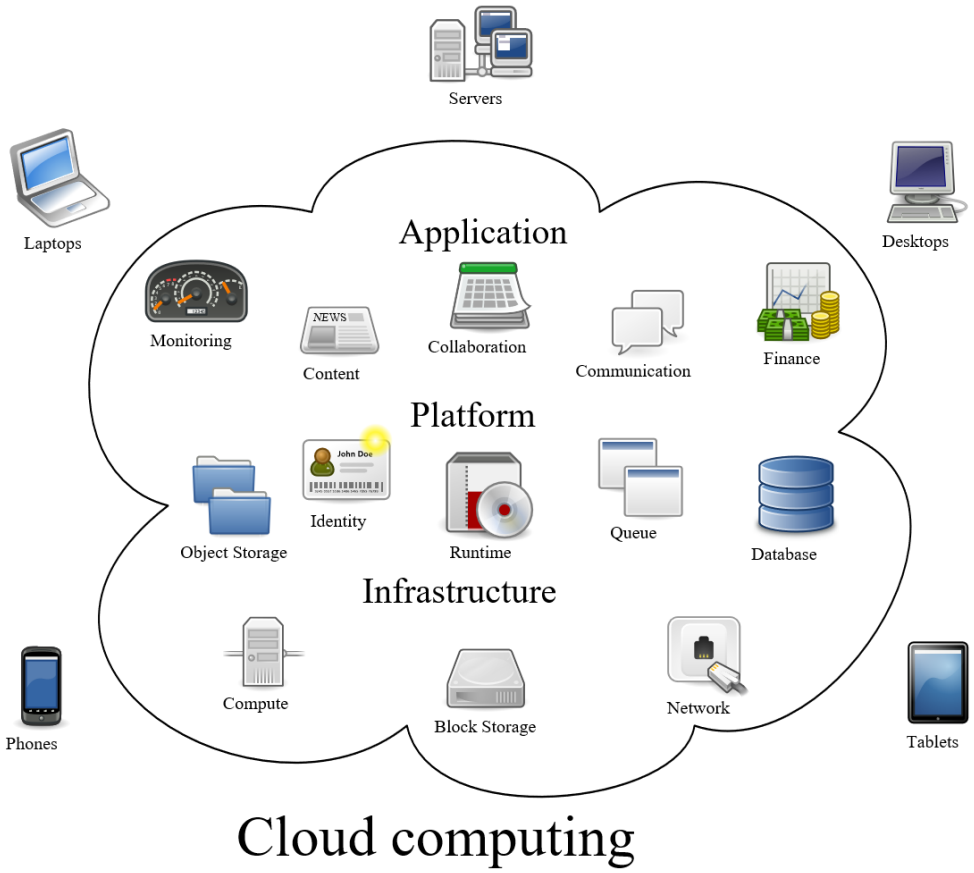
Principles and History of the Cloud

Key Cloud Concepts

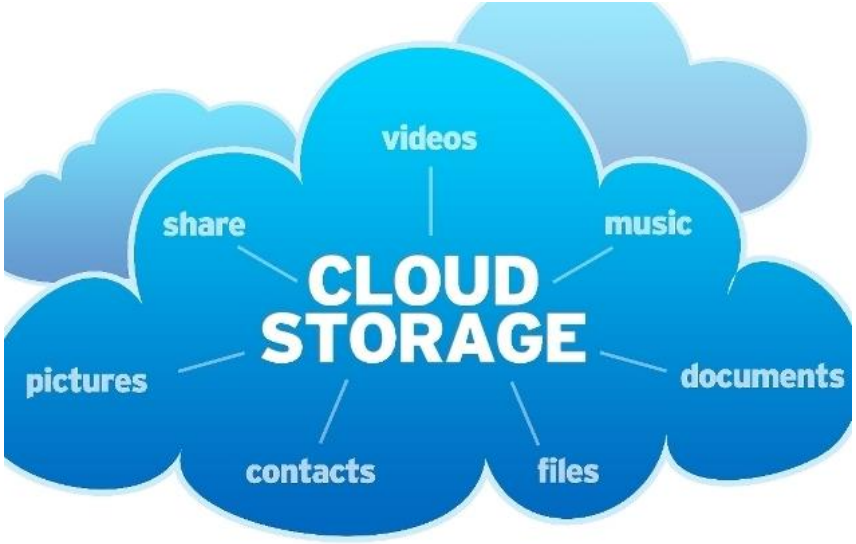
Cloud Deployment Models: Public, Private and Hybrid Cloud

