PeopleCert DevOps.

# PeopleCert DevOps Fundamentals

The Full Stack Of Culture, Practices & Automation

Study Guide

# Copyright Details



The contents of this workshop are protected by copyright and can be reproduced under the Terms of Use agreed between PeopleCert and the ATO using this material only.

Material in this presentation has been sourced from the bibliography listed in the certifications Syllabus.

No part of this document may be reproduced in any form without the written permission of PeopleCert International Ltd. Permission can be requested at www.peoplecert.org.

e-mail: info@peoplecert.org, www.peoplecert.org

**Copyright © 2017-2019 PeopleCert International Ltd.**
All rights reserved. No part of this publication may be reproduced or transmitted in any form and by any means (electronic, photocopying, recording or otherwise) except as permitted in writing by PeopleCert International Ltd. Enquiries for permission to reproduce, transmit or use for any purpose this material should be directed to the publisher.

**DISCLAIMER**
This publication is designed to provide helpful information to the reader. Although every care has been taken by PeopleCert International Ltd in the preparation of this publication, no representation or warranty (express or implied) is given by PeopleCert International Ltd. as publisher with respect as to the completeness, accuracy, reliability, suitability or availability of the information contained within it and neither shall PeopleCert International Ltd be responsible or liable for any loss or damage whatsoever (indicatively but not limited to, special, indirect, consequential) arising or resulting of virtue of information, instructions or advice contained within this publication.)

Document Version: PC-DOF 1.0 | September 2019

PeopleCert
All talents, certified.

# PeopleCert: A global leader in certification


- ✓ **Web** & **Paper based** exams in **25** languages
- ✓ **2,000 Accredited Training Organisations** worldwide
- ✓ Delivering exams across **200 countries every year**
- ✓ Comprehensive Portfolio of **500+ Exams** and Growing

AXELOS GLOBAL BEST PRACTICE · PeopleCert DevOps. · digital marketing institute · IASSC · ISACA · Language Cert · LEAN IT ASSOCIATION · Quality Software Development · ECDL

PeopleCert

All talents, certified.

# How to Use This Document

This document is your **PeopleCert DevOps Fundamentals Study Guide** to help you prepare for the **PeopleCert DevOps Fundamentals examination**.

It is meant to provide you with a clear outline of everything covered in the course presentation by your instructor that will be on the PeopleCert DevOps Fundamentals exam.

Your exam will be closed book. You will be given 60 minutes to complete it. It contains 40 multiple choice questions and to pass the exam you must achieve a grade of 70% or higher, or a minimum of 28/40 correct responses. For further details on your exam, including more information on question types and learning objectives, please refer to your course syllabus.

As you follow along, you may see that some material in the presentation is not replicated here. This study guide does not contain questions, activities, knowledge checks, or other material in the presentation that are facilitated by the instructor. It also does not contain content that is not examinable, but instead is designed to reinforce learning or add value to your course experience.

# Table of contents

# Course Contents

**Introduction:**

What is DevOps?

**Module 1:**

The Urgency For DevOps

**Module 2:**

Key DevOps Principles & Concepts

**Module 3:**

The Full Stack – People & Culture

**Module 4:**

The Full Stack – Processes & Practices

**Module 5:**

The Full Stack – Technology & Automation

**Wrap Up & Review**

# Agenda

| Module |
| --- |
| Getting Started and Introduction |
| Module 1: The Urgency for DevOps |
| Module 2: Key DevOps Principles & Concepts |
| Module 3: The Full Stack – People & Culture Layer |
| Module 4: The Full Stack – Processes & Practices Layer |
| Module 5: The Full Stack – Technology & Automation Layer |
| Wrap Up & Review |
| Questions |
| Exam Information & Exam |

# Introduction

- PeopleCert DevOps Fundamentals

What Is DevOps?

# The Key Concepts Of DevOps

**DevOps** is a coined noun used to describe an evolution of existing IT best practices from ITIL, Lean and Agile into a development and operations approach that supports automation and continuous delivery, and encourages a culture of collaboration and learning to help IT deliver business value better, faster and cheaper than ever before.

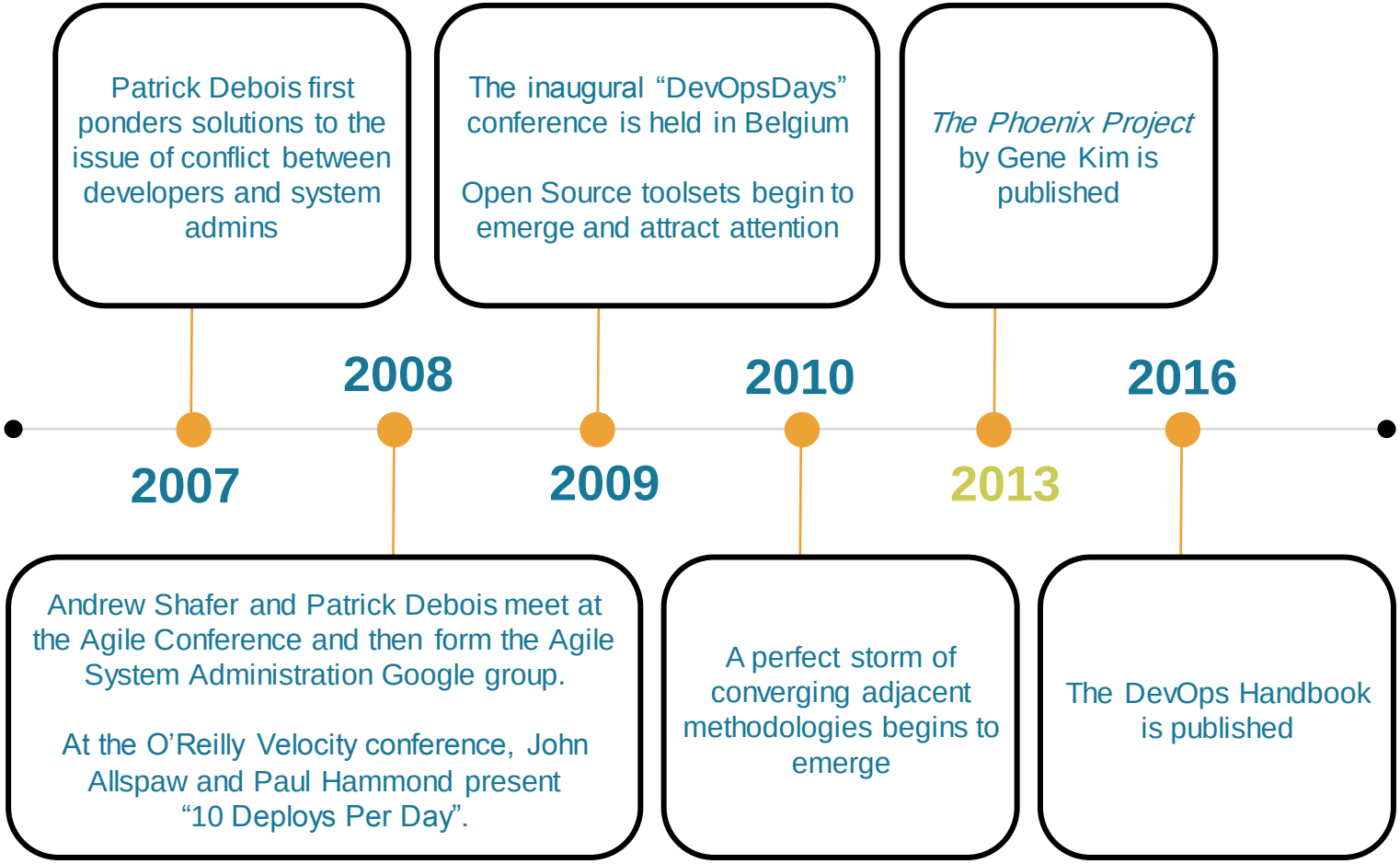
- **DevOps Is:**
  - A movement, philosophy or ideology that promotes a set of principles, practices and values
  - About the relationship between development and operations
  - A cultural shift that permeates throughout an entire organization
  - Focused on communication and collaboration
  - About automation and new tools to support that communication and collaboration
  - Concerned with balancing the speed and change required by Development with the reliability and stability required by Operations

**DevOps Is Not:**

- An adjective to describe a specific job title or type of team
- Only applicable to start-ups or web applications already suited to it
- A way to eliminate IT Operations
- A collection of tools that focus on automation without the corresponding cultural shift
- Lacking the flexibility required to be workable in a regulated environment

**For more information, check out this video:**
*The History of DevOps,* https://www.youtube.com/watch?v=o7-IuYS0iSE

# The History Of DevOps



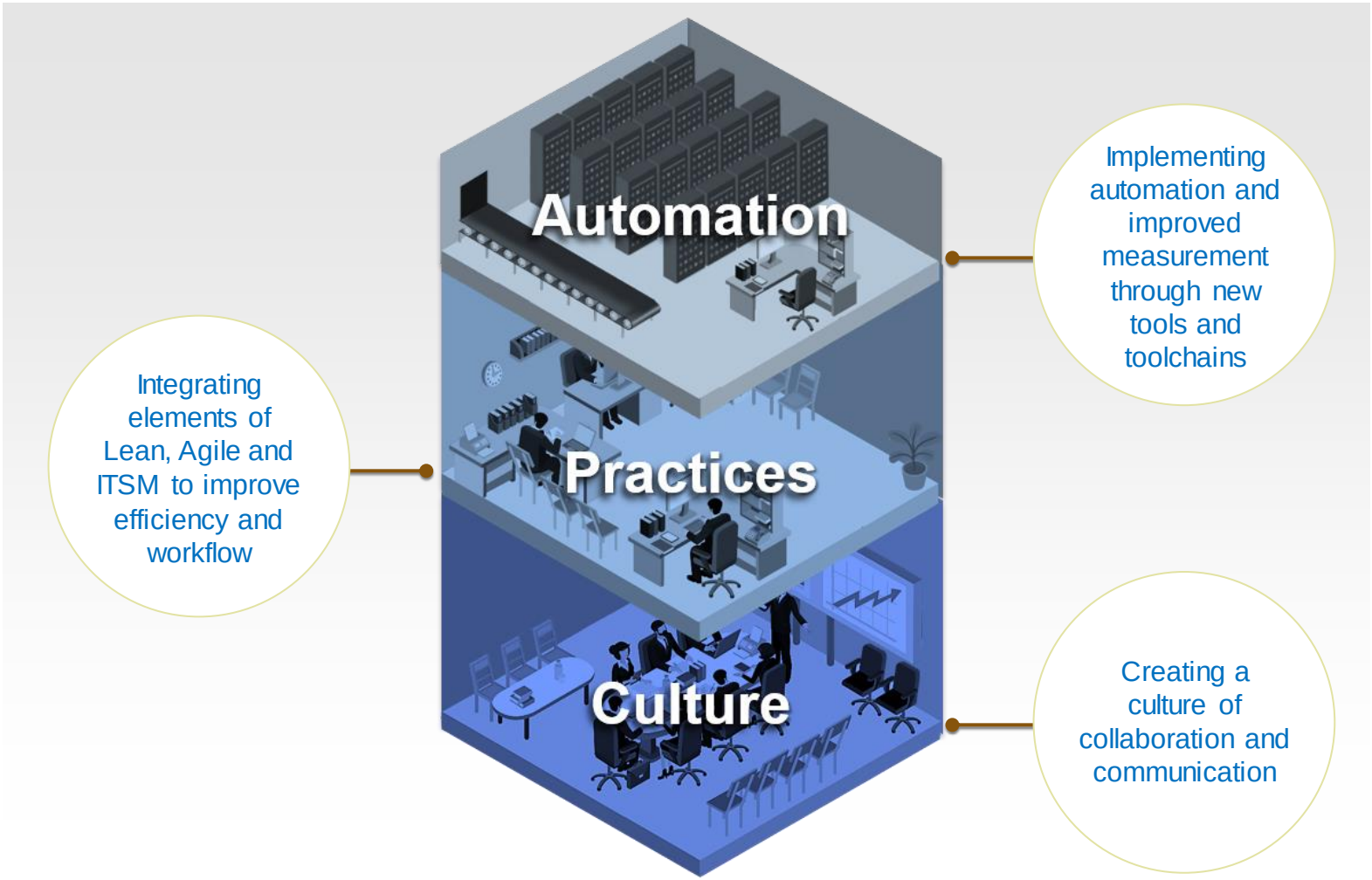
# DevOps Across Industries

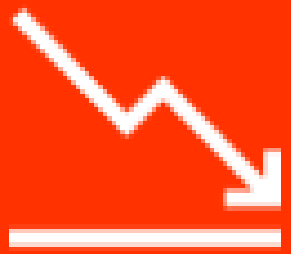

While web start-ups like Spotify, Netflix and Amazon were some of the obvious early pioneers of DevOps, it can be and has been implemented successfully at all kinds of organizations – large and small, flexible and rigidly regulated or structured, new and old, in all kinds of industries.

Retailers like Walmart, Target and Starbucks have embraced DevOps as a way to stay technically innovative and remain competitive. Schools like Stanford have turned to DevOps as a way to transform as online universities and extensive use of e-Learning become the norm. Even the U.S. government has made investments in DevOps, with U.S. Customs and Border Protection adopting it in addition to the Department of Justice and Federal Reserve! DevOps has even helped NASA to dream bigger and continue to work towards sending us to Mars and beyond. As DevOps is becoming more mainstream, the list of enterprises introducing DevOps programs has dramatically grown.

**For more information, check out this video:**
*What Is DevOps?,* https://www.youtube.com/watch?v=bYQhlpa9eoc

# The DevOps Full Stack





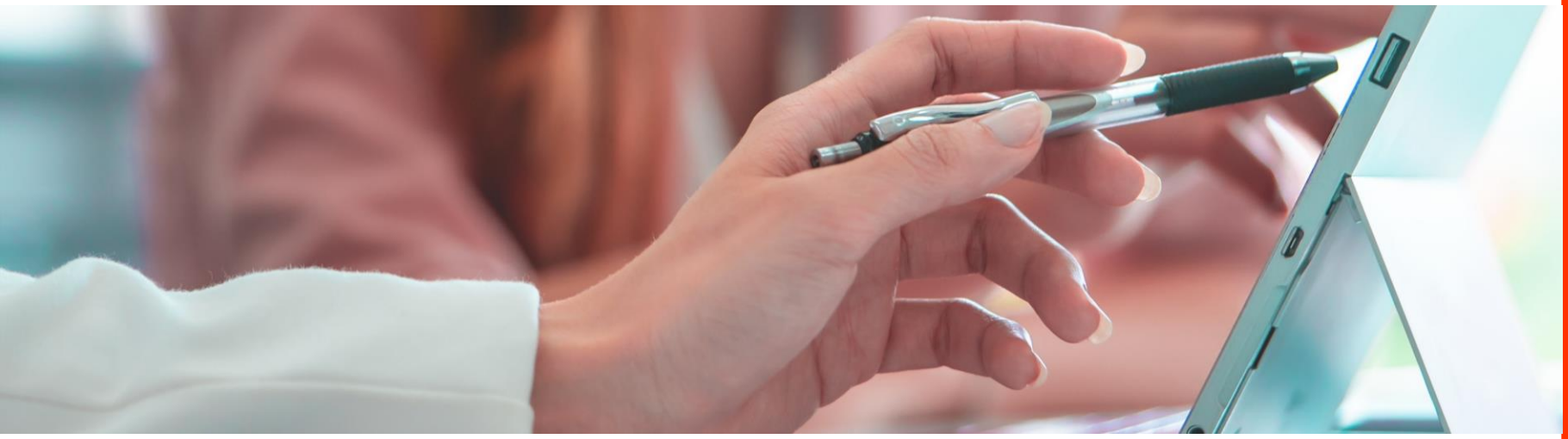
PeopleCert DevOps Fundamentals

# Module 1:
# **The Urgency For DevOps**

**Topics:**

- The Business Value Delivery Problem
- The IT Value Delivery Problem

# What Is Business Value?



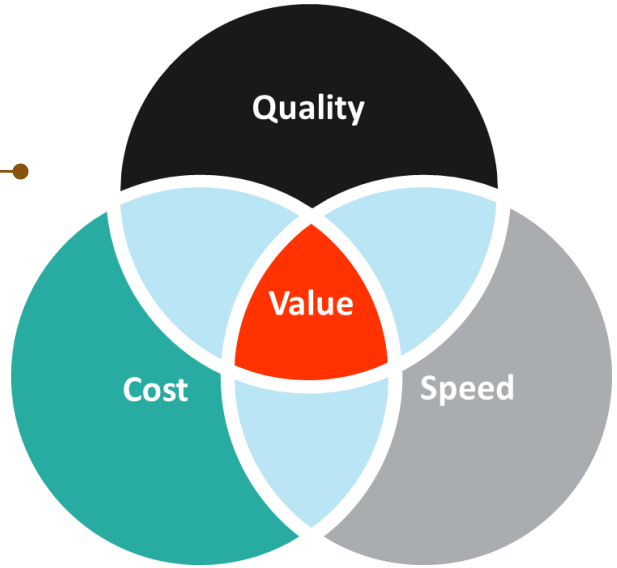
**Business Value** is the level to which a service meets a customer's expectations or exceeds them.

Business value has the customer at its core. It is the level to which the business meets their expectations.

However, the business creates value through services. In response to customer requirements, it sets the expectation. From the customer's perspective, value consists of the business achieving its own stated objectives, keeping its own promises.

Business value is also never static. It is always in flux, responding to the needs and expectations of customers as times change and as competition changes.

# Dimensions Of Business Value



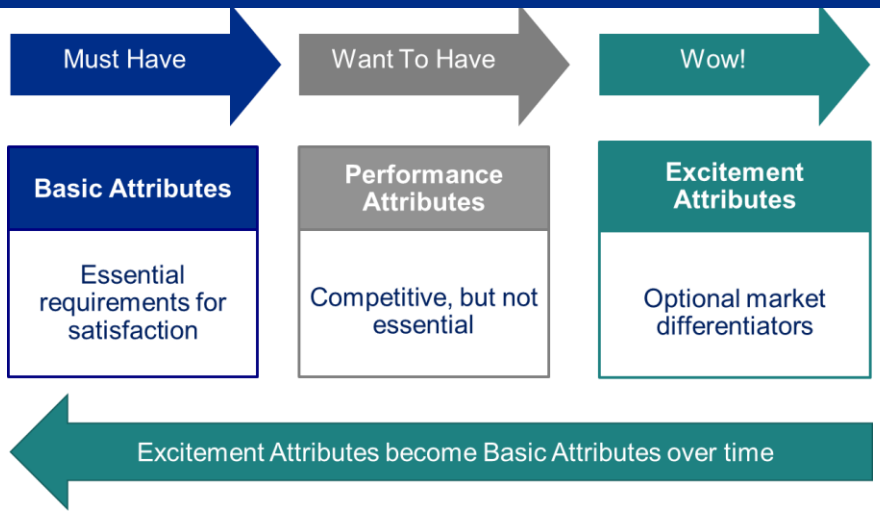

**Quality**
This is the customer's requirements and the warranty they have that the product will meet their needs.

**Cost**
This is often measured against how much a customer is willing to pay.

**Speed**
Customers need things when they need them. Delivering quality doesn't matter if it is delivered a day late.

# The Customer Value Hierarchy

**The Kano Model** defines attributes of customer value and states that they drift over time from those that are delighters to those that are performance needs until finally they become essential basic needs.




Understanding these customer value attributes helps us understand why IT is increasingly being asked by the business to deliver better, faster and cheaper to achieve business value.

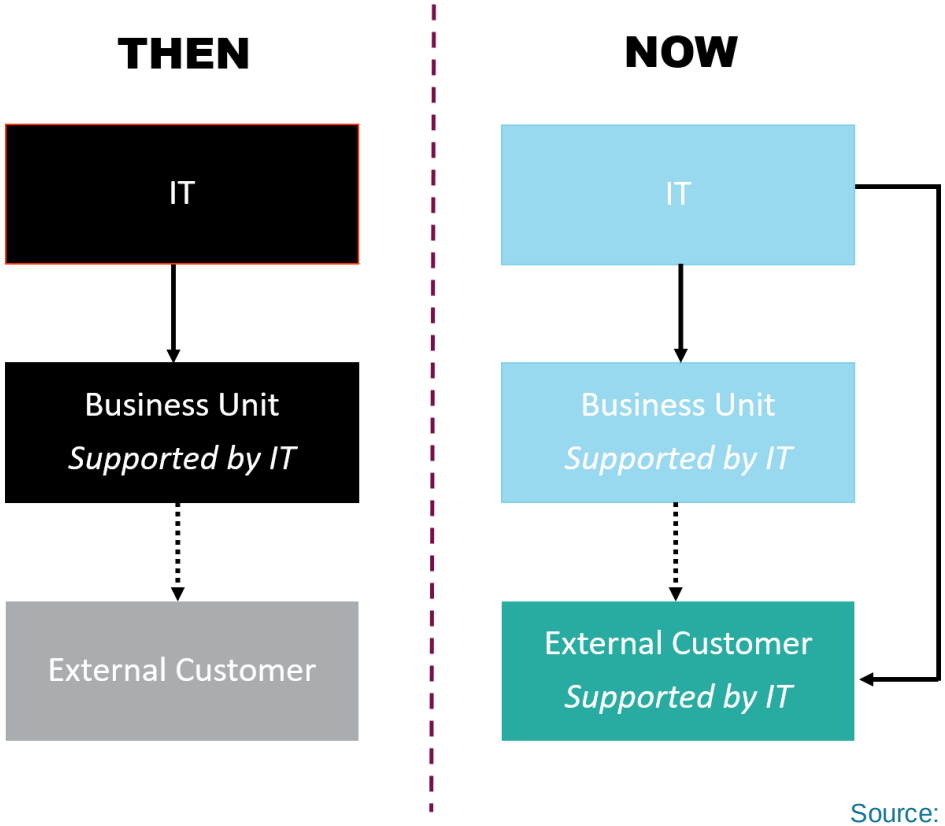
Customer expectations change and which category an attribute belongs to along the Kano model also drifts over time. The thing that is innovative and new that customers would never expect today over time becomes the thing that all companies offer as a standard and that customers demand. Continuing to offer the Excitement Attributes that differentiate you in the market means constant innovation, improvement and research to stay ahead of the curve.

Source: https://www.kanomodel.com

# IT Is Business Value

This drift is driven by customer expectations and by the level of performance from competing products. In recent decades, that drift is towards technology.

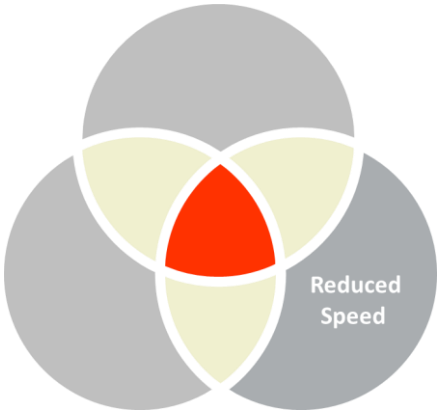
This is changing the role that IT plays in creating business value.



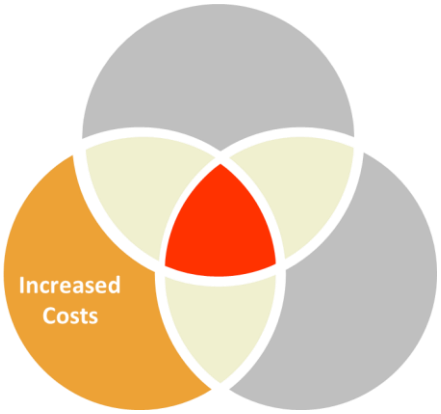

Source: https://www.kanomodel.com

# IT Impacts Business Value


- **Escaped Defects:** Defects move from development into operations and production without being caught
- **Poor Service Availability:** Overly complex or fragile legacy systems make it difficult to implement changes
- **Inability To Integrate Data:** IT systems developed as independent application stacks with their own data structure and database lack interoperability with other business systems




- **Slow Speed To Market:** Overly complex systems and processes result in increased defects and rework and deliver things slower
- **Lack Of Shared Priorities:** Processes for balancing and gaining agreement on shared priorities from multiple business units are immature
- **Backlog Of Work:** Increased rate of demand and inability to scale means that the backlog of work continues to grow and age




- **Rising Maintenance Costs:** Lack of prioritization to remove, standardize or consolidate processes and tools increases maintenance costs over time
- **Stagnation Of Growth:** As the budget to keep the lights on rises, capacity for funding innovation and growth decreases
- **Inability To Map Costs To Value:** Lack of insight into actual costs to deliver services limits investment

# External Drivers Of Change

## The Emergence Of Cloud Technology

The emergence of cloud technology has had an enormous impact on IT and contributes to increasing demand from the business for speed, flexibility and innovation. When it is more economical for organizations to lease a technical capability than to own it, the switch is inevitable.

## Global Supply

Global supply options mean that the business has many more options available to them if they decide to seek out alternative IT delivery models. The globalization of the market economy puts new pressures on local delivery options to up their quality, increase their speed and lower their costs.

## Workforce Automation

Workforce automation is changing how businesses think about traditional IT operations, monitoring and IT management work, such as running batch jobs, backing up IT systems and deploying changes to production systems.

While these tasks have been done manually in the past, they are rapidly moving towards automation in respect to improved quality, speed, cost and ability to quickly fall back and recover to previous versions.

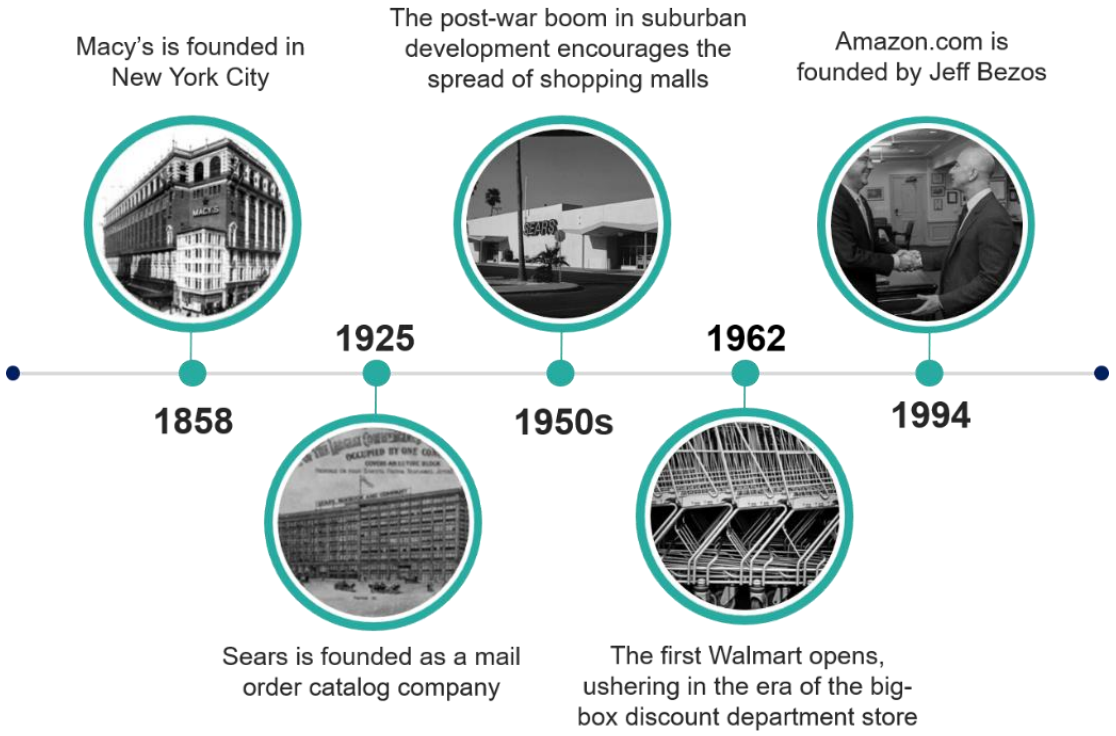
## Digitally Disruptive Competition

Newly formed, built-on-the-web, app-enabled brands are now able to leverage cloud-based services and software as a service (SaaS) business automation for core business processes in a matter of weeks if not days. So enabled, they come to market with new and disruptive products in a fraction of the time it would take a traditional brick and mortar organization to respond.

# Disruptive Competition In Action

In 2015, Amazon supplanted Walmart as the most valuable retailer in the United States by market capitalization. Amazon, as an example, encapsulates all of the external drivers and pressures being exerted on businesses today – it utilizes cloud technology and automation at a very high level to distribute its products and services globally in ways that no other company in its path has been able to match. In doing so, Amazon has forced all businesses it competes with to keep up or fall behind. And it's done this very quickly, decimating even the older, more established companies who have existed for decades, like Sears.

With this new degree of external pressure, organizations are forced to finally address long-standing cultural issues, political barriers and leadership gaps that have traditionally brought most improvement programs to their knees. So now is the time to change, and to change rapidly in order to survive – the banner many organizations are flying, to give a name to that change is "DevOps".



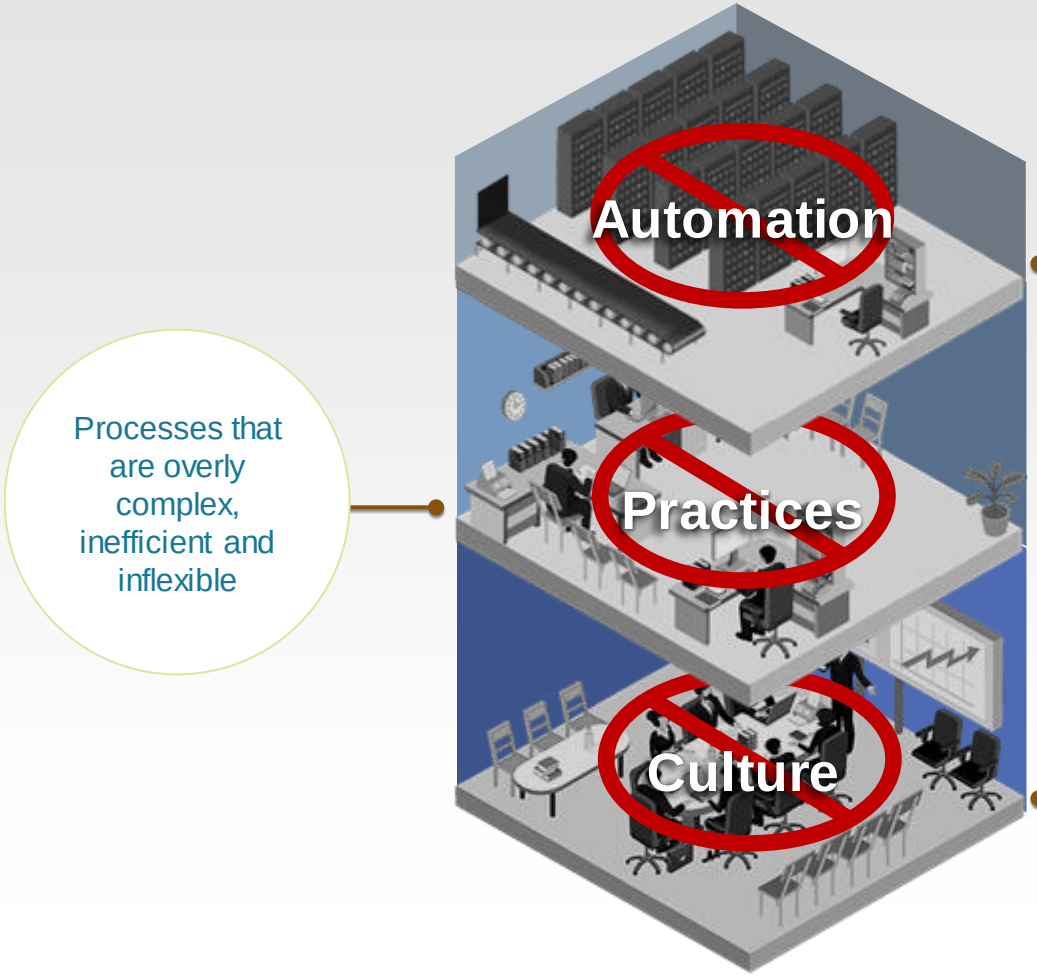
# The Organizational Goals Of DevOps

- Faster time to market
- Better integration between IT and the business
- Improved responsiveness
- Improved code and deployment quality
- Increased productivity
- Better visibility and transparency
- Increased flexibility and agility

A **Digital Transformation** is a profound transformation that encompasses all organizational activities, processes, skills and cultural attitudes.

- Digital Transformation allows businesses to leverage the changes and opportunities that are presented by digital technologies and their accelerating impact in a way that is strategized and prioritized to become an advantage and not a disadvantage.
- From the organizational perspective, this is an urgent requirement, and facilitating it means removing that disconnect between the business and IT.
- From the IT perspective, usually the need for a digital transformation is understood. The issue is that it seems impossible in the current environment.

# Why Is IT Unable To Deliver Value?




Legacy tools and systems in place that do not support automation and integration

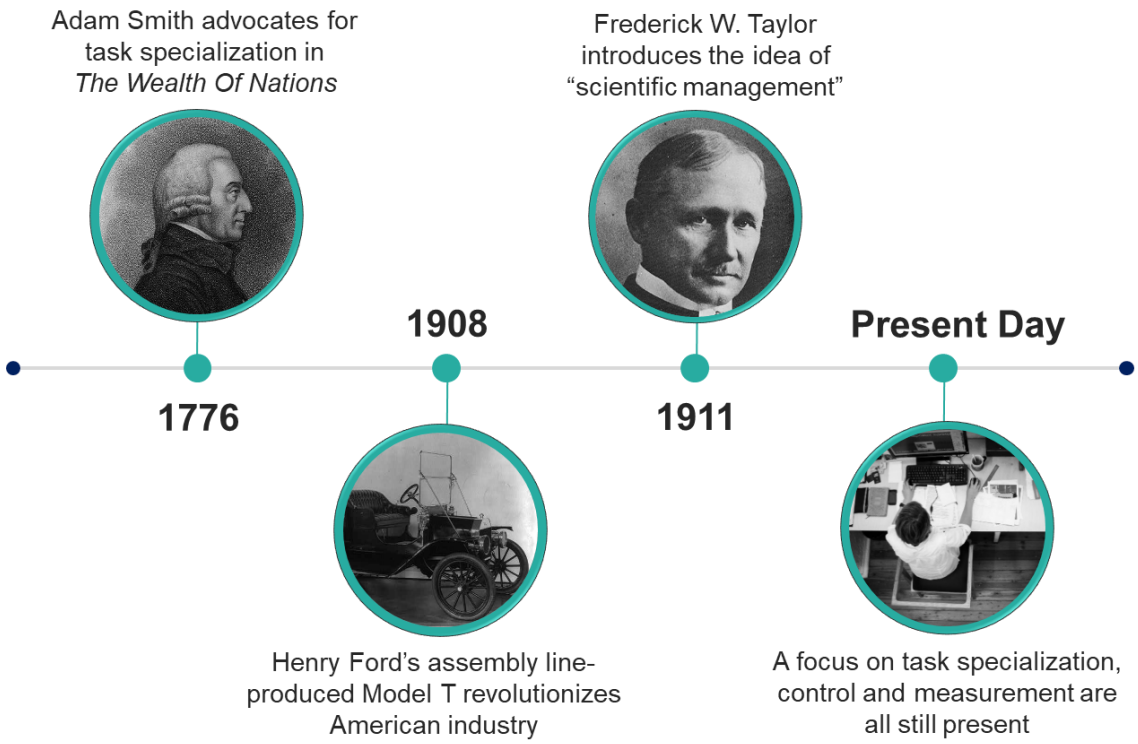
Processes that are overly complex, inefficient and inflexible

A culture structured in a way that creates silos that inhibit collaboration, innovation and automation

# How Silos Became The Norm

Current organizational structures have been inherited from earlier successful approaches such as assembly line production and scientific management. The focus on task specialization, control and measurement in organizations today creates silos.

Silos reduce collaboration, and roles and departments are not designed to be cross-functional. Each individual or functional team views their objectives only through the lens of their individual tasks and responsibilities, especially those which they are measured against. They lack the proper incentives or resources to respond to the needs of other teams, or the business as a whole.



# A Silo Mentality Leads To A Lack Of Systems Thinking

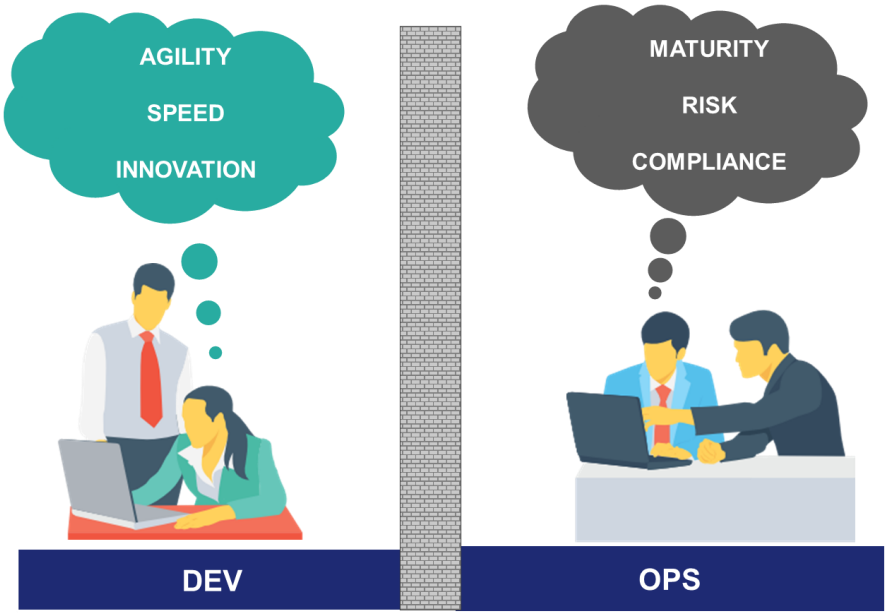
A **Silo Mentality** occurs when a team or department shares a set of common tasks but operates distinctly from other groups, with their power derived from association with a function or shared technical knowledge.

When you view yourself as only existing in your team or area, there is limited focus or value placed on a sense of collective responsibility for customer results. There is also limited focus or value placed on the individual's role in the larger organization – on the work that is handed off to them and on the results and impact of their work once it is handed off to someone else. The incentives that cause a silo mentality and a lack of systems thinking distort your sense of accountability.

**Systems Thinking** means understanding that your function is an interrelated and interdependent part of a larger system, which is defined by boundaries and more than the sum of its parts.

# Silos Create A Wall Of Confusion

**The Wall Of Confusion** is a scenario in which Dev and Ops don't fundamentally understand why the other side does things the way that they do or what their needs are, which impacts their ability to come together to deliver business value.




In almost every organization, the Dev and Ops teams operate in silos, accountable for and measured against competing sets of priorities and concerns. They are expected to simultaneously pursue two very different goals:

- Respond to rapid changes happening in customer and business expectations to remain competitive
- Provide customers and the business with stable, reliable and secure service

This puts Dev and Ops into a position where they are both required to work together to produce the best possible outcomes, but they also cannot work together because their goals are directly in opposition with each other.

# Local Optimization & Process Complexity

- Without systems thinking or a sense of accountability to the larger organization, when problems arise and solutions are required, the designed and implemented processes result in local optimization.

- Locally optimized processes, practices and solutions mean that work may get done within each individual area, but challenges with prioritization and productivity come in the hand off between groups with work often in a non-active waiting state. It also means that there is no attention paid to the benefits of shared capabilities and resources, and instead they are re-created or rebuilt within the area. This generates a high likelihood of redundant processes and resources and encourages non-integrated solutions.

**Local Optimization** is an environment that is structured and built to produce the best results for the individual or team. While it is important to create local efficiencies, you should be able to see how process design that is locally optimized within a silo can potentially be a problem.

# The Negative Influence Of Bureaucracy

In a 1922 essay, German sociologist Max Weber laid out some of the negative consequences of bureaucracy, as listed below. Regarding Western societies, Weber referred to bureaucracy as an "iron cage" that trapped individuals in systems based solely on efficiency, rational calculation and control.

| Strengths Of Bureaucracy | Weaknesses Of Bureaucracy |
|---|---|
| ▪ A division of labor into spheres of influence, specialization<br>▪ A definite hierarchy and authority<br>▪ Clear norms and rules<br>▪ Job assignment based on skills & qualification<br>▪ Promotion by seniority<br>▪ Oversight and governance over the incumbent of each office<br>▪ Better than feudal/traditional forms where people got appointed by favoritism or bribes | ▪ Becomes an Iron Cage of Control<br>▪ Red Tape grows over time as a reaction to process failure<br>▪ Hard to change or reverse<br>▪ Divisions of labor promote silo behaviors & culture<br>▪ Hierarchy can mean long approval times (e.g. must go up and down chains of command to get things done)<br>▪ Decision bias towards silo vs. the overall organizational goals |

Bureaucracy can have a negative impact on the processes and practices of IT and adds levels of complexity, redundancy and inefficiency. The problem arises when it is not fit for purpose and begins to impose too many constraints.