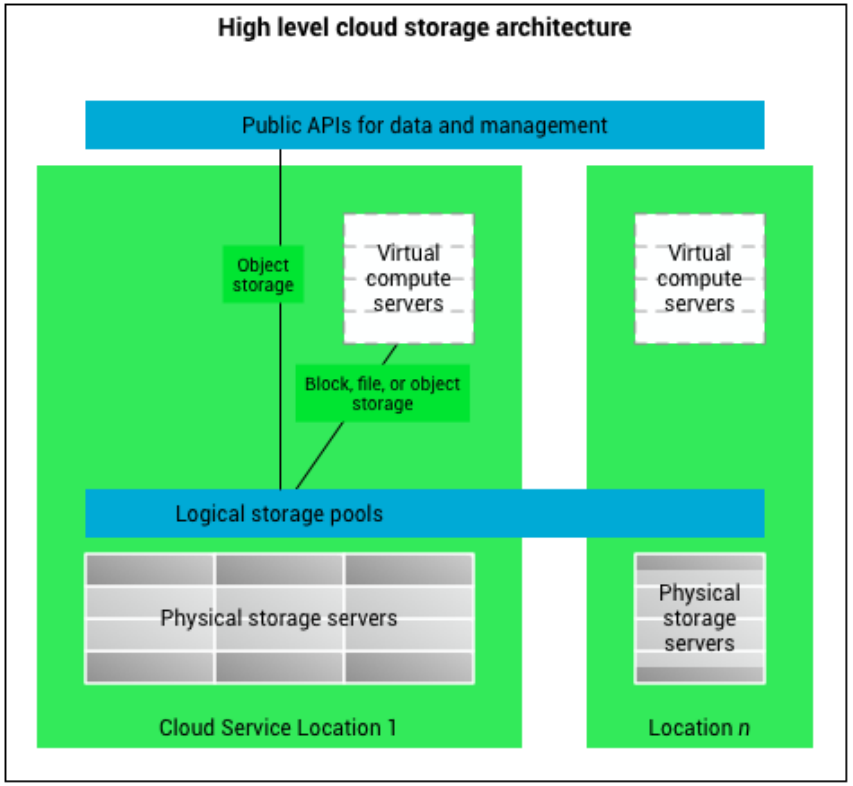


Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS)



Different Cloud Storage Types





Automated Provisioning

Automated provisioning, also called self-service provisioning, is the ability to deploy an information technology or telecommunications service by using pre-defined procedures that are carried out electronically without requiring human intervention.