Bureaucracy is notorious for adding layers of complexity but then not correspondingly removing them when they are no longer needed. This creates layers upon layers of legacy processes that organizations do "just because", and apply universally and without nuance.

# Complex Processes Result In Waste

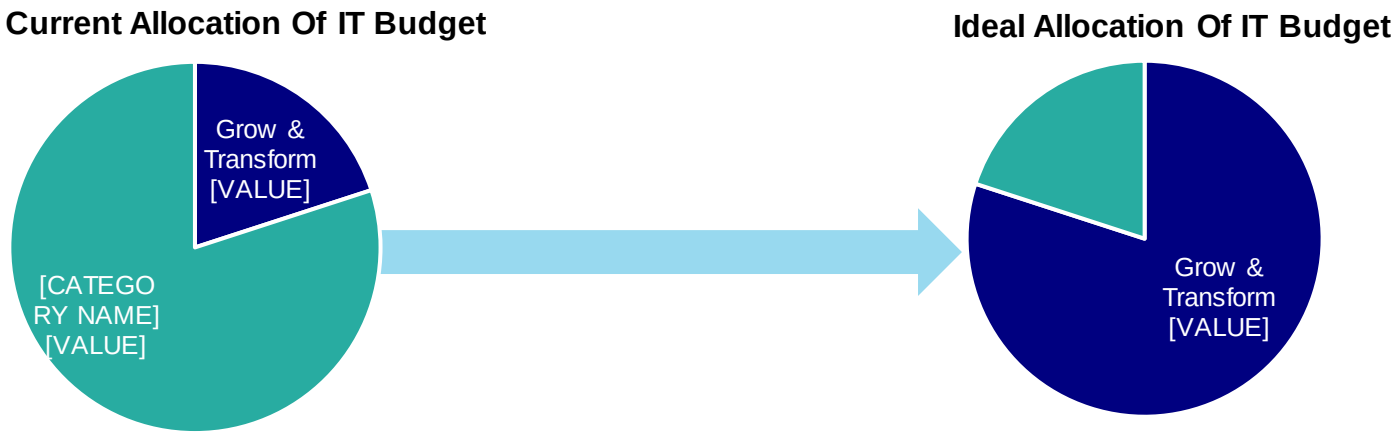
The result of silos that are locally optimized but not integrated or able to collaborate, as well as negative bureaucracy, is waste. Wasted time and energy and effort and resources. Aspects of processes that get in the way and are unnecessary.

- Incidents that occur hundreds of times without anyone looking at the root of the problem
- Supplier contracts that expire without knowledge until an incident occurs
- IT assets that are poorly tracked, controlled or inventoried
- Incident tickets that disappear into an IT back office black hole until someone shouts loudly enough
- Multiple Service Desks all with their own tools and separate processes
- Massive amounts of wasted server capacity due to a lack of capacity and demand management
- Redundant or duplicate tools purchased by various IT departments in the same organization
- Multiple change management processes due to political boundaries

# A Big Shift Is Required

- Year over year, the percentage of money spent to simply run the business also grows and the capacity for innovation decreases. The challenges that exist in processes and practices carry over into the technology and tools IT uses. Because IT lacks the capacity to innovate, grow or transform, they often find themselves without the resources or time to move to new tools, or falling back on legacy tools because the existing processes won't support anything else.

- The silo mentality and local optimization also discourage integration and the sharing of tools across teams, resulting in an organization where even those teams that do use the same tools do not use them the same way or in ways that are the most efficient. Without integration, automation is then impossible, and many processes that could be automated continue to be manual. This results in a reliance on systems and tools that over time become propped up and forced to barely function with a patchwork of short-term fixes and intentions to fix things later.



# IT Downward Spiral


**Stage 1: Operations Inherit A Challenge**

IT Operations encounters applications and infrastructure in their daily work that are overly complex, not documented properly, or fragile and prone to failure. Under pressure simply to maintain stability for customers, they don't have time to address the issue. Because of this, a failure occurs.

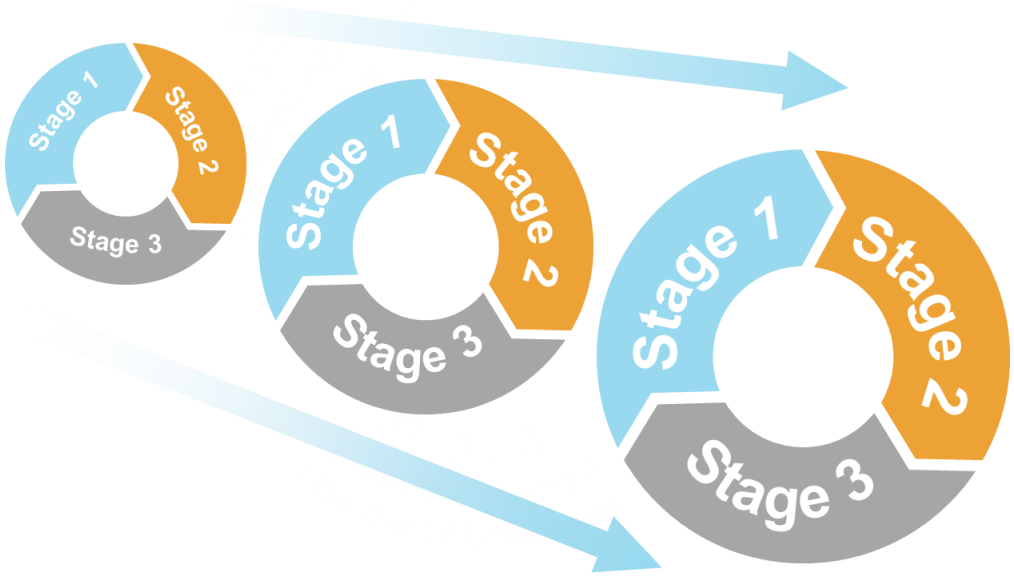
**Stage 2: Development Is Pressured To Fix It Fast**

Someone attempts to compensate for the failure by promising a bigger, bolder course correction. Development is asked to solve this new technical challenge quickly and is forced to cut corners to do so. They rush the fix into release telling themselves that they will return to the project at a later date, but then they never have the time.

**Stage 3: Operations Inherits A New Set Of Challenges**

A rushed, potentially flawed solution to the problem is implemented, which then brings its own challenges to Operations, which then leads right back to the first stage all over again.

# The Accumulation Of Technical Debt

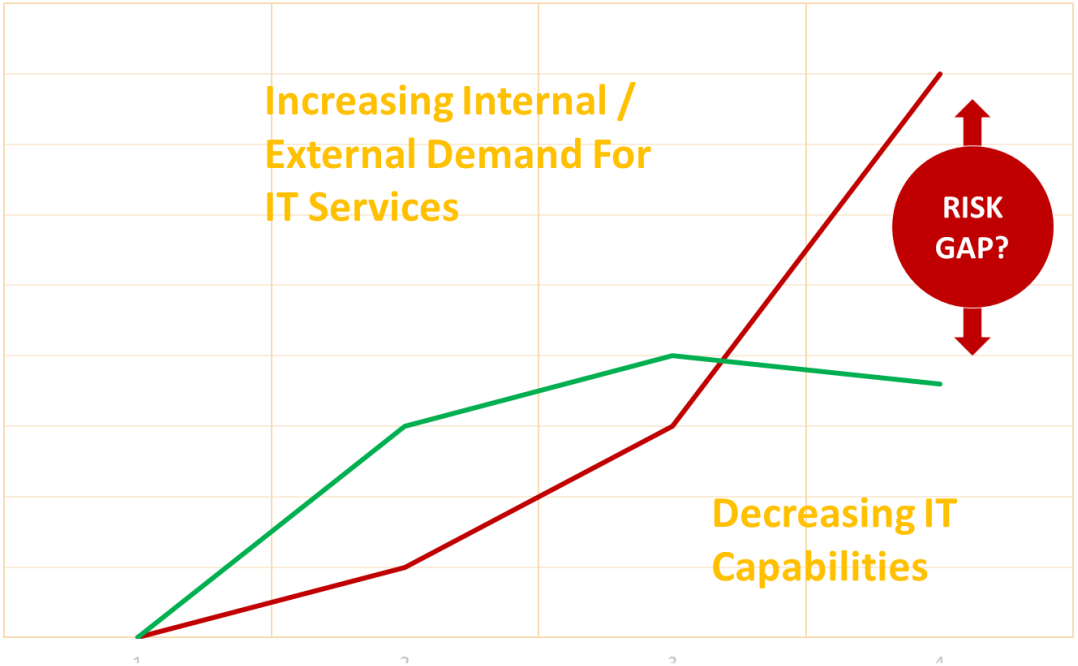



**Technical Debt** is the accumulation of complicated workarounds and rework that occurs when easy solutions are consistently implemented instead of the best solutions.

Technical debt accumulates in much the same way that monetary debt does. As the cost of rework and overly complex processes demands time from those in Operations, it then decreases their ability to address previous failures or provide feedback to Dev, which then leads to Dev being asked to compensate for those failures and having less time to address past releases that were flawed, and so on. This is similar to the way interest on monetary debt decreases the ability to pay back the original debt.

Once that happens, things only get worse because production cycles get slower and slower. People get less and less ambitious about undertaking new projects, and fewer new projects get completed.

# What Happens If IT Can't Deliver Value?



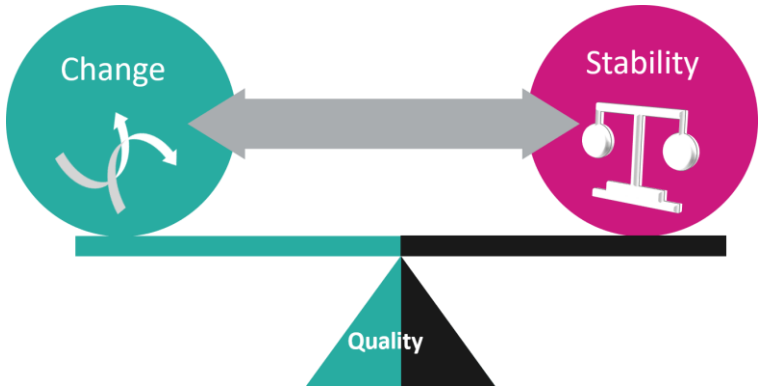

- The same growth of cloud technology and global supply options that drive the increased demand from the business also results in IT seeing increased competition.

- There is an observable trend towards the usage of "Shadow IT" investments, where a line of business purchases their IT services directly from the external market, or funds their own IT capability due to the perception of poor quality, slow speed or higher costs of existing options. "Shadow IT" is just another way of saying "loss of market share for internal IT".

# IT Goals Of DevOps

- Faster delivery of features
- Increased collaboration and communication
- Faster resolution of problems
- Continuous software delivery
- More productive teams
- Reduced complexity to manage
- More time to innovate

## DevOps Helps IT Find The Balance




- DevOps emerged to help IT find the balance between **change** and **stability** that the business requires to deliver value.

PeopleCert DevOps Fundamentals

# Module 2:
# **Key DevOps Principles & Concepts**

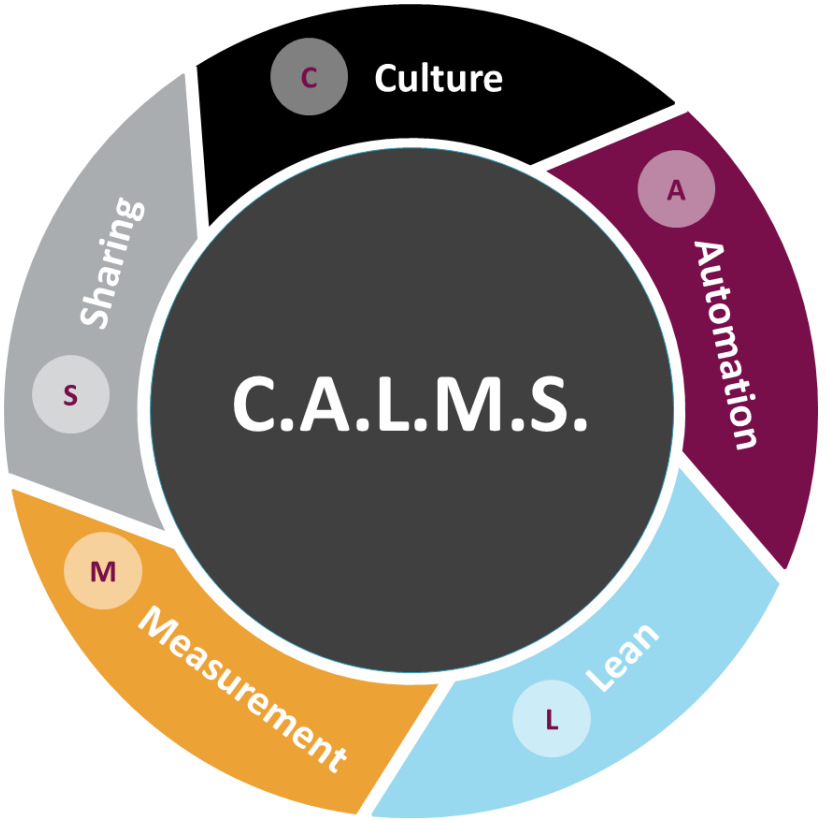
**Topics:**

- C.A.L.M.S. & The Three Ways
- Continuous Delivery Across The Deployment Pipeline
- The Scope Of The Full Stack

# C.A.L.M.S.

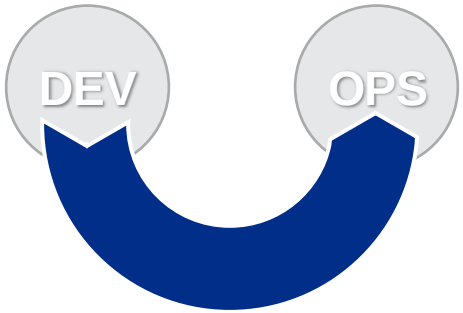
- CALMS is an acronym originally developed by John Willis and Damon Edwards in 2010, and later further refined by Jez Humble, as a way to understand the key pillars and concepts that underpin DevOps.

- **Culture:** An understanding that the values, beliefs and attitudes that permeate an organization and IT are important and need to shift.
- **Automation:** A belief that anything can be automated and that manual processes should be automated wherever possible.
- **Lean**: An emphasis on ideas from the Lean value system and methodology to reduce process complexity and streamline the flow of work.
- **Measurement**: A belief in the importance of measuring everything and building processes that are visible and transparent.
- **Sharing**: An emphasis on collaboration and communication between development and operations allowing them to effectively integrate.



# The First Way: Flow

**For more information, check out this video:**
*The Three Ways Of The Phoenix Project:* https://www.youtube.com/watch?v=nUOXDEvplRc&x=mobref




The **First Way** focuses on understanding how work flows – at its most basic, how work moves from Dev to Ops and then from the functional areas of the business to customers – from left to right, one team to another.

This means identifying how individual pieces fit together – understanding what things are moving from team to team, why they are moving, and what will happen to them from there.

Flow is about identifying the constraints discussed earlier – understanding where the bottlenecks are and where productivity is slowest so that you can then prioritize for the biggest impact.

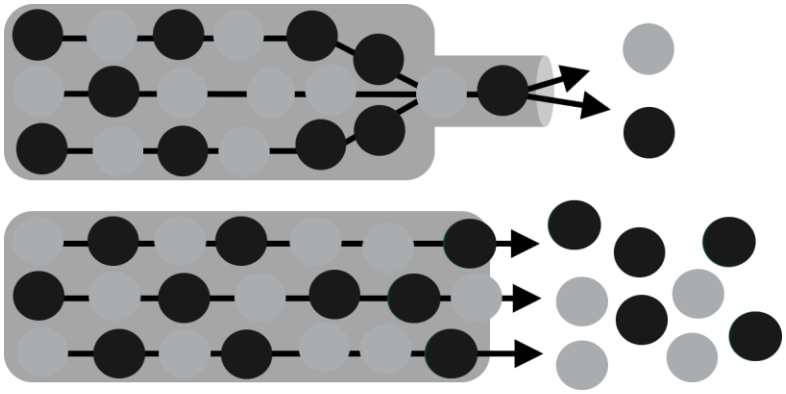
It is only if you understand how work flows, that you can clearly identify obstacles contributing to the "wall of confusion" and eliminate it.

Source: https://itrevolution.com/the-three-ways-principles-underpinning-devops

# The Theory Of Constraints

Dr. Eliyahu Goldratt's Theory Of Constraints states that no complex system or process can be more efficient or stronger than its limiting constraint or bottleneck.

Achieving flow means identifying where constraints exist. It also means having a clear understanding of everything happening along the value stream for each process, the set of activities that make a process up from beginning to end, and the people involved. And not in a vague way, with assumptions or best guesses.



Source: *The Goal,* Dr. Eliyahu Goldratt

## How Do You Increase Flow?

| | | |
|---|---|---|
| Increase the transparency and visibility of work | Place limits on Work In Progress (WIP) | Work in smaller batches |
| Reduce and limit the number of handoffs | Identify and prioritize constraints or bottlenecks | Continually improve to eliminate waste wherever possible |

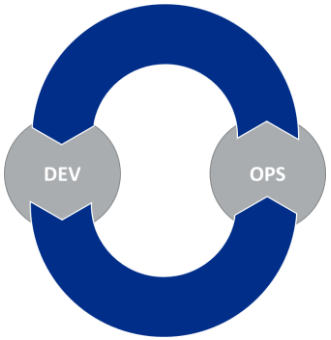
Source: https://itrevolution.com/the-three-ways-principles-underpinning-devops

# The Second Way: Feedback

The **Second Way** describes the principles that enable the reciprocal fast and constant feedback from right to left at all stages of the value stream.

In IT, work happens within complex systems. The risk of catastrophic failure or consequences is very high. There tends to be a high degree of interconnectedness with tightly coupled components, and this variability and complexity means that doing the same thing twice will not always necessarily produce the same results.

This means that safety, reliability and stability become very important – systems thinking becomes crucial. Feedback loops help to create safer, more resilient systems of work because they provide more opportunities to detect and correct errors.




**How Do You Increase Quality & Resiliency?**

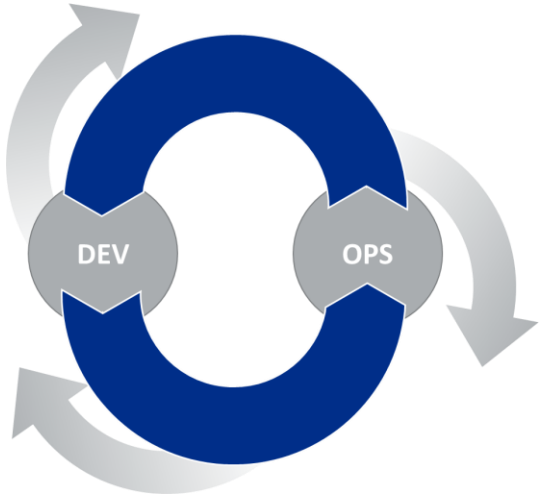
- Detect problems earlier, as they occur
- Swarm to solve problems and gain new knowledge
- Push quality closer to the source
- Optimize globally vs. locally

Source: https://itrevolution.com/the-three-ways-principles-underpinning-devops

# The Third Way: Continual Experimentation

The Third Way focuses on the creation of a culture of continual experimentation and learning to enable the ongoing creation of knowledge for individuals, teams and the organization.

It involves breaking down silo culture, which is often a culture of fear and low trust, and helping IT to embrace failure as a learning experience and to see the value in trying new things and innovating.