Cut provisioning time to minutes

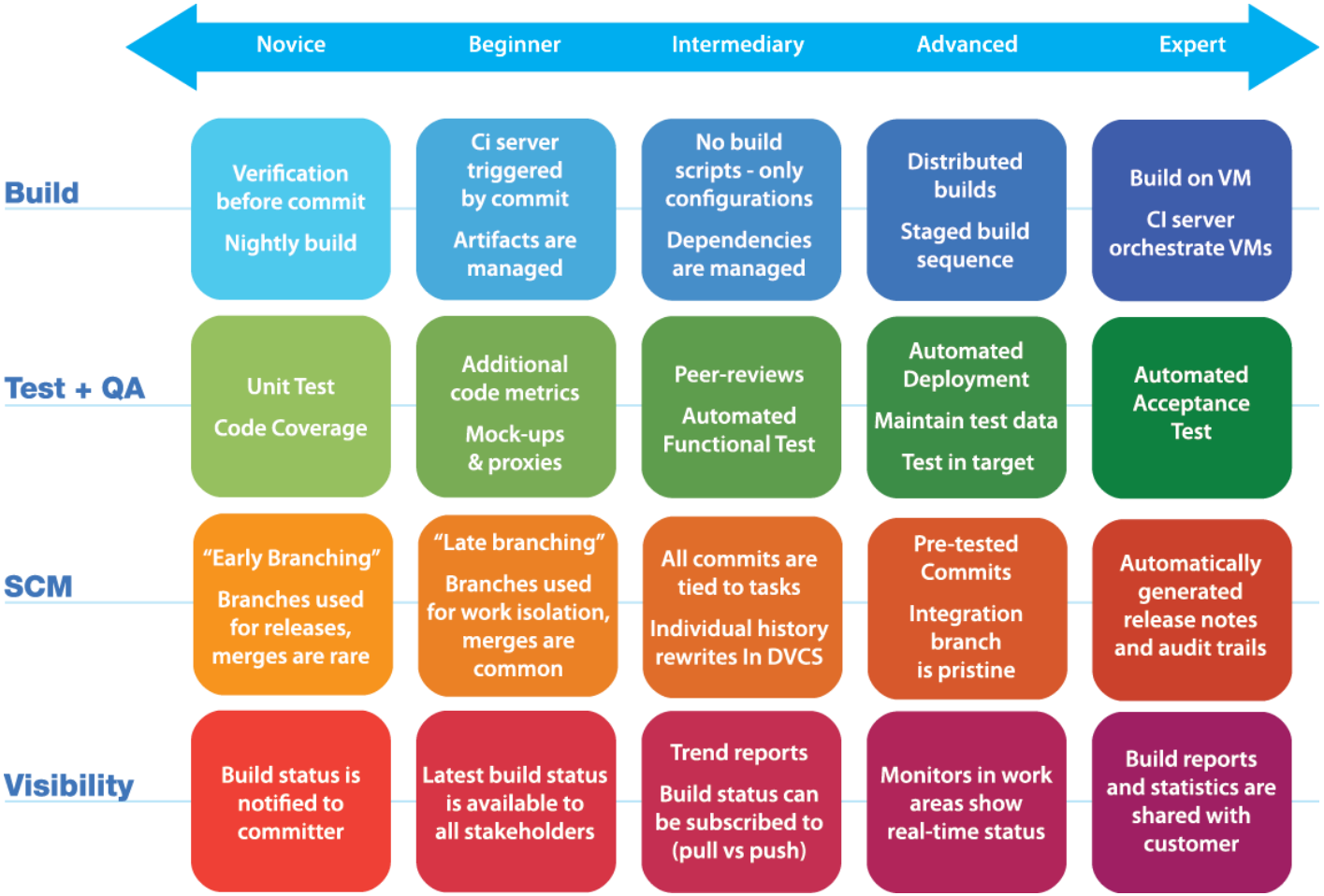
Accelerate bare metal and VM provisioning