## How Do You Encourage Experimentation?

| | | |
|---|---|---|
| Enable a safe culture for learning | Institutionalize continual daily improvement | Turn local knowledge into global improvements |
| Implement resilience patterns in daily work | Transform leadership to encourage learning | Approach failure as a learning opportunity and plan on failing faster |

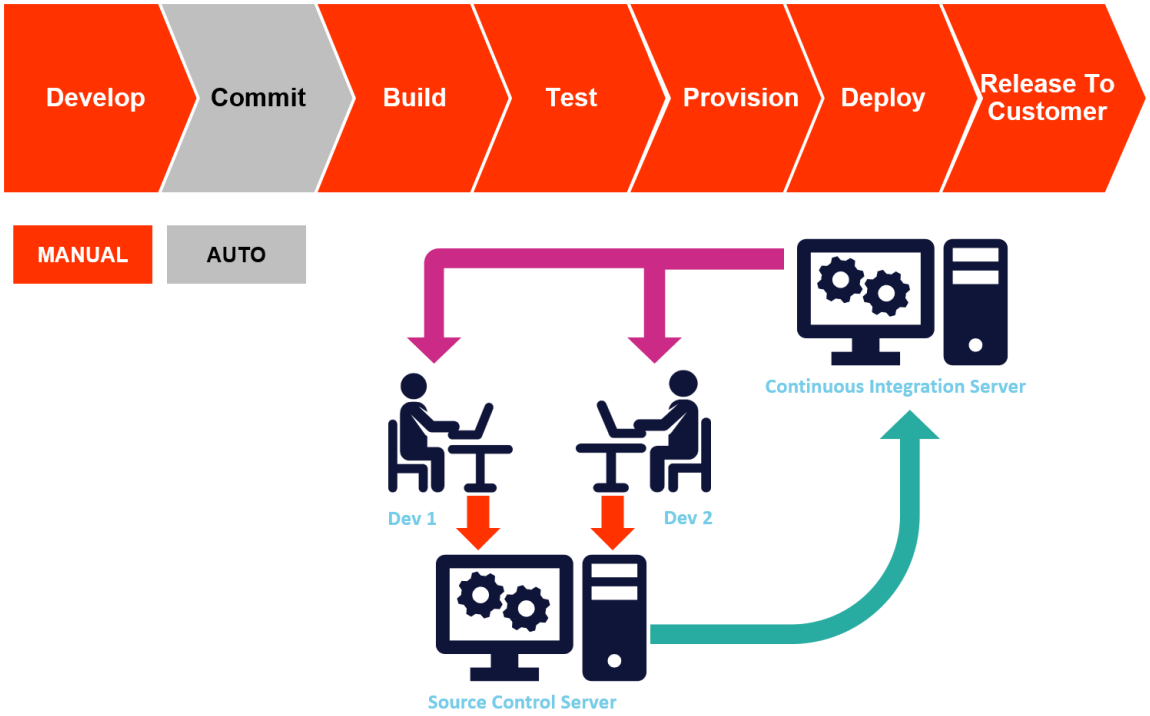
Source: https://itrevolution.com/the-three-ways-principles-underpinning-devops

# The Deployment Pipeline

**The Deployment Pipeline** is a process model that gives you a starting point from which to provide transparency and controls as work moves through the various sets of tests and deployments towards release.

# Continuous Integration

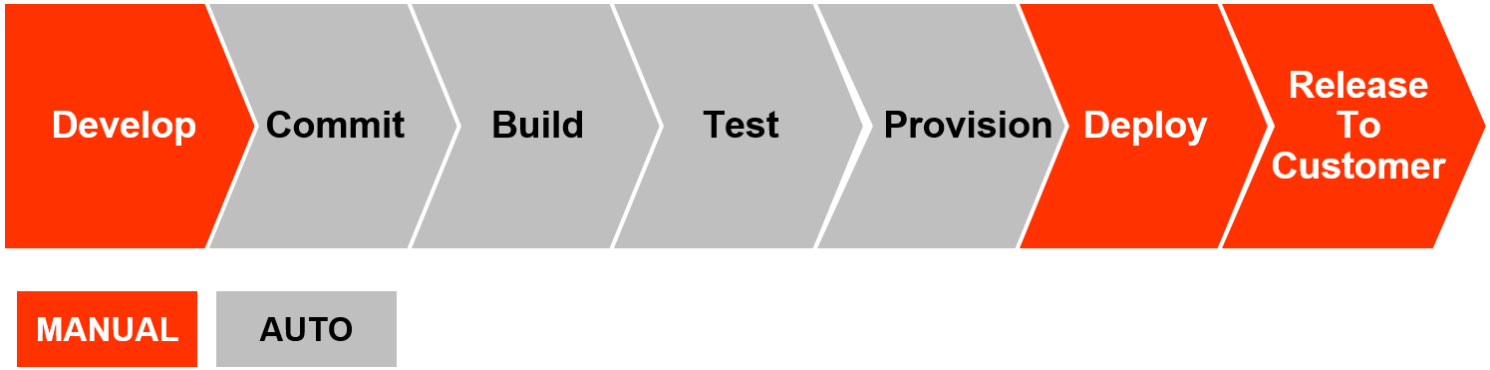
**Continuous integration** is the practice of merging all developer working copies into a shared mainline (a code repository or a main code trunk) throughout the day.




Within an automated Continuous Delivery process, continuous integration covers mainly the build stage. Usually, continuous integration applies to integrating, building and testing code within the development environment.

# Continuous Delivery

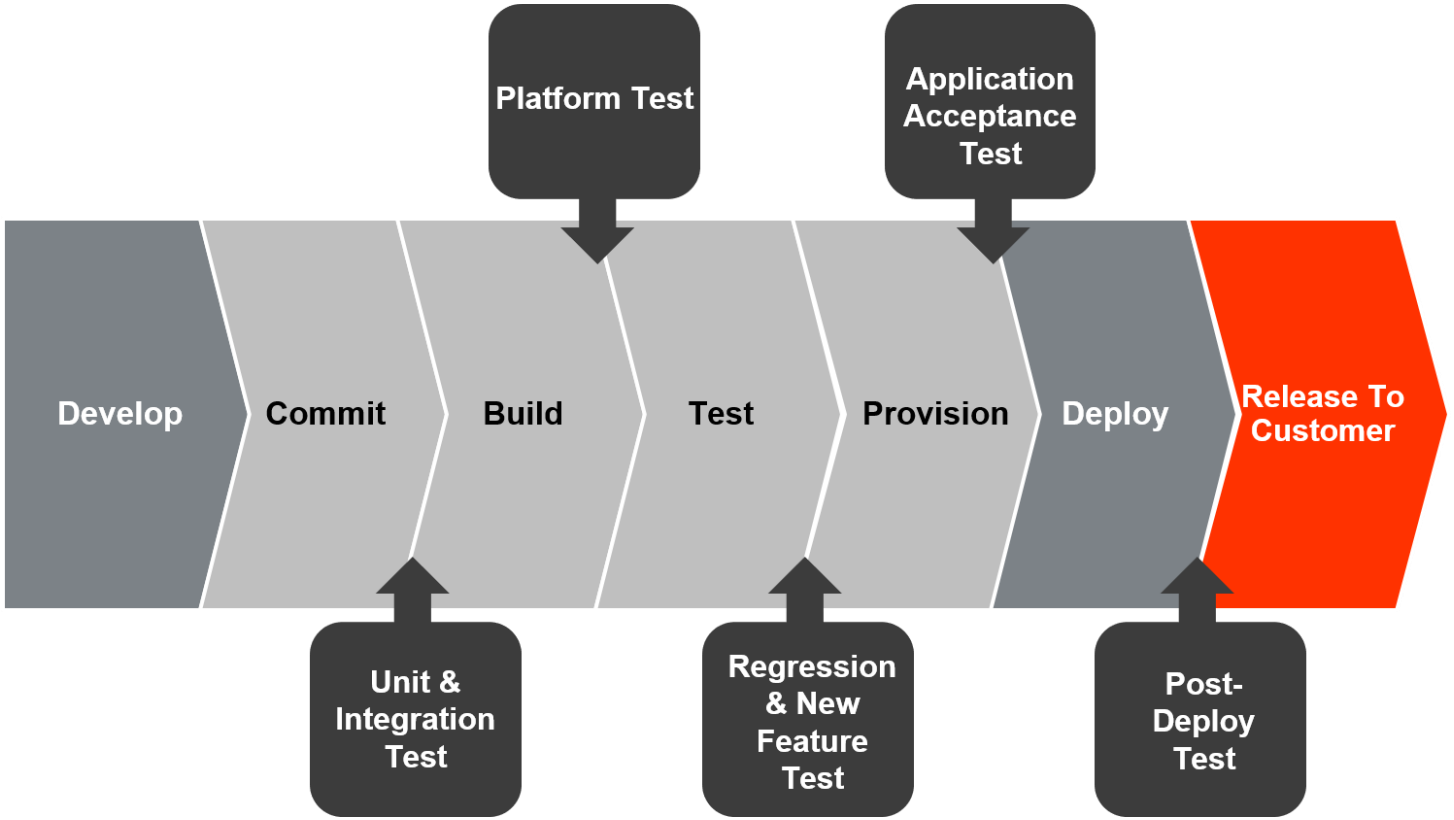
**Continuous Delivery** is a set of practices designed to ensure that code is always able to be deployed rapidly and safely throughout its lifecycle to production, achieved by pushing the executables into a production-like environment and conducting automated testing to detect problems.




Continuous Delivery doesn't mean that every single change gets deployed to production. It simply means that every change could be deployed to production at any time. It is not the same as continuous deployment. In continuous delivery, release is a manual, human decision. Continuous delivery requires that if any change causes an automated test to fail that the developers stop the deployment pipeline and bring the system back towards being in a deployable state. Continuous Delivery is an essential DevOps practice. It is only through the continuous delivery of code that you can be confident that changes are producing business value.

# Continuous Testing

**Continuous Testing** involves the execution of automated tests at every stage of the Deployment Pipeline. It provides immediate feedback at each stage to mitigate risk. Automated continuous testing is a key component of continuous integration and delivery. It ensures that the code and environment operate properly and remain in a deployable state.

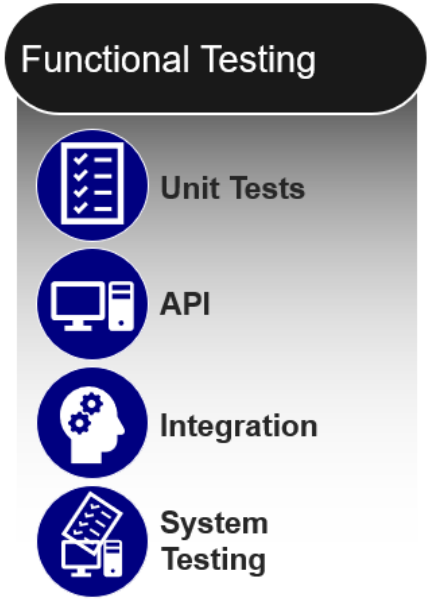


# Functional & Non-Functional Testing

There are two key testing types that you must differentiate between related to continuous testing – functional and non-functional testing.

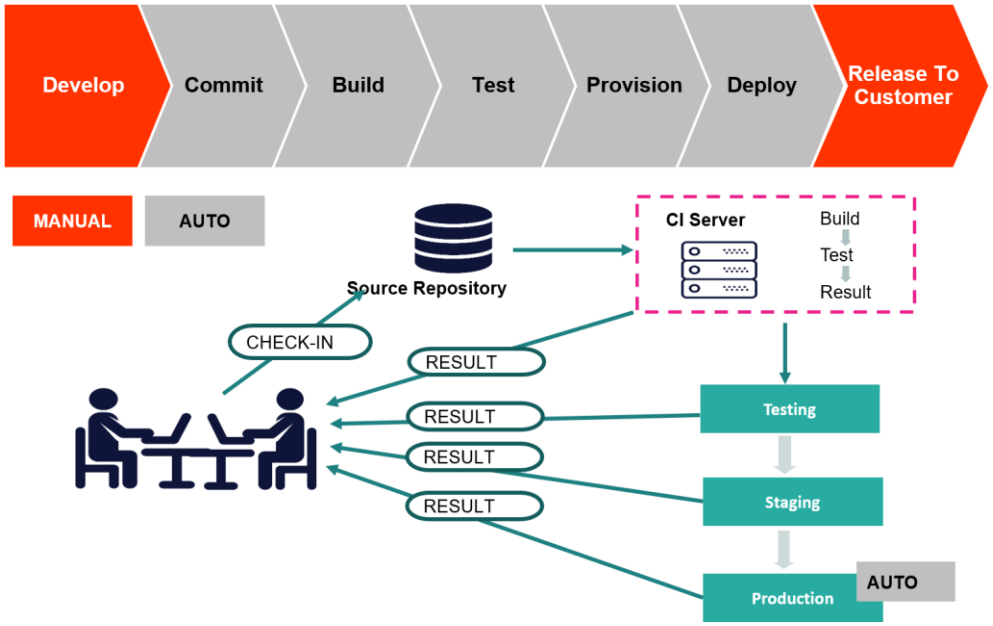
Functional testing is the testing of the features which are necessary for the product to work. It tests against the Critical To Quality components which the customer would reject if failure occurs.

Non-functional testing is the testing of system operation rather than specific outcomes. This means that it tests against things which the end user – the customer – may not be aware underpin the functional performance they see.




# Continuous Deployment

**Continuous Deployment** is an extension of the concept of Continuous Delivery. In Continuous Deployment, all changes that pass automated tests are then automatically pushed into production. It automates the step that was previously manual in continuous delivery, and enables multiple deployments per day.




- Continuous Deployment should be the goal of DevOps, however it is understood that it may not be practical in all cases. Many organizations face regulatory or other requirements that make Continuous Deployment not viable.

- **Deployment is not necessarily the same thing as release. Release is always a manual decision.**

# The Benefits Of Continuous Delivery

**Continuous Delivery across the deployment pipeline means…**

| | | | | |
|---|---|---|---|---|
| Effective test data management | Comprehensive, fast and reliable test and deployment automation | Trunk based development and continuous integration | Application code, app and system configuration all in version control | Incorporating security (and security teams) into the delivery process |

**Which leads to…**

| | | | | |
|---|---|---|---|---|
| Less rework | Lower levels of deployment pain | Higher levels of IT performance (higher throughput and stability) | Lower change failure rates | Identifying strongly with the organization you work for |

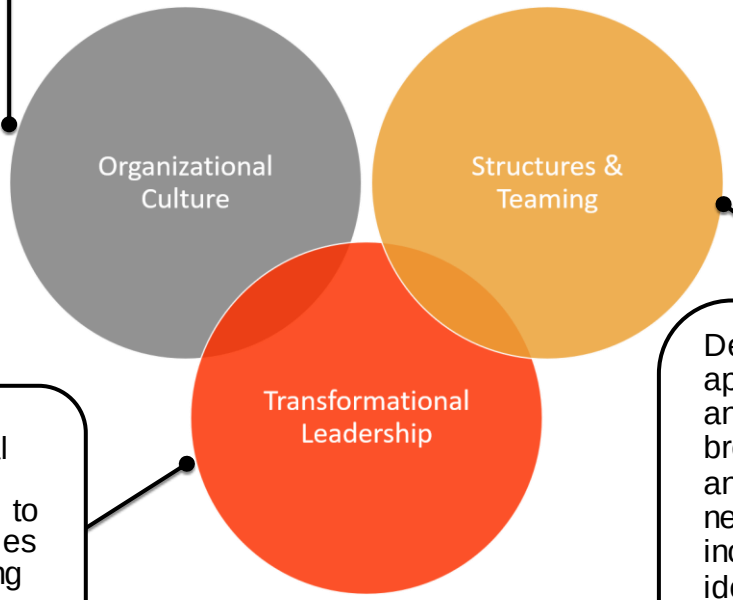
**Higher levels of organizational performance**.

**For more information, check out this video:**
*Continuous Delivery Is A Team Sport:* https://www.youtube.com/watch?v=9XmvFvdTObY

Source: *2016 State Of DevOps Report*, Puppet Labs

# DevOps Transforms People & Culture

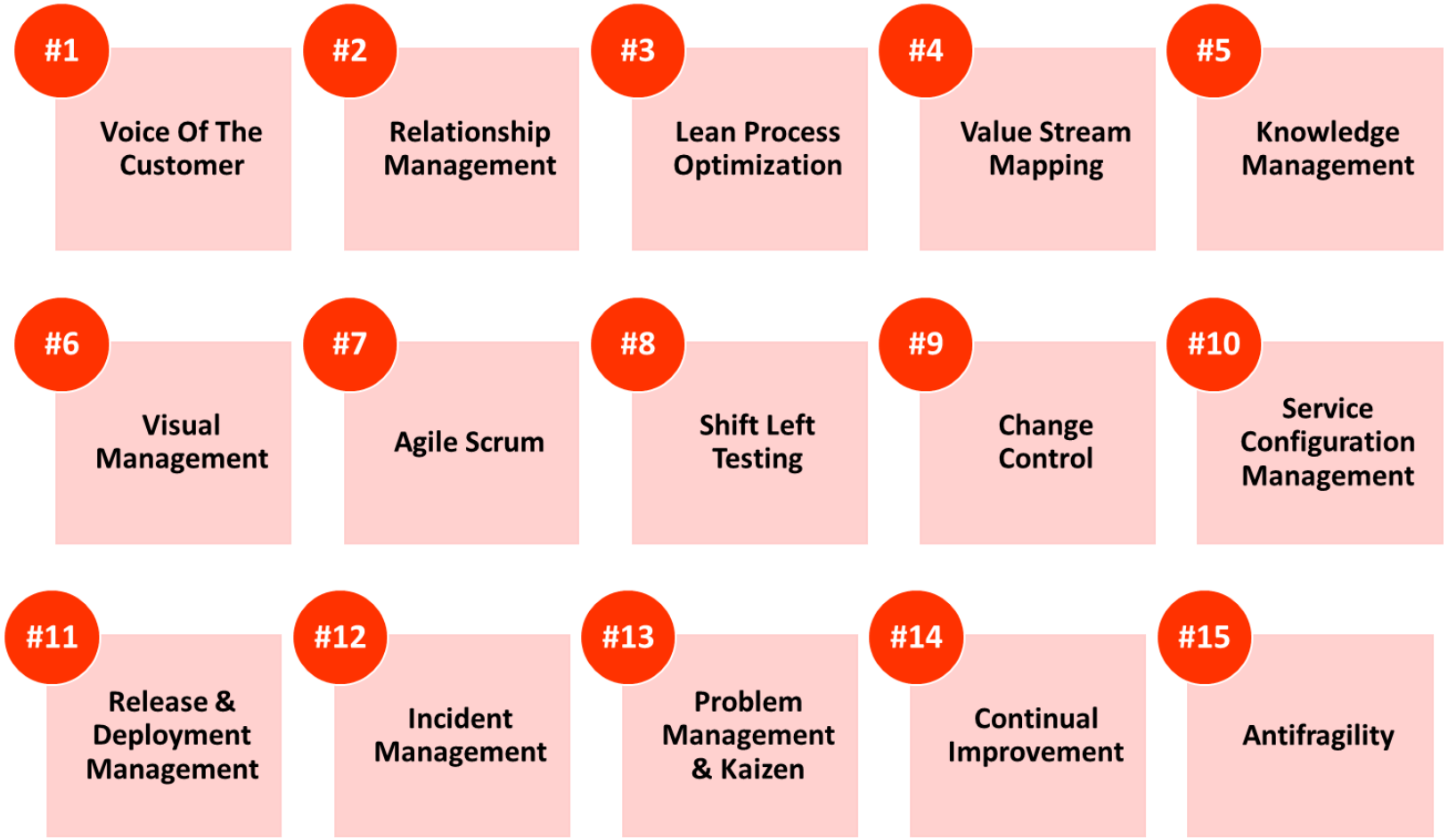
DevOps focuses on the barriers that currently exist within IT and breaking down silos and the wall of confusion. This means encouraging a culture of communication, collaboration and experimentation.



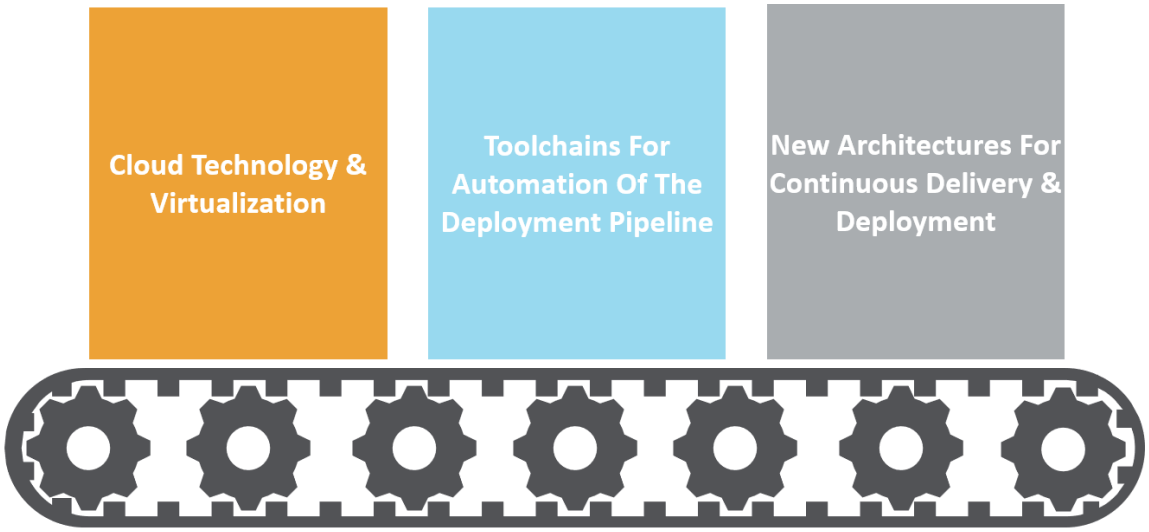

DevOps addresses the need for transformational leadership, which encompasses new ways to determine shared priorities and get everyone working towards the same goals.

DevOps advocates for new approaches to structures and teams within IT that break down silo mentality and encourage the necessary culture. This includes concepts and ideas like cross-functional teams, self-organization and product and platform teams.

# DevOps Transforms Processes & Practices



# DevOps Transforms Automation




DevOps includes a layer that addresses challenges related to technology – how to take the right processes and practices and speed them up even further through integration and automation. This involves integrated toolchains – understanding which tools to use, how to integrate them across teams and to enable automation of a wide variety of processes.

It involves developing a high sophistication and maturity level related to automation of the deployment pipeline and an understanding of how to leverage cloud technology and virtualization to further speed things up and increase collaboration and efficiency.

PeopleCert DevOps Fundamentals

# Module 3:
# The Full Stack – People & Culture

**Topics:**

- What Is A DevOps Culture?
- Transformational Leadership
- What Is A DevOps Team?

# Organizational Culture

**Organizational Culture** is the pattern of shared assumptions and values learned within an organization. It takes its cue from observable patterns of behavior in an organization that are picked up over time by those who work there.

- Organizational cultures are created by external adaptation and internal integration. This means that they aren't static. They're dynamic and evolve over time in response to change.
- Organizational culture doesn't exist in a vacuum within the organization. It is shaped in part by the broader culture of the society in which it operates.
- Organizational culture also isn't as monolithic as many assume it to be – there can be subcultures within an organization (sometimes IT is one of them) with their own mentalities, values and attitudes.

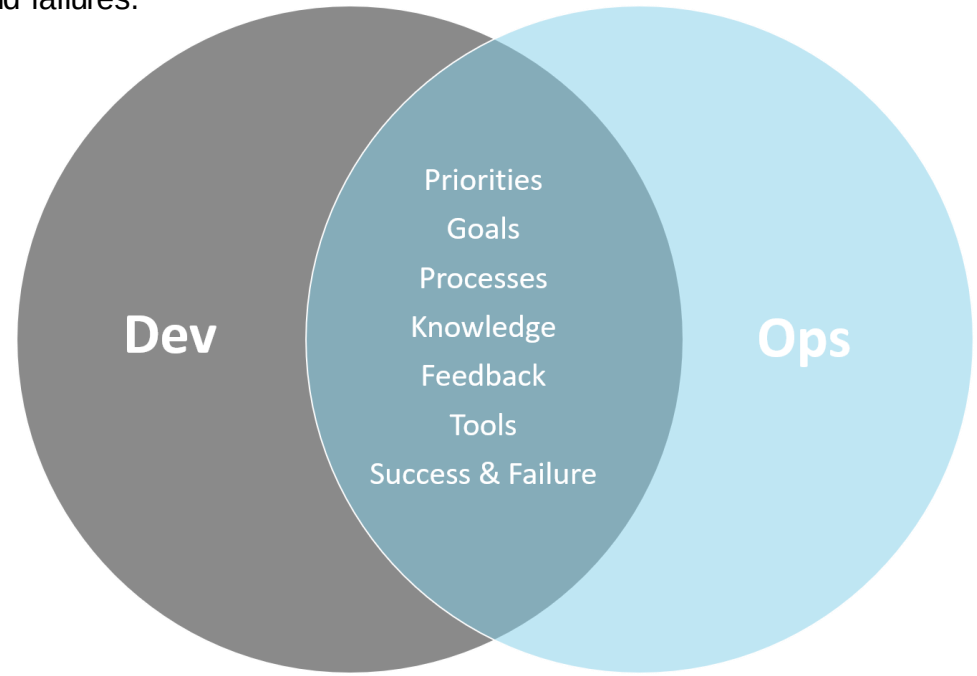
# Key Characteristics Of Collaboration

- When we talk about the kind of culture that supports DevOps, the most important thing we are talking about is **collaboration**.
- Breaking down the wall of confusion and the silos it causes requires prioritizing collaboration and teamwork – breaking down the silo mentality that built that wall in the first place.
- Collaboration can be a challenge. Organizations are often structured and set up in ways that discourage it.
- Culturally, there are some key behaviors and attitudes that act sometimes as barriers to the kind of collaboration DevOps demands, and that have to be addressed.
- This may not be easy – many organizations have a wide range of barriers in place that actively discourage communication, transparency, trust and accountability. But without those things, collaboration becomes impossible.

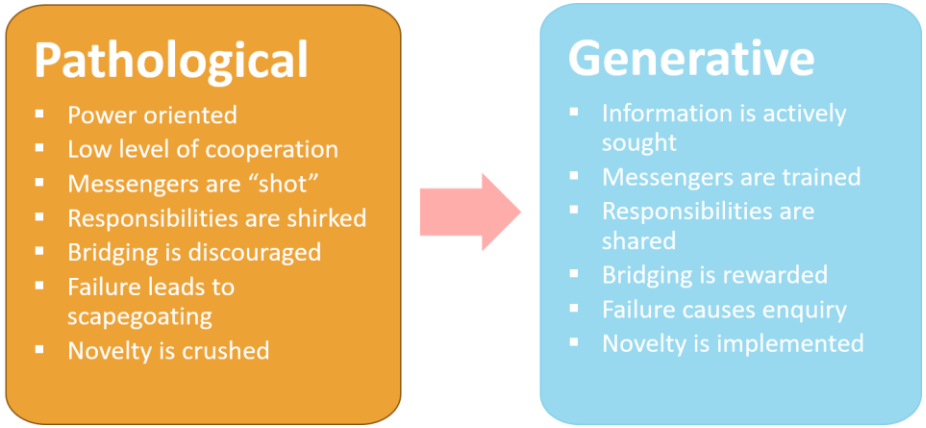
# DevOps Is About Sharing

- Sharing and the collective spirit is crucial to DevOps. Without it, successful adoption is impossible, as it is the key way that DevOps addresses the wall of confusion happening between Dev and Ops within IT.

- There is a need for Dev and Ops to share priorities and goals to help break down the silos that are currently a problem for IT. There is also a need to share processes and knowledge in order to achieve flow as emphasized in the First Way. The Second Way is about the sharing of feedback as central to DevOps. And of course, Dev and Ops ultimately need to understand that they have always shared successes and failures.



# DevOps Encourages A Generative Culture




- Many of the things that we identified as barriers to DevOps – as barriers to communication, transparency, trust and accountability – are things classified and categorized by the sociologist Ron Westrum as being indicative of a Pathological organizational culture. In his model, which was designed to predict safety and performance outcomes in the healthcare industry, he actually described three types of organizations – Pathological, Bureaucratic and Generative.

## DevOps Encourages Experimentation

DevOps encourages failure, but **responsible failure** – calculated risks, failing fast in small ways, and then learning and implementing solutions that actually work in the aftermath.

Source: *A Typology of Organizational Cultures,* R Westrum

# What Is Transformational Leadership?

**Transformational Leadership** is a leadership style that shifts the organizational culture towards something generative, reinforces a shared set of priorities and goals and that supports DevOps.

## Transformational Leadership Characteristics

According to *the 2017 State of DevOps Report*, Transformational Leadership is "a model in which leaders inspire and motivate followers to achieve higher performance by appealing to their values and sense of purpose, facilitating wide-scale organizational change." It identifies five characteristics of a Transformative Leader, which are:

- Vision
- Inspirational Communication
- Intellectual Stimulation
- Supportive Leadership
- Personal Recognition

Source: *2017 State Of DevOps Report*, Puppet Labs

# Transformation Requires Vision


An aspirational expression of what the organization would like to achieve that serves as a clear guide for future action.

## Contrasting Leadership Styles

| Traditional Leadership | Transformational Leadership |
|---|---|
| Works to a financial plan | Works towards a vision |
| Demands quick results | Demonstrates patience |
| Leads with answers | Leads with questions |
| Climbs the ladder rapidly or finds a new ladder | Gradually earns way up the ladder |
| Sees teams as a means to an end | Develops individuals and enable effective teamwork |
| Proud of having an open door policy | Learns and leads by being present and involved |

Transformational leadership, which has its basis in Lean leadership, favors facilitation over command and control. It trusts that each team member brings their own skills and knowledge to the team, and shifts the leadership role to one which gets the right people together, supports them and gets out of their way.

# Authority Distribution In Teams


Source: *The Psychology Of Self-Management In Organizations*

# Traditional vs. Self-Organizing Teams

| Traditional Teams | Self-Organized Teams |
|---|---|
| Individual focuses on their responsibilities | Individual is asked to focus on end-to-end process |
| Individual is accountable only for goal achievement | Individual is asked to be transparent, adaptable |
| Manager controls daily tasks | Team controls daily activities as a group |
| Management style is direct and authoritative | Manager acts as a coach / facilitator |
| Leadership is top down | Leadership is shared by the team |
| Many narrow jobs performed by individuals | Process is subdivided into tasks, team completes the whole task |

**For more information, check out this video:**
*Drive: The Surprising Truth About What Motivates Us:* https://www.youtube.com/watch?v=u6XAPnuFjJc

# Organizational Models For DevOps

**Conway's Law** is that "organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations".

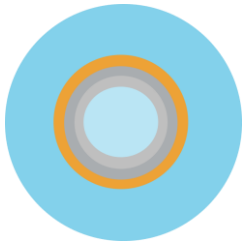
## Traditional Organizational Models

**Vertical organizations** require a constant exercise of power by one person to keep everyone else in line. This structure works well to tame chaotic environments, but is very reactive and focused on short-term goals.

**Matrix organizations** is an attempt to overcome the siloed shortcoming of the vertical structure by introducing an additional reporting accountability on a horizontal basis.

## Organizational Models For DevOps

**Product organizations** are about beating the competition, and managing by objectives. It is still command and control, but it is more of a meritocracy and rewards accountability and innovation.

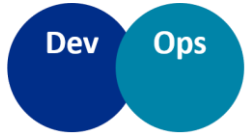
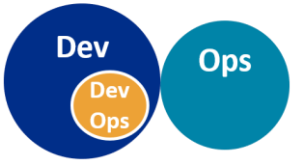
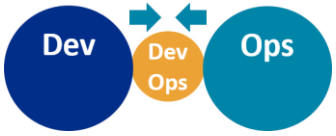


In an **Adaptive organization**, there is a focus on culture, empowerment and employee motivation. It is about stakeholders, not shareholders and works to resemble a family by being autonomous and self-organized.

Source: http://itrevolution.com/devops_enterprise_forum_guidance

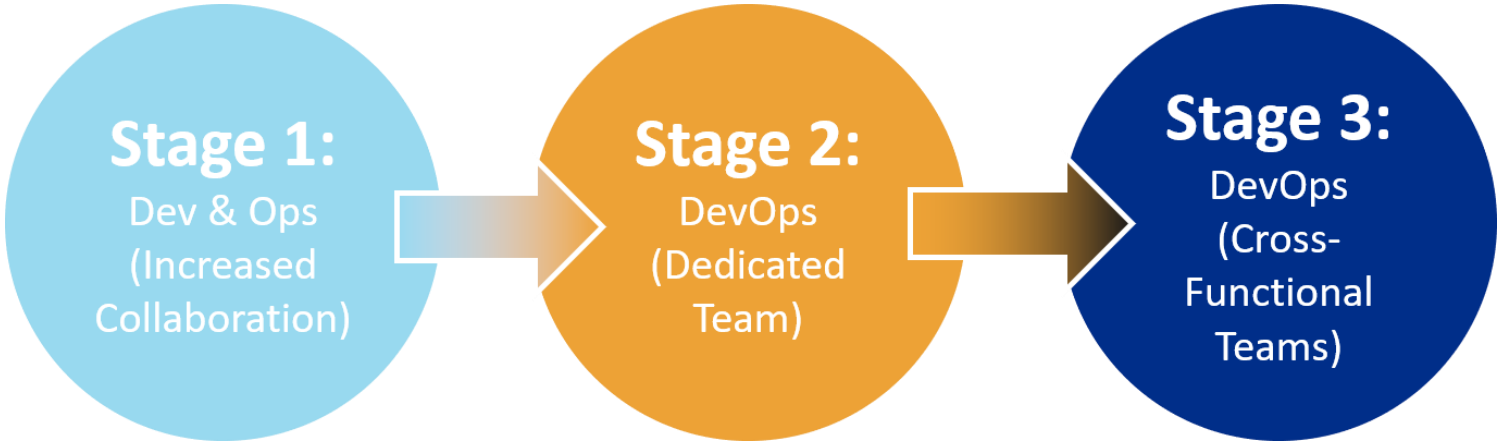
# Ideal Structures For DevOps

- While the term "DevOps Team" is often used, there is a danger to simply create another functional silo that sits between Dev and Ops. DevOps isn't necessarily just about creating a specific type of team. It isn't prescriptive about what a "DevOps Team" must be. Instead, it embraces a variety of patterns and potential structures that can enable the culture, practices and technologies necessary. Which works for your organization will depend upon many variables.

| Structures That Don't Work | | Structures That Work | |
|---|---|---|---|
| DevOps Team Silo: |  | Dev & Ops Collaboration: |  |
| Dev Doesn't Need Ops: |  | Ops As Infrastructure As A Service (IaaS): |  |
| Dev As A Tools Team: |  | DevOps Team With Expiry: |  |
| DevOps = DevOps Engineer |  | | |

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License. It is attributed to Matthew Skelton and the original can be found at http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish

# The Evolution Of DevOps Teams




## Getting Away From Task Orientation

- Discourages creativity and encourages focus on task execution only
- Jobs become tedious and boring
- Individuals become entrenched in one specialization
- Leaders are encouraged to micromanage and hoard information
- Micromanagement leads to lack of focus on relationships and innovation
- Promotes an attitude of completing your task and not caring about anything else
- Promotes a culture of blaming

Source: *Principle-Based Organizational Structures* by Dean Meyer

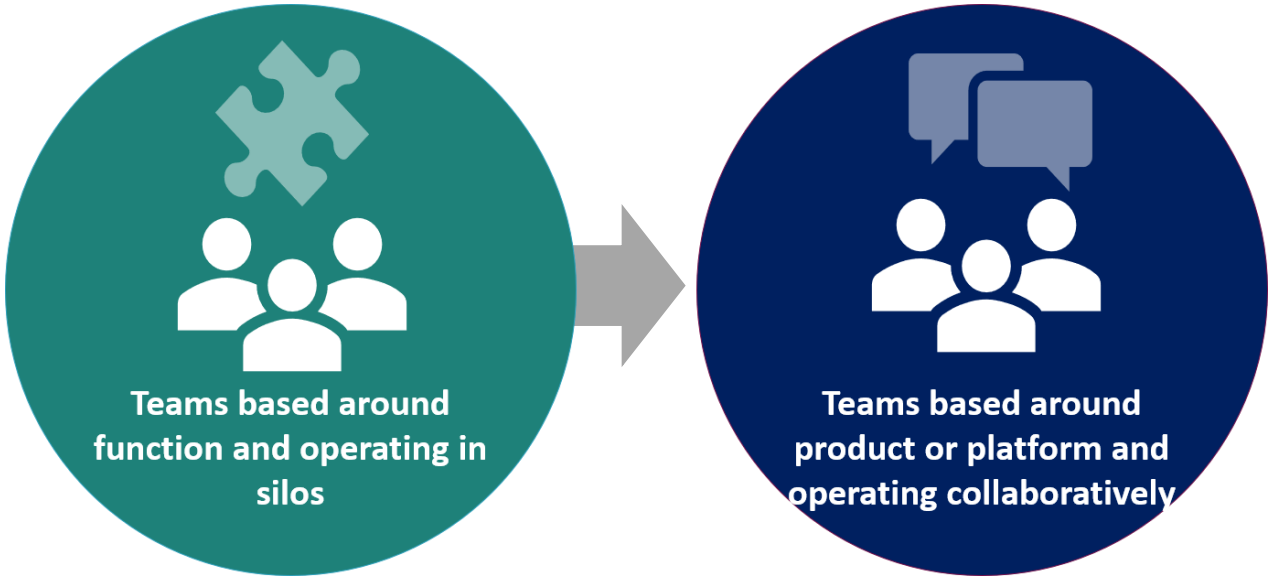
# Building A Cross-Functional Team

**Cross-functional Teams** are built with representatives of all disciplines who have responsibility for developing and deploying a service. They are designed to be empowered, self-managed and self sufficient. These teams may be temporary, assembled only to work on short-term projects. But, they may also be permanent. Most importantly, these teams resemble the adaptive, self-organizing teams.


## DevOps Requires Practical Specialists

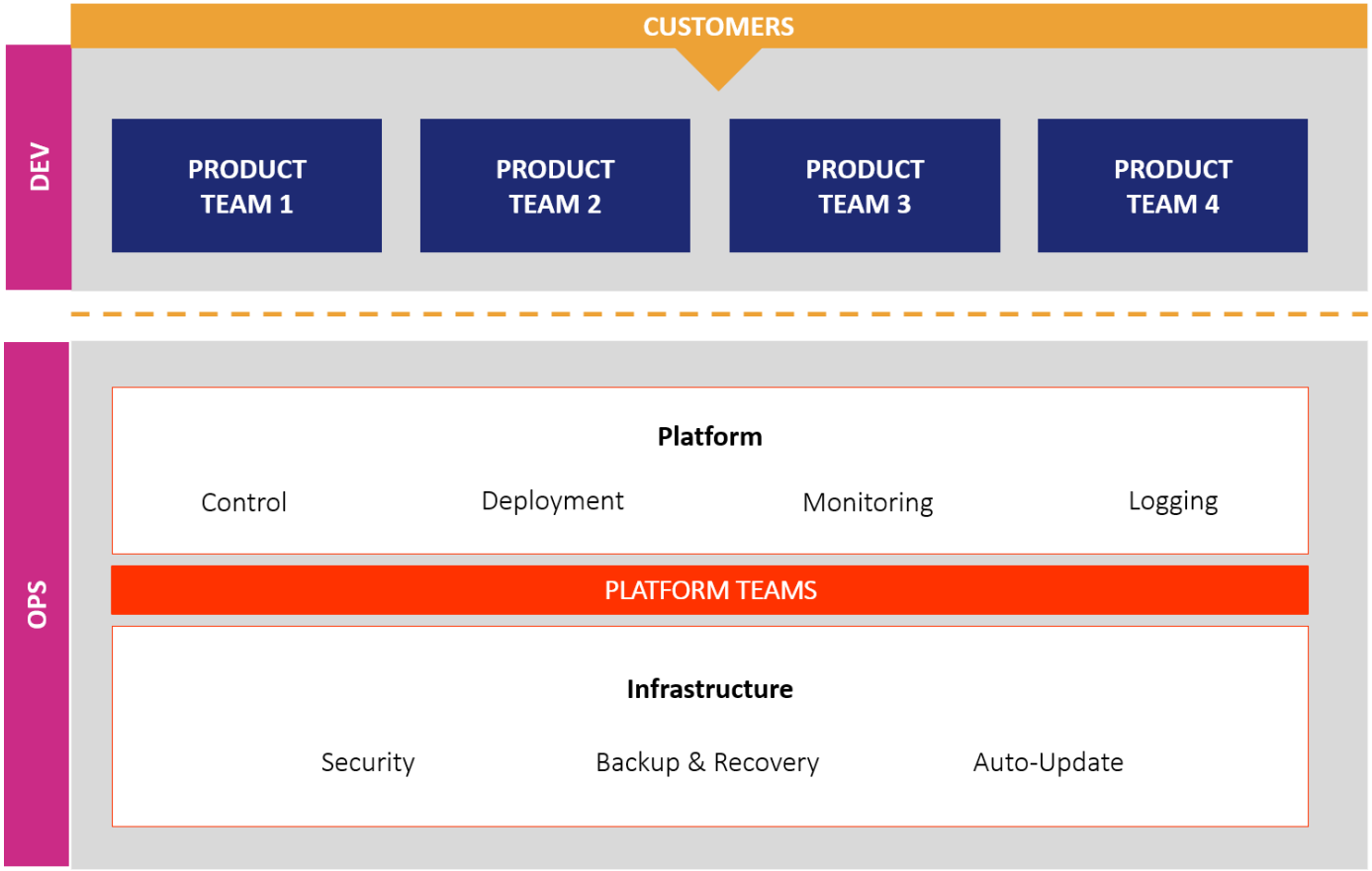
- DevOps requires practical specialists. Generalists may not be knowledgeable or skilled enough in relation to specific tasks or requirements, while pure specialists may have difficulty being asked to work outside of their specialization to take on new challenges.