Adopt cloud infrastructure and add capacity quickly

Easily launch and install Docker containers

Platform features and application maturity

Continuous Delivery Maturity Matrix

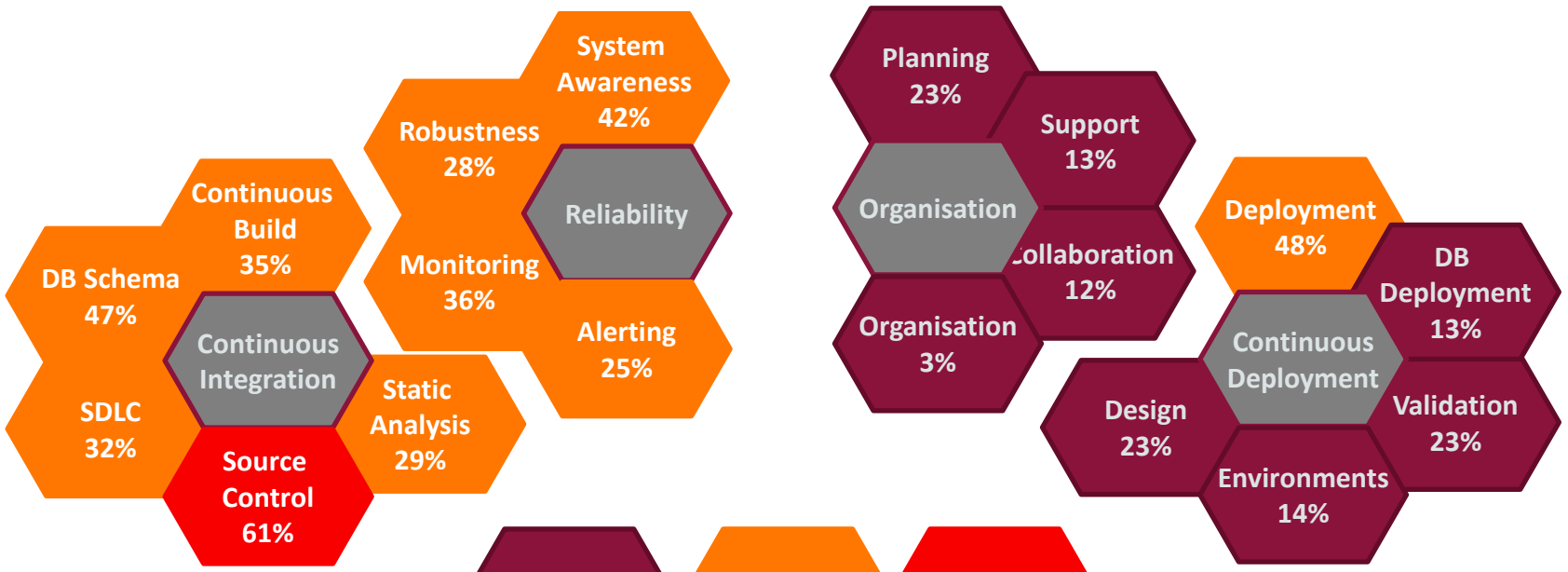


Platform features and application maturity

Table. Cloud application maturity level.

Maturity Level	Description
Level 3: Adaptive	• Application can dynamically migrate across infrastructure providers without interruption of service. • Application can elastically scale out/in appropriately based on stimuli.
Level 2: Abstracted	• Services are stateless. • Application is unaware and unaffected by failure of dependent services. • Application is infrastructure agnostic and can run anywhere.
Level 1: Loosely Coupled	• Application is composed of loosely coupled services. • Application services are discoverable by name. • Application compute and storage are separated. • Application consumers one or more cloud services: compute, storage, network.
Level 0: Virtualized	• Application runs on virtualized infrastructure. • Application can be instantiated from an image or script.

OVERALL MATURITY



-29%

The importance of monitoring indicators

Your organization is now committed to DevOps methodology.

1. How do you find out how well it works?
2. How do you know if it's working at all?

Many new DevOps organizations are surprised to find that it was easier to start then they thought, but hard to keep alive.

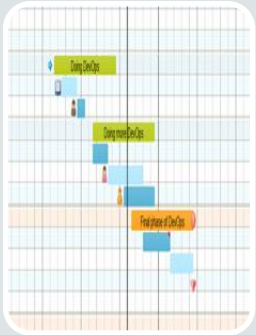
- This can be because of menacing habits, or just because the team is too far into the weeds.

In a word, they need metrics!

Choose the right metrics

First: Measure Outcomes. Measure Direct Evidence.