# A Shift To Platform & Product Teams




- DevOps teams work best when architected around products or platforms vs. specific functions.
- Product and Platform teams require cross-functional skills. They demand more from each individual, but they allow people to dedicate themselves to fewer projects and see them through to completion, or to be responsible throughout the entire lifecycle.
- Product and Platform teams reduce hand offs, encourage feedback, and motivate and inspire team members by granting them autonomy and recognizing their broad skillsets. They address the problems of task orientation.

# Product & Platform Teams




**For more information, check out this video:**
*Spotify Engineering Culture Part 1:* https://www.youtube.com/watch?v=4GK1NDTWbkY

PeopleCert DevOps Fundamentals

# Module 4:
# The Full Stack – Processes & Practices

**Topics:**

- The Evolution Of DevOps Practices
- The 15 Essential Practices Of DevOps

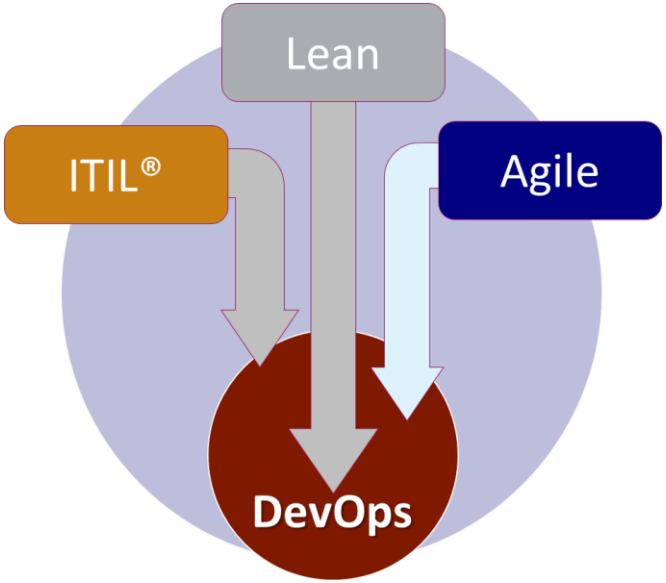
# DevOps Leverages Best Practices

- Many of the practices that are utilized by DevOps fit comfortably within the larger scope of IT Service Management, and have their origins in a variety of other ITSM frameworks and methodologies.

**Service Management** is a set of specialized organizational capabilities for enabling value for customers in the form of services.

When implementing DevOps, it is easier and faster to adopt and adapt an existing reference then to create or invent one from scratch. That is why DevOps utilizes practices from documented frameworks already in existence, including ITIL, Lean and Agile.

These quality systems are based on a common logic tree that states that to improve something it must be first defined. Once it is defined it can then be stabilized and controlled enabling measurement and monitoring. Based on the measurement gaps opportunities for Improvement are identified that provide information on how the capability should be re-defined as a new or modified reference going forward.

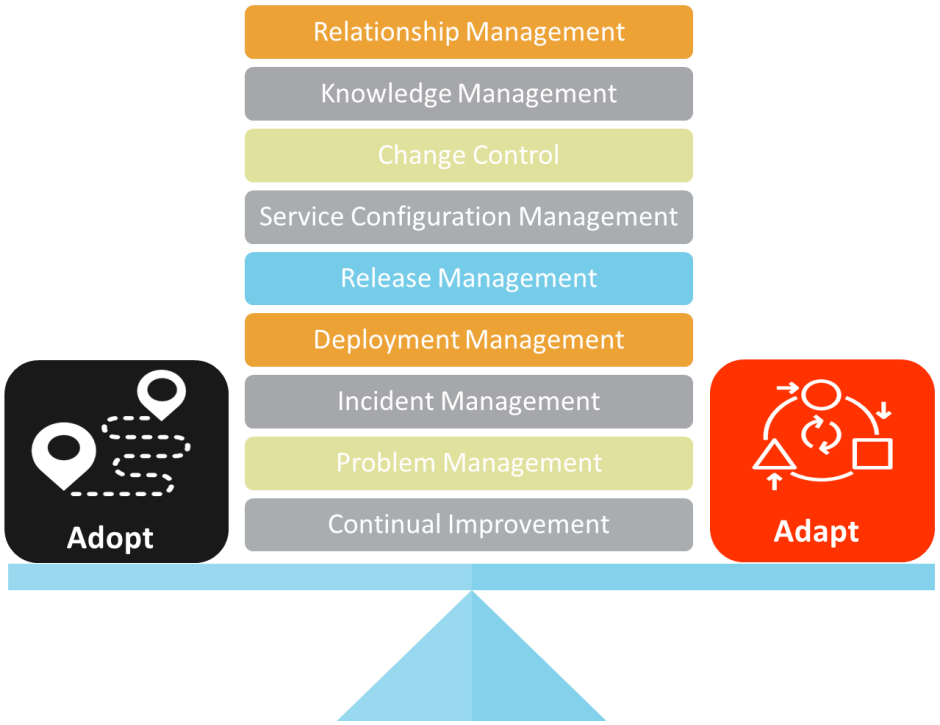



Source: *ITIL Foundation ITIL 4 Edition*

ITIL® is a registered trade mark of AXELOS Limited, used under permission of AXELOS Limited. All rights reserved.

# Adopting & Adapting ITIL Practices for DevOps

**ITIL** offers best practice guidance for IT service management.




This slide is designed to reinforce that ITIL is designed to be adopted and adapted, and is not prescriptive or vendor specific.

ITIL is not meant to be adopted as a one size fits all solution – it must be adapted based on the services it supports and the underlying IT technology.

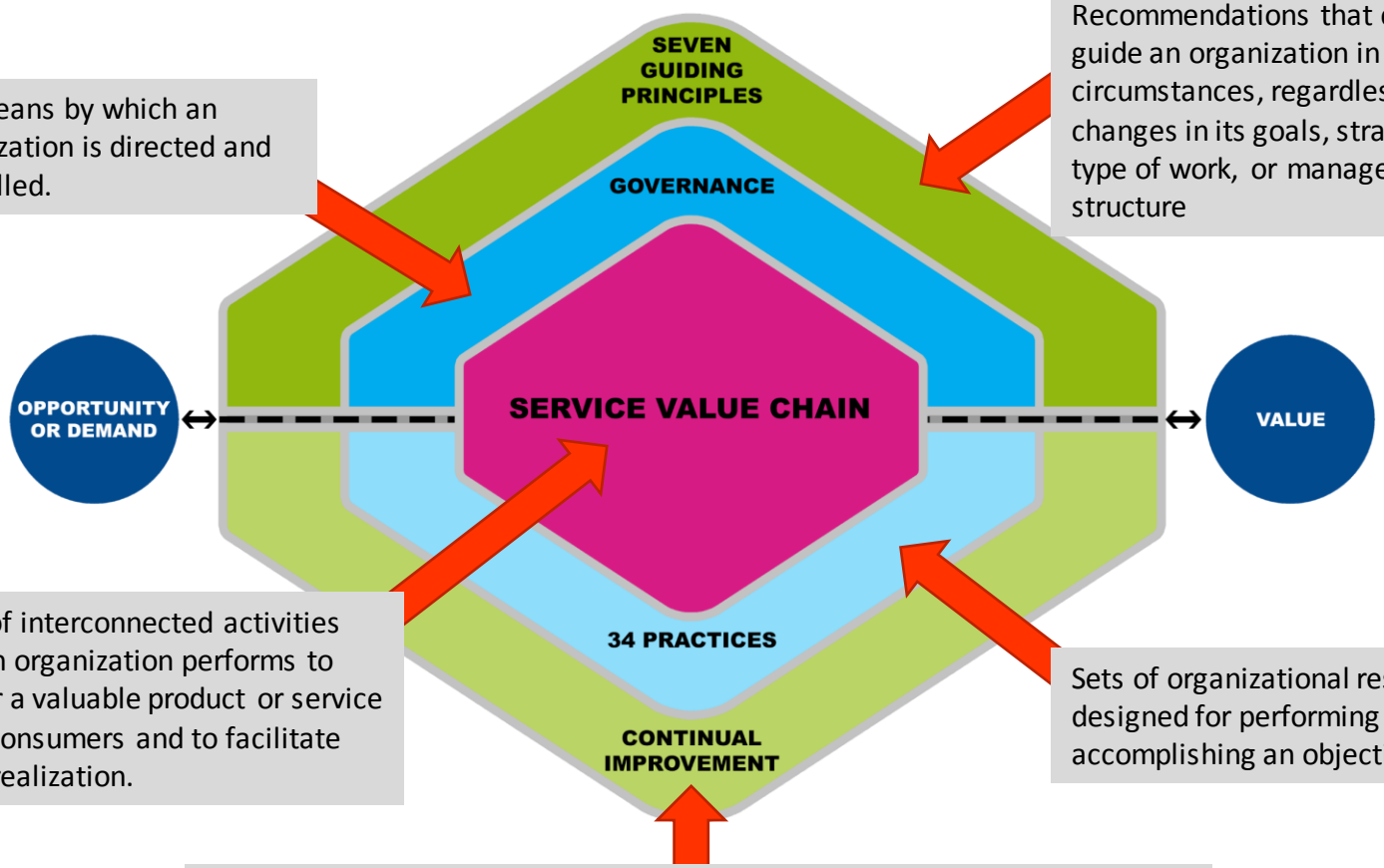
Adopting ITIL means committing to a service-oriented, customer-focused culture and an organization-wide shift.

Adapting ITIL means understanding the best practices and why they are recommended, then considering them carefully against your organizations needs, goals and objectives.

In this course, there are several ITIL practices that are discussed in ways that we have adapted for the DevOps conversation.

**The ITIL Service Value System (SVS)** at the center of ITIL provides a holistic way of understanding the inputs of opportunity and demand into IT, the elements of ITSM, and the outputs from IT (the achievement of value for the organization, customers and stakeholders).

# The ITIL Service Value System




The means by which an organization is directed and controlled.

Recommendations that can guide an organization in all circumstances, regardless of changes in its goals, strategies, type of work, or management structure

A set of interconnected activities that an organization performs to deliver a valuable product or service to its consumers and to facilitate value realization.

Sets of organizational resources designed for performing work or accomplishing an objective.

A recurring organizational activity performed at all levels to ensure that an organization's performance continually meets stakeholders' expectations.

Copyright © AXELOS Limited 2019. Reproduced under license from AXELOS Limited. All rights reserved.
(*Figure 4.1 The ITIL Service Value System – ITIL® Foundation, ITIL 4 edition, 2019)*

# The Evolution Of Lean

**Lean** is a quality system focused on flow. It is all about increasing customer value, eliminating waste, and continuous improvement.

Lean grew out of the Toyota Production System, a series of innovations implemented by Toyota following World War II. The goals at that time were to stop the constant slowdowns of work and to repair poor quality by studying each stage of their processes, to waste as little as possible of whatever resources were available, and to orient work around actual sales rather than on sales or production targets. This would increase continuity and speed of production but while also allowing Toyota to offer a wider range of products.

Today, Lean is a critical system not only in manufacturing, but has been applied to and proven to be effective within the service industry and IT as well.

**Lean is not about doing more *with* less.**
**It is about doing more *by doing* less.**

Principles Of Lean Thinking


Source: *Lean Thinking,* James P. Womack & Daniel T. Jones

# The Agile Manifesto

## Agile Values


## Agile Principles

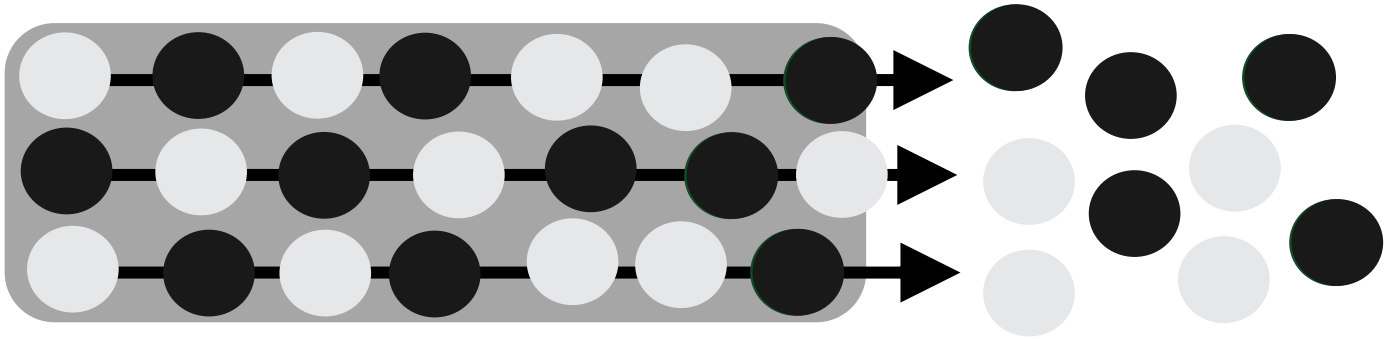
- Satisfy customers through early and continuous software delivery
- Accommodate changing requirements throughout the development process
- Deliver working software frequently
- Enable collaboration between the business stakeholders and developers throughout the project
- Support, trust and motivate the people involved
- Enable face-to-face interactions
- Measure progress as working software
- Support a sustainable development pace with Agile processes
- Ensure that attention to technical detail and design enhances agility
- Focus on simplicity
- Encourage architectures, requirements and designs with self-managing teams
- Reflect regularly on how to become more effective

Source: *The Agile Manifesto*, http://agilemanifesto.org

# DevOps Speeds Things Up




- **DevOps is the solution that works.** It utilizes Lean thinking and practices, along with Agile project management and development tools like Scrum, and takes them further to address that final constraint and bottleneck keeping IT from delivering value – that wall of confusion, the barrier between Dev and Ops that is keeping them from moving at the same speed and in the same direction.



# The 15 Essential Practices of DevOps

1. Align yourself around business value by defining **Voice Of The Customer**
2. Increase collaboration and break down silos with **Relationship Management**
3. Identify constraints and bottlenecks with **Value Stream Mapping**
4. Eliminate waste wherever possible and reduce handoffs with **Lean Process Optimization**
5. Enable knowledge sharing and create a culture of learning with **Knowledge Management**
6. Create transparency in the value stream and make work visible with **Visual Management**
7. Work in smaller batches and create feedback loops with **Agile Scrum**
8. Detect problems as the occur and fail faster with **Shift Left Testing**
9. Achieve the right balance of flexibility and stability with **Change Control**
10. Embed information and resources where you need them, when you need them with **Service Configuration Management**
11. Support automation and create flow with **Release & Deployment Management**
12. Swarm to solve incidents with **Incident Management**
13. Turn knowledge into improvements that push quality closer to the source with **Problem Management & Lean Kaizen**
14. Institutionalize daily improvement with **Continual Improvement** models
15. Embrace failure as a learning opportunity with a focus on **Antifragility**

# 1. Voice Of The Customer

## Understanding Utility & Warranty

- **Utility** represents functional requirements. It describes those requirements of a service which are “fit for purpose” – does the service do what it is supposed to do?

- Is the performance of the service supported? Are there constraints present that will inhibit its functioning? Utility is the functionality of the service to meet a specific need.


VALUE

**Warranty** represents non-functional requirements. It describes those requirements of a service which are “fit for use” – is the service actually delivered?

Warranty means that the service is available and works when the customer needs it, with appropriate capacity, continuously and securely**.**

# 1. Voice Of The Customer (Continued)

## Lean Voice Of The Customer (VOC)

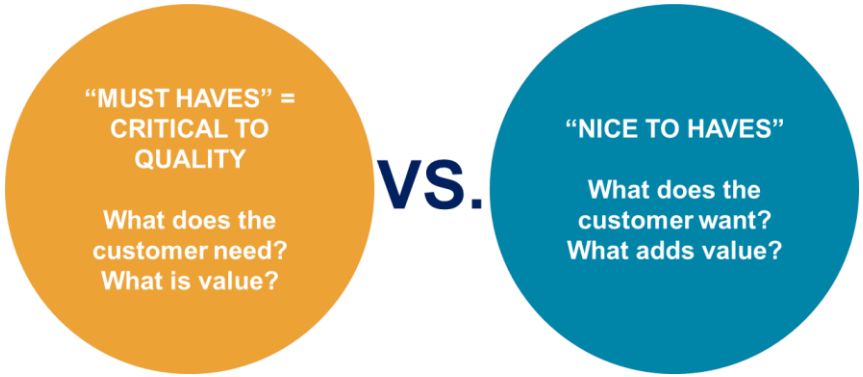
**Voice Of The Customer (VOC)** is how Lean proposes understanding the end-to-end customer experience, because ultimately the customer is the one who defines what value is.

Lean proposes considering two types of customers: those who actually use your product or service, and then those who are next in line in the Value Stream.

It is important to consider the four questions when defining value within Lean:

- Who is the customer?
- What does the customer want?
- What adds value to the customer?
- What is the customer willing to pay for?

## Understanding Critical To Quality




**Critical To Quality** items are must haves. According to the principles of Lean, it is the Critical To Quality value items that should be prioritized and focused on. If you aren't delivering those, then you aren't really delivering value.

# 2. Relationship Management

Relationship Management seeks to establish and maintain the relationship between IT and internal business stakeholders, identify business value and ensure that IT understands and is able to deliver it.


The person who really makes the Relationship Management process effective by acting as a liaison between IT and internal business partners and stakeholders

The role of the Relationship Manager is generally full-time as it is critical to maintaining the necessary relationships and ensuring that the services provided by IT are fit for purpose and deliver the necessary business value.

This role is sometimes seen as creating silos because it is perceived that they enable the business and IT to avoid communication. Nothing could be further from the truth. The Relationship Manager can and should be a role that enables collaboration by acting less as a middleman and more as a facilitator, making introductions and ensuring that both sides understand each other.

# 3. Value Stream Mapping

## What Is A Process?

A **Process** is a structured set of activities designed to accomplish a specific objective. They take one or more defined inputs and turn them into defined outputs.

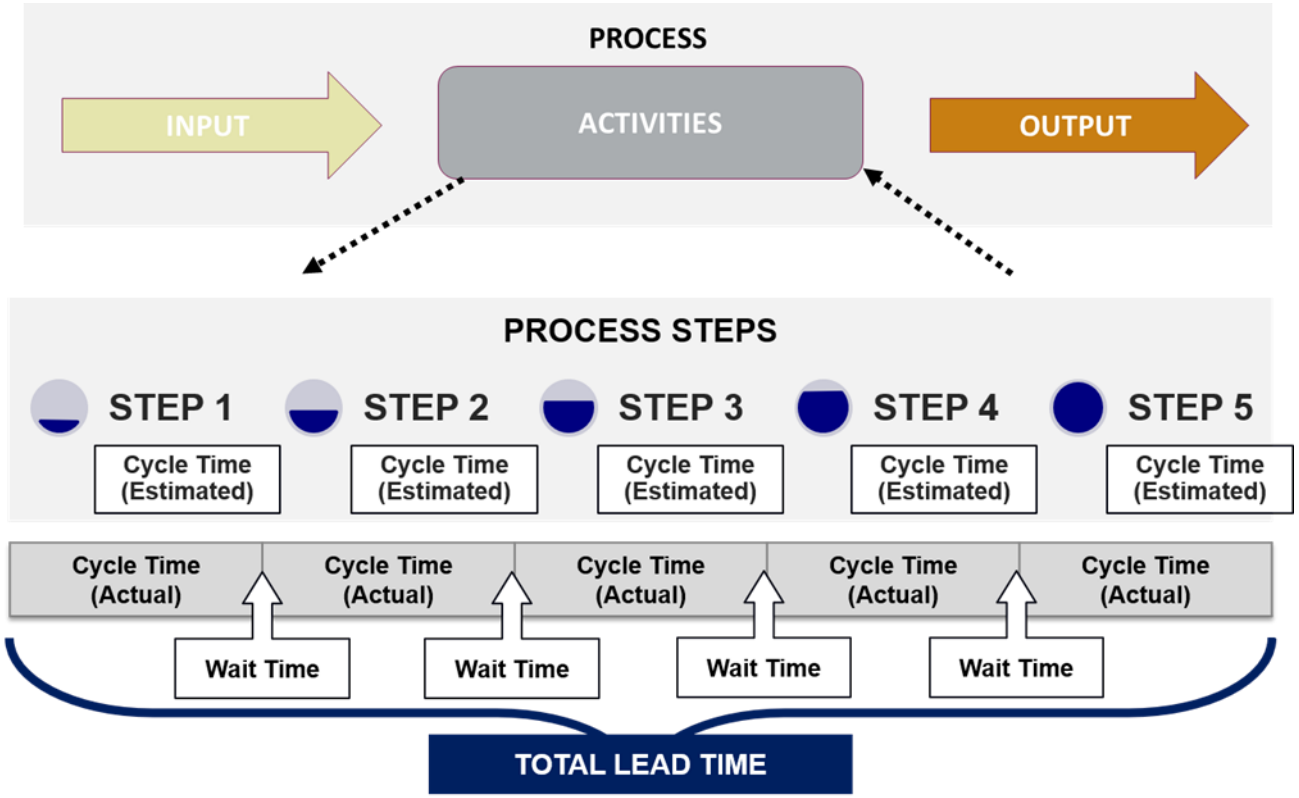

## The Value Stream

The **Value Stream** provides a holistic view of IT as delivering end to end business value. Understanding not only your own place within the value stream but the value stream as a whole is essential for leadership and governance.

One of the key principles of Lean is to map the value stream – it means measurement and metrics, using data to focus in on particular problem areas or determine where the waste is and why it is happening so that you can then prioritize effectively and find those areas where a quick win is possible.

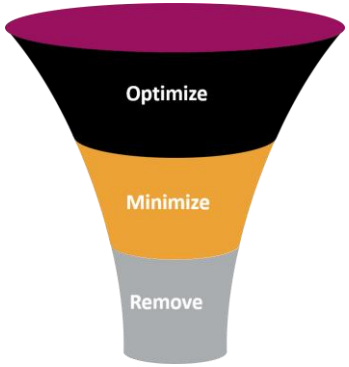
# 3. Value Stream Mapping (Continued)

- **Cycle Time** is the time spent actually creating products or services.
- **Process Time** is the total time spent actually creating products or services.
- **Wait Time** is the time spent waiting for next steps to begin.
- **Lead Time** is the time between input and output – the Process Time and Wait Time totaled together.



# 4. Lean Process Optimization

- **Value-Add** work should be optimized. It is only work that the customer actually experiences and sees as value. It is what the customer is willing to pay for.
- **Necessary Non-Value-Add** work should be minimized. This is work that is not value-add, but that must be done.
- **Non-Value-Add** work should be removed.




## Lean Waste: "Muda"

**T**ransporation
**I**nventory
**M**otion
**W**aiting Time
**O**ver-Production
**O**ver-Processing
**D**efects & Rework

& Non-Utilized Talent

## Other Forms Of Value Loss

**"Mura"**
**Variances & Variation**
Examples include: variability in volume, spread in the outcome of processes

**"Muri"**
**Overburden & Inflexibility**
Examples include: inability to scale up or down with demand, fixed service time frames

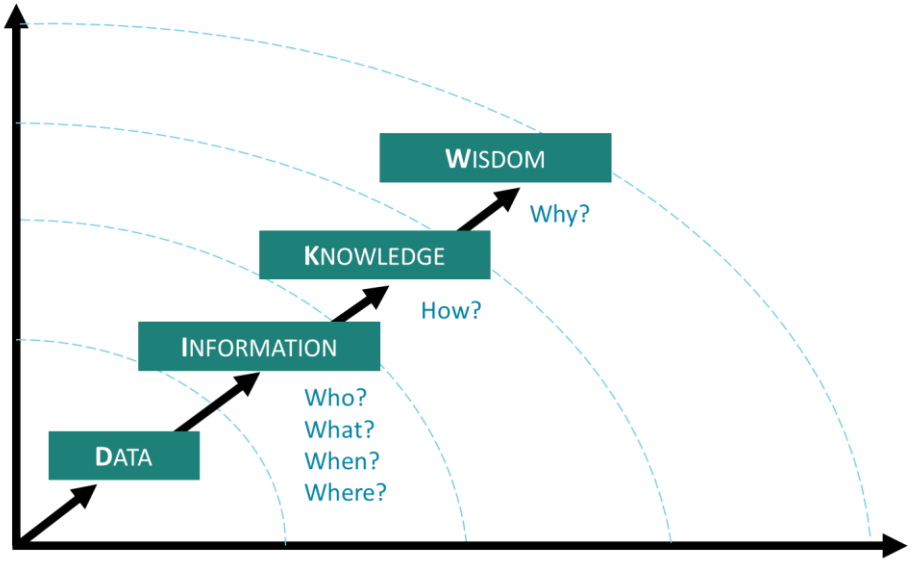
# 5. Knowledge Management

**Knowledge Management** is the practice of embedding and making accessible the essential knowledge that is required to support a DevOps environment and ensure stable, effective services.

- The purpose of Knowledge Management is to:
  - Share perspectives, ideas, experience and information
  - Ensure these are available in the right place, at the right time, to enable informed decisions
  - Improve efficiency by reducing the need to rediscover knowledge

## The DIKW Model

The **DIKW Model** helps us to understand the difference between a lack of data or information and a lack of knowledge or wisdom.



# 5. Knowledge Management (Continued)

- **Data** is objective facts, observations, signs or symbols. It has no context or meaning and is not useable until context is applied.
- **Information** is organized or structured data which has been given a context or purpose.
- **Knowledge** is information framed by individual experiences, values, insight or intuition.
- **Wisdom** requires higher level contexts and relational thinking.

## Examples of Knowledge Management Systems

| | |
|---|---|
| Collaborative software technologies, such as Slack or an internal wiki | Knowledge databases, such as a document management system |
| Mentoring and job shadowing | Embedding diverse skillsets on cross-functional teams |
| Informal gatherings such as knowledge fairs to brainstorm ideas or teach others | Visual Management tools such as Kanban |

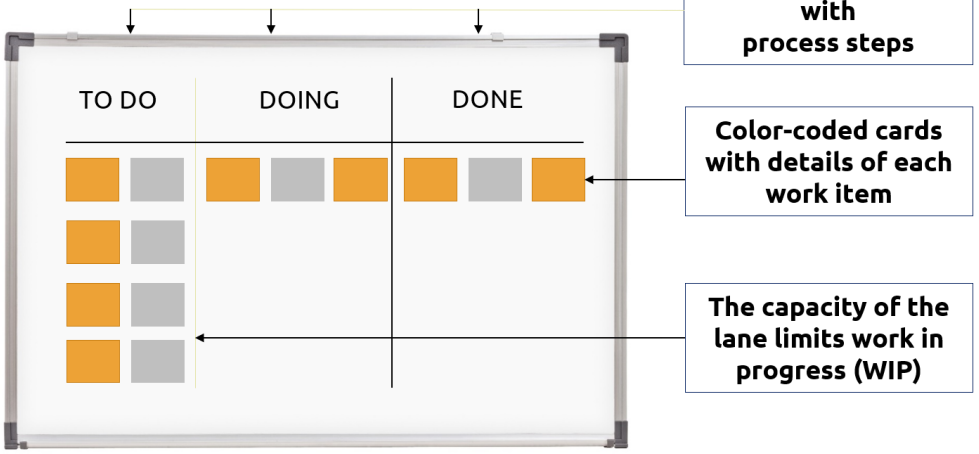
# 6. Visual Management

**Visual Management** is a tool used to make the control and management of an organization as simple as possible. The practice involves using visual controls in plain sight rather than text-based ones, where it is easier to interpret, remember and keep at the forefront of one's mind.

- Visual Management has two aims:
  - Increase the efficiency and effectiveness of a process and a value stream by making the steps visible.
  - Make problems, abnormalities, or deviations from standards visible to everyone. When these deviations are visible and apparent to all, corrective action can be taken to immediately correct these problems.

## Utilizing Kanban

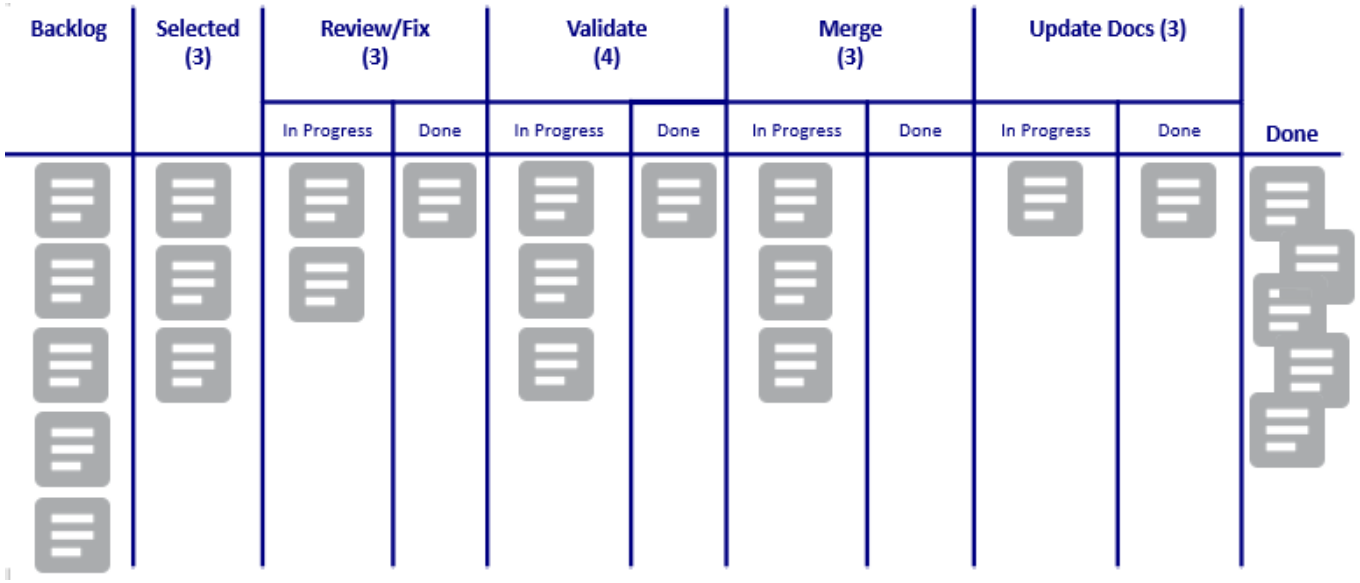
**Kanban** emerged in the 1940s as part of the initial evolution of Lean manufacturing. It provided a way for assembly line workers to notify downstream partners of demand for parts or other work. This allowed for transparency and increased communication, and it standardized processes.



# 6. Visual Management (Continued)

## Push vs. Pull Systems & WIP Limits

In a **Pull System**, products or services are pulled through the process by customer demand, in contrast to a push system, in which products or services are pushed through the process based on forecasted demand. Pull systems reduce waste in processes.




**Work In Progress Limits** are specific caps or stopping points for adding new work that ensure that the amount of work never exceeds the capacity of the system to complete it.

# 7. Agile Scrum

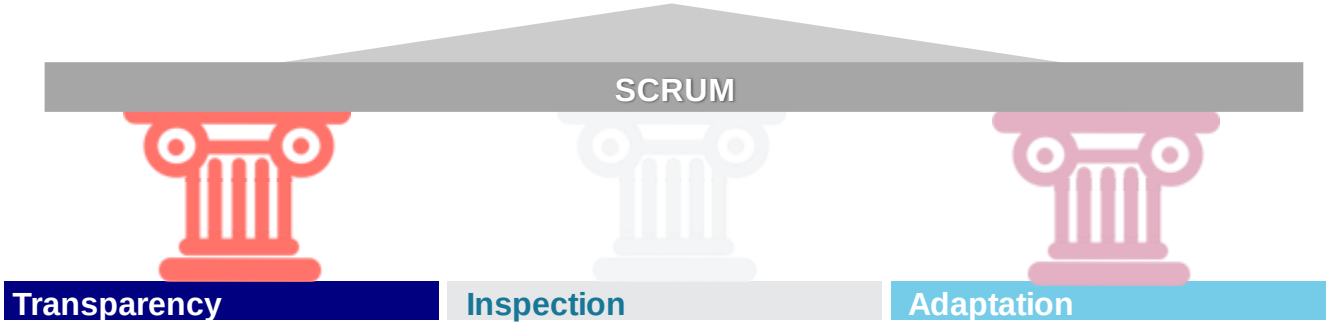
## Understanding Scrum

While for many, Agile and Scrum are now synonymous, Scrum has actually been used since the early 1990s to manage complex product development. It is an adaptable process framework within which various processes, tools and techniques can be applied. It promotes the development of products in an iterative way that results in more frequent releases with the highest quality outcomes possible.

Scrum is built on three essential pillars.

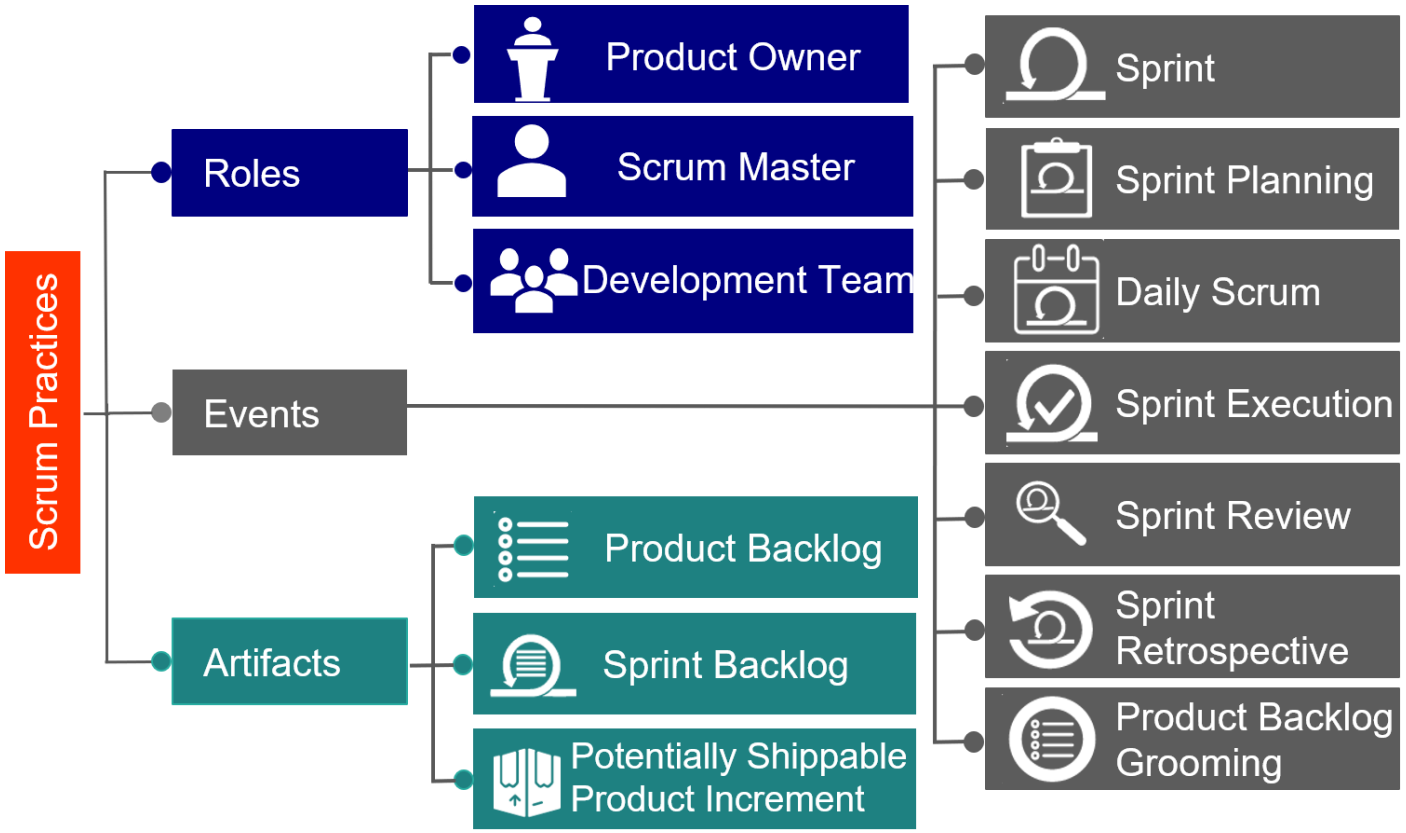
- It is about **inspection**. In Scrum, inspections of progress and artifacts are frequently and consistently performed throughout the project and feedback is gathered to ensure that everything is still on track.
- It is about **transparency**. It means making as much of the process as possible visible to those responsible for the outcome, and having everyone in agreement on a standard definition of "done".
- It is about **adaptation**. When an inspection occurs, if any aspects of a process will result in defects, errors or reduced quality, adjustments are made immediately. There are formal events that ensure this.

Scrum is about learning, not predicting.



# 7. Agile Scrum (Continued)

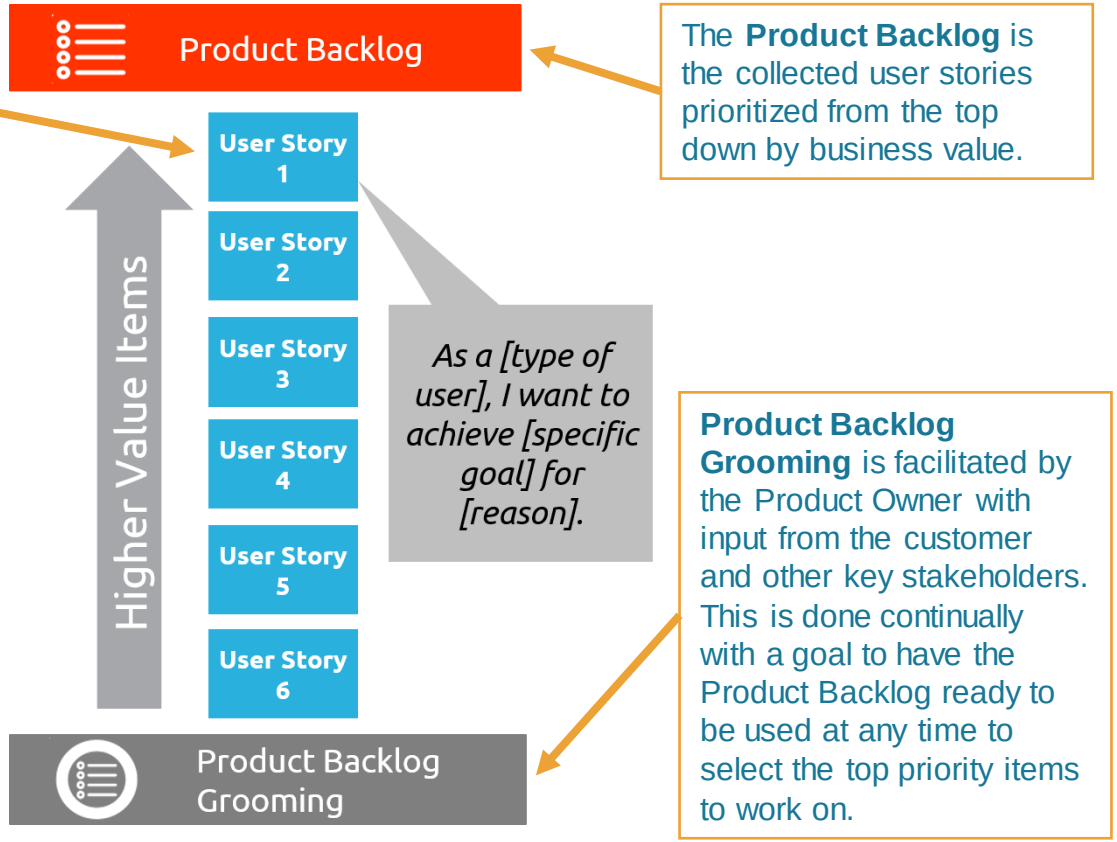
Scrum At-A-Glance



# 7. Agile Scrum (Continued)

## The Product Backlog

The **User Stories** are product features or bug fixes that are looked at from the perspective of the end user.