More DevOps Metrics That Matter



Culture

- Retention
- Satisfaction
- Callouts

Process

- Idea-to-cash
- MTTR
- Deliver time

Quality

- Tests passed
- Tests failed
- Best/worst

Systems

- Throughput
- Uptime
- Build times

Activity

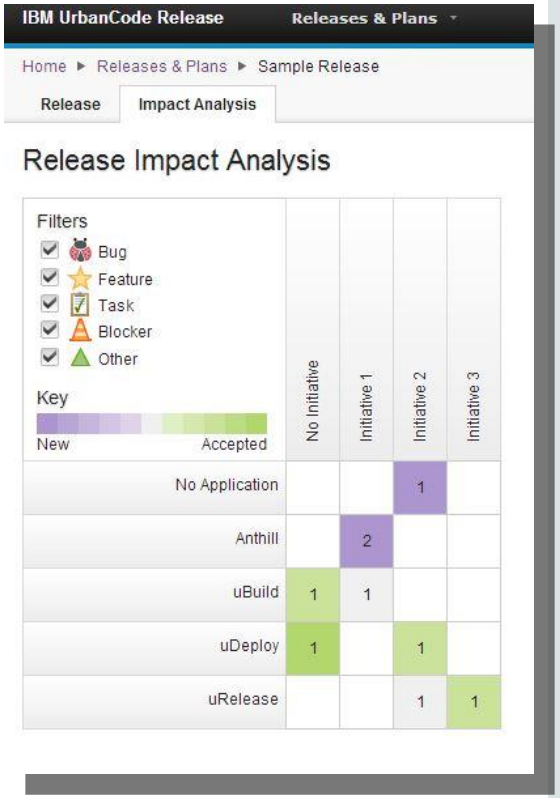
- Commits
- Tests run
- Releases

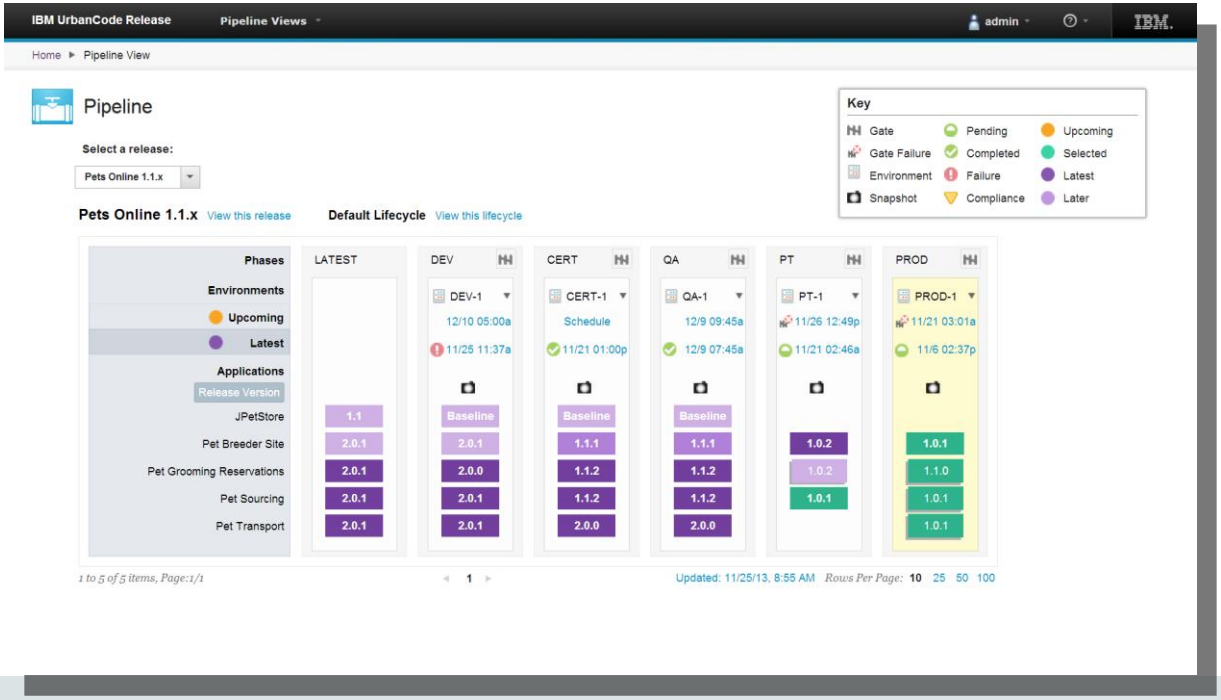
Impact

- Signups
- Checkouts
- Revenue

Monitoring, recording and Analysis

- Dashboards
  - Change Management
  - Mapping
  - RTC Delegated UI Dialogs
  - Release Progression
- Release Impact Analyses
- Pipeline View
- Federated Deployment Dashboard






Trogdor
ONTASK
DASHBOARD
72° Clear | Feb 6, 2017 • 12:53PM
PRODUCTION AGE
3 days
February 3rd, 2017
APP - PRODUCTION
Healthy
2/2 Instances
CAMUNDA - PRODUCTION
Healthy
2/2 Instances
MICROSERVICE BUILDS
Passing
17/17 Builds
OPEN MERGE REQUESTS
6 Open
3 Repositories
AUTOMATED TESTS
Running...
20% Complete
431 Passing • 0 Failing
APP - QA
Healthy
1/1 Instances
CAMUNDA - QA
Healthy
1/1 Instances
NODE MODULE BUILDS
Passing
8/8 Builds
SPRINT GOALS
In Progress
Trogdor 2017-3
0 In Danger • 0 Failed

Tools

Tools – Test First & Test Automation

Tools – Version Control

Tools – Build


Maven
<APACHE ANT>
Gradle
Gulp
GRUNT
Make:
MS Build
Bundler
Brunch
Nant

Tools – Continuous Integration

Jenkins

Atlassian Bamboo

TeamCity

cruisecontrol

Travis CI

circleci

Microsoft Visual Studio Team Foundation Server

CODESHIP

Tools – Continuous Deployment

Tools – Configuration Management

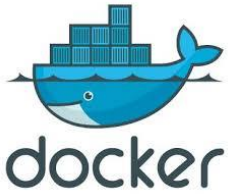
CHEF

ANSIBLE

SALTSTACK

CFEngine

Tools – Cloud, Virtualization and Container



Tools – Monitoring and Dashboard

Tools – Database Management

Sqitch

Tools – Code Quality