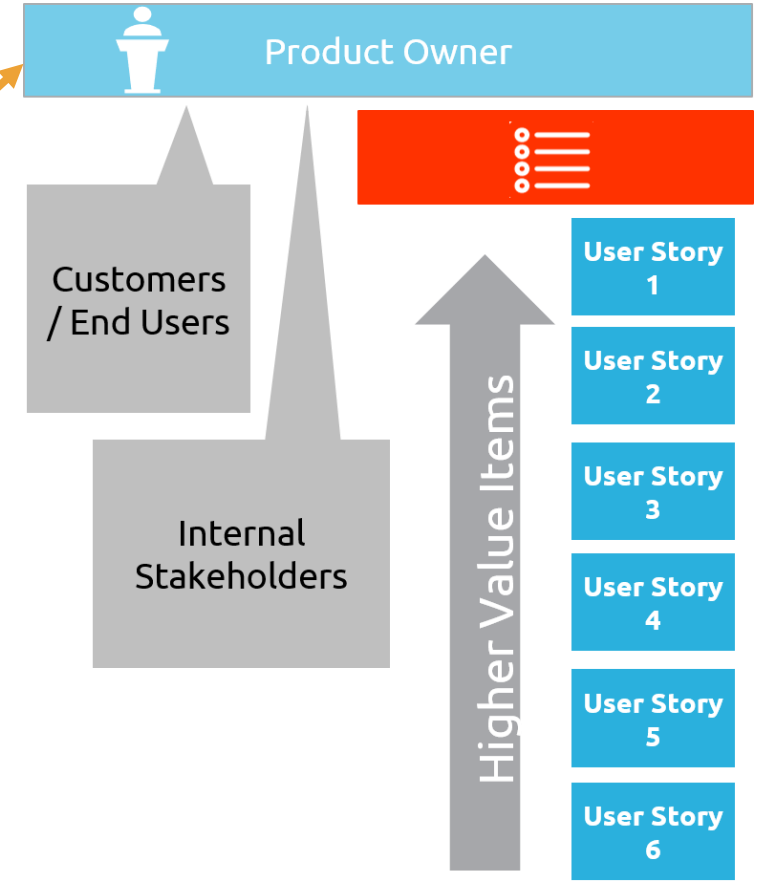

The **Product Backlog** is the collected user stories prioritized from the top down by business value.

**Product Backlog Grooming** is facilitated by the Product Owner with input from the customer and other key stakeholders. This is done continually with a goal to have the Product Backlog ready to be used at any time to select the top priority items to work on.

# 7. Agile Scrum (Continued)

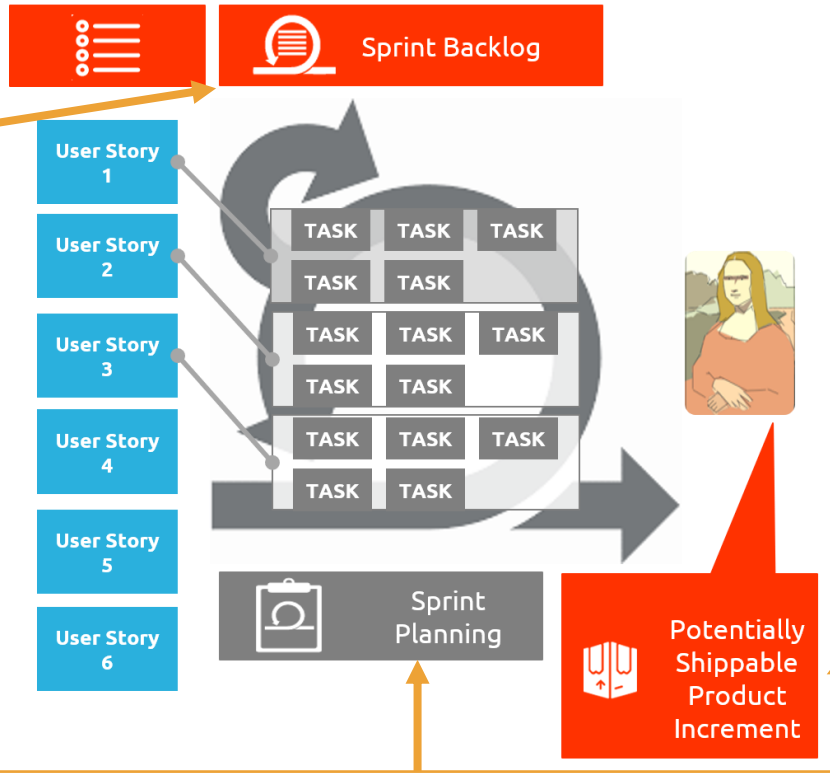
The Product Owner

The **Product Owner** is responsible for creating and maintaining the Product Backlog. They are in constant communication with the customer and collaborate with the team.



# 7. Agile Scrum (Continued)

## The Sprint Backlog




The **Sprint Backlog** is the result of the Product Backlog being refined and prioritized, a focused list on items to be worked on during a Sprint.

**Potentially Shippable Product** Increments are the items completed based upon the Sprint Backlog according to an agreed "Definition of Done", the acceptance criteria around both functional and non-functional requirements.

The **Sprint Planning** meeting is between the Product Owner and the Development Team. The Product Owner presents and describes each item in the Product Backlog beginning at the top, and the Development Team questions each item, determining key tasks necessary for its completion and estimating the effort involved.

# 7. Agile Scrum (Continued)

The Scrum Master & Development Team

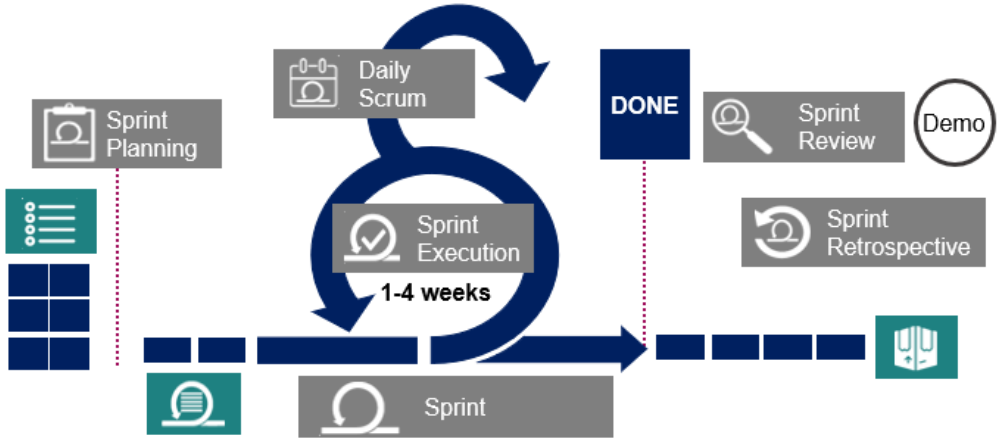

The **Scrum Master** ensures that the Scrum Framework is followed entirely and correctly and can fill in for the Product Owner. They facilitate as a coach, and are trained and able to solve problems. They clarify any and all questions anyone has about Agile and Scrum, and can fill in for the Product Owner if they are absent for any reason to take over maintenance of the Product Backlog.

The **Development Team** is a set of technical experts who are self-organized and cross-functional, able to develop solutions to problems. They own and manage the Sprint Backlog, and are accountable to build, test and implement the Product Backlog items brought over.

# 7. Agile Scrum (Continued)

## Scrum Events & Timeboxing

In Scrum all events are timeboxed. This means they are done within a previously agreed maximum amount of time. When the time allocated is reached, work is stopped and accomplishments are evaluated.

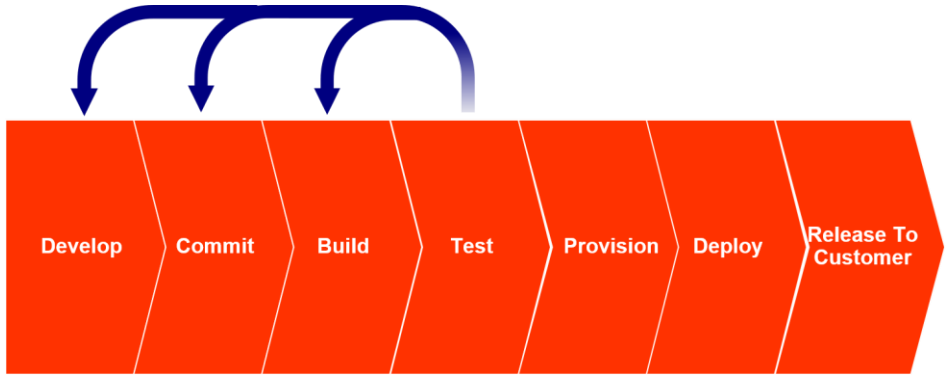



- The **Sprint** is the timebox including Sprint Planning, Sprint Execution, Sprint Review, and Sprint Retrospective.
- **Sprint Execution** begins after Sprint Planning and ends when time for Sprint Review and Retrospective is subtracted from the overall Sprint. During Execution, items in the Sprint Backlog become WIP and are worked on until completed in accordance with a Definition of Done.
- **Daily Scrum** is a meeting held for the Development Team to be transparent and accountable for their progress
- The **Sprint Review** is held when Sprint Execution ends. All items completed in accordance with the Definition of Done are demonstrated to the Product Owner, customer and other key stakeholders. This allows everyone to be in agreement on accepting the final products.
- **The Sprint Retrospective** is for the Development Team to reflect on the past Sprint and make any adjustments or refinements (inspect and adapt) to how they conduct the Sprint.

# 8. Shift Left Testing

## The Shift Left

**Shift Left** ensures that quality is built in earlier in the development process so that issues are detected earlier and can be resolved, and defects or errors don't impact production.




## Putting Quality First

The Quality First model advocates limiting scope and functionality, but providing instead a high quality product that meets the non-functional requirements, and then introducing more features and functionality over time through the use of Sprints. Doing this allows you to be more flexible with the order in which features are delivered. The Quality First model underpromises and overdelivers rather than the other way around.

You are ALWAYS delivering high quality.


Source: *Practice To Perfect: The Quality First Model*

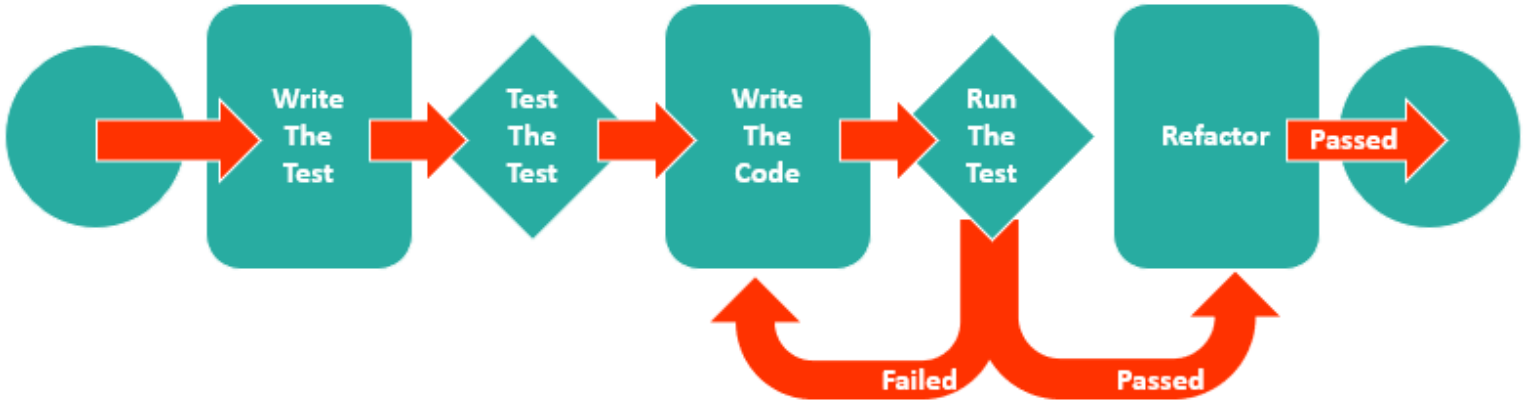
# 8. Shift Left Testing (Continued)

## Test Driven Development (TDD)

**Test Driven Development (TDD)** is the practice of preparing test scenarios before a program is written so that the goal for the programmer is to write something that can pass the exact test.

The benefits of TDD are:

- Programmers know exactly what others expect from them and can work more productively;
- Programmers are encouraged to create the simplest possible program, just to pass the test. This is desirable in Agile environments, since we do not want to waste time creating something perfect, while all we need is a good enough solution to do the job. However, we usually need to have an eye on refactoring to make sure that the solution is still "good enough".



# 9. Change Control

**Change Control** is an ITIL practice that controls the lifecycle of all changes, enabling beneficial changes to be made with a minimum disruption to IT services. It optimizes overall business risk.

## Change Types

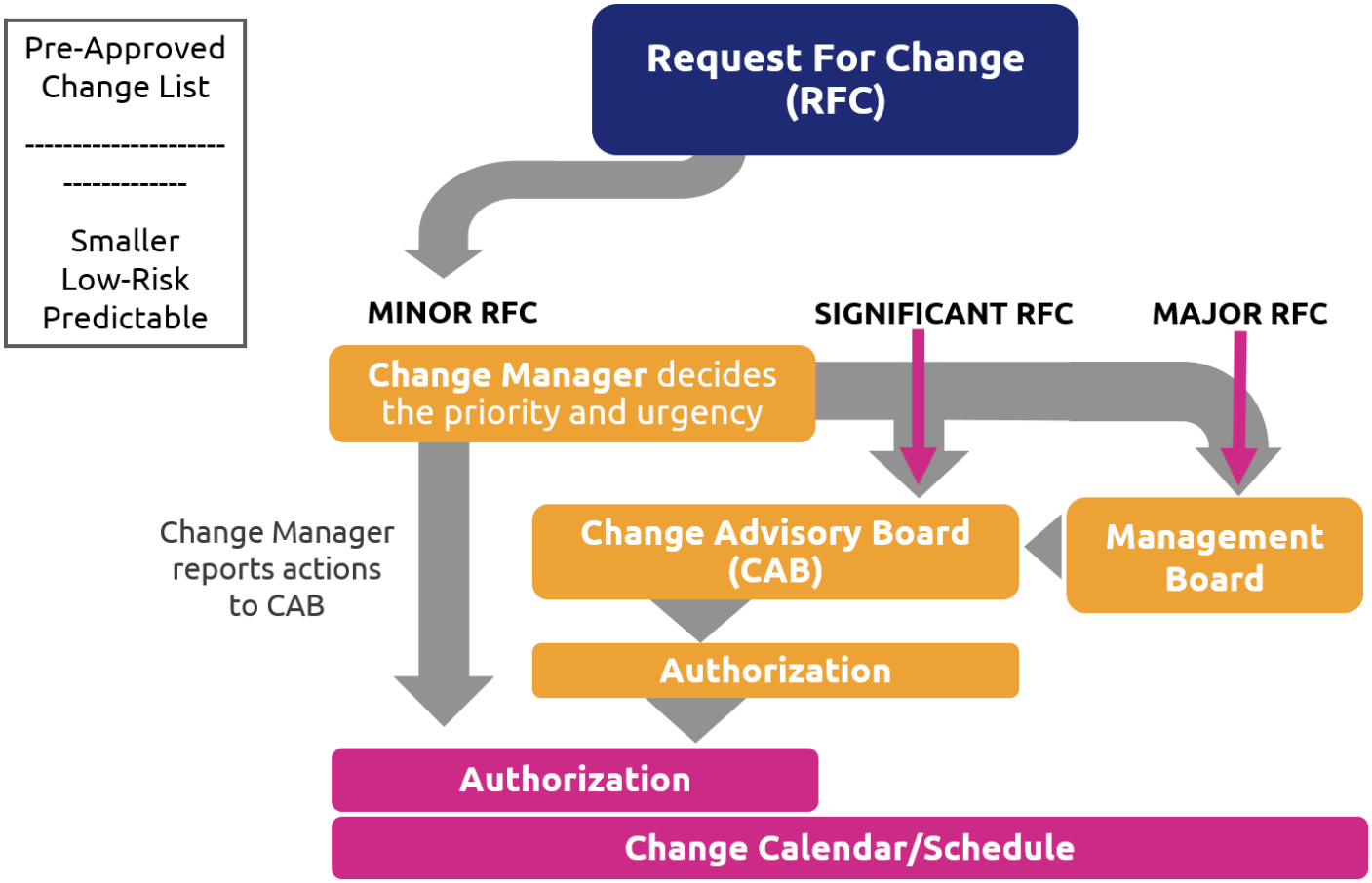
**Standard Changes** are low risk, generally common and have a specific associated procedure or work instruction that has been proven effective in previous implementations. These changes can be pre-authorized.

**Emergency Changes** need to be implemented immediately as a response to a major incident or critical need.

**Normal Changes** are any changes that are not standard or emergency.

# 9. Change Control (Continued)

The Traditional Change Approval Process



# 9. Change Control (Continued)

## Normal Change Approvals for DevOps

### Traditional Normal Change Model

Create and Record RFC → Review and Approve RFC → Assess and Evaluate RFC → Authorize and Monitor Build and Test → Authorize and Monitor Deployment → Review and Close RFC

### Adaptive Normal Change Model

Create and Record RFC → Review and Approve RFC → Authorize and Monitor Deployment → Review and Close RFC

# 10. Service Configuration Management

**Service Configuration Management** is an ITIL practice that is crucial for DevOps because it is essential for automation. This process is about maintaining knowledge of the current state of every component in the infrastructure.

- In a DevOps environment, Service Configuration Management encompasses the practices and tooling that automate delivery and operation of infrastructure. Tools are used to:
  - Model infrastructure;
  - Continually enforce desired configurations;
  - Remediate unexpected changes or "configuration drift", differences over time between primary and secondary infrastructure configurations.

## Infrastructure As Code (IAC)

Automated Service Configuration Management is achieved in DevOps by treating infrastructure as code that can be managed with the same tools and processes that developers use, such as version control, code review, automated testing and small deployments. Infrastructure As Code is based on a few practices:

- Using definition files and ensuring that all configuration is defined in executable definition files
- Implementing self-documenting systems and processes rather than instructions for humans to execute.
- Versioning everything and keeping all code in your source control so that every change is recorded.
- Continuously testing systems and processes and rapidly finding errors in infrastructure configuration.
- Emphasizing small changes rather than big batches, so it's easier to find errors and revert if needed.
- Keeping services available continuously.

Source: https://martinfowler.com/bliki/InfrastructureAsCode.html

# 11. Release & Deployment Management

**Together, Release & Deployment Management** provide guidance related to planning, scheduling and controlling releases as they transition from the development environment into production.

**Release** refers to the actual making of new and changed services and features available.

**Deployment** refers to the moving of new or changed service components to live environments

## Release & Deployment Management Techniques

**Automate Service Configuration Management with IAC**

Service Configuration Management that is automated and utilizes IAC supports your Release Management process and helps to increase reliability and reduce error by automating the process of installing and configuring software. It makes the release process more repeatable because you can run scripts over and over in both testing and production environments while maintaining version control.

**Rebuild Servers Regularly**

Rebuilding servers regularly by utilizing virtual machines means being able to easily destroy and recreate your servers. This reduces the impact of configuration drift and forces you to ensure that your infrastructure code contains accurate details of how servers need to be configured.

**Implement Zero-Downtime Deployments**

Implementing zero-downtime deployments means finding ways to release your software without affecting end users. This is critical when you are rebuilding your servers regularly, because otherwise the required down-time would be unacceptable.

Source: https://techbeacon.com/3-devops-techniques-stress-free-release-management

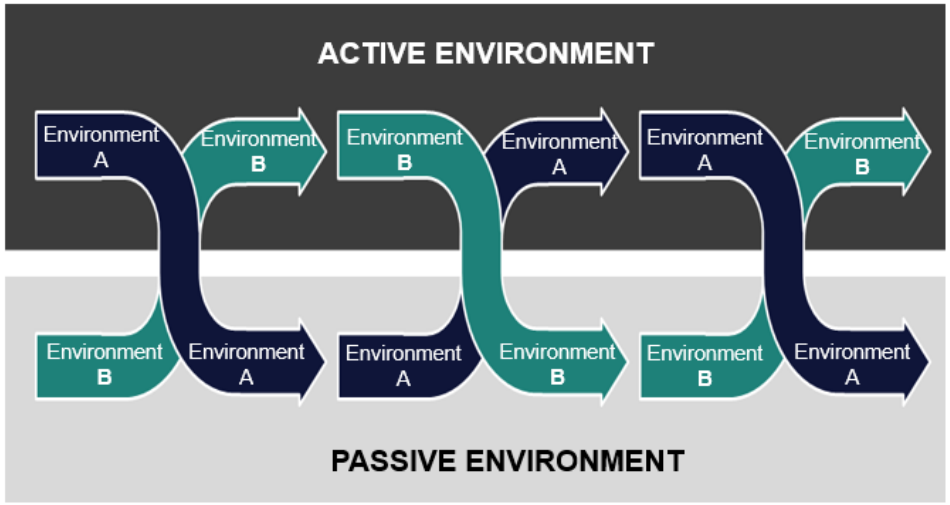
# 11. Release & Deployment Management (Continued)

## Blue-Green Deployment

**Blue-Green Deployment** means setting up two identical production servers, and directing all of the user traffic going to only one of the server sets – an "Active" environment that is in production. This then leaves you with a "Passive" environment that is only used in development that users cannot see or use.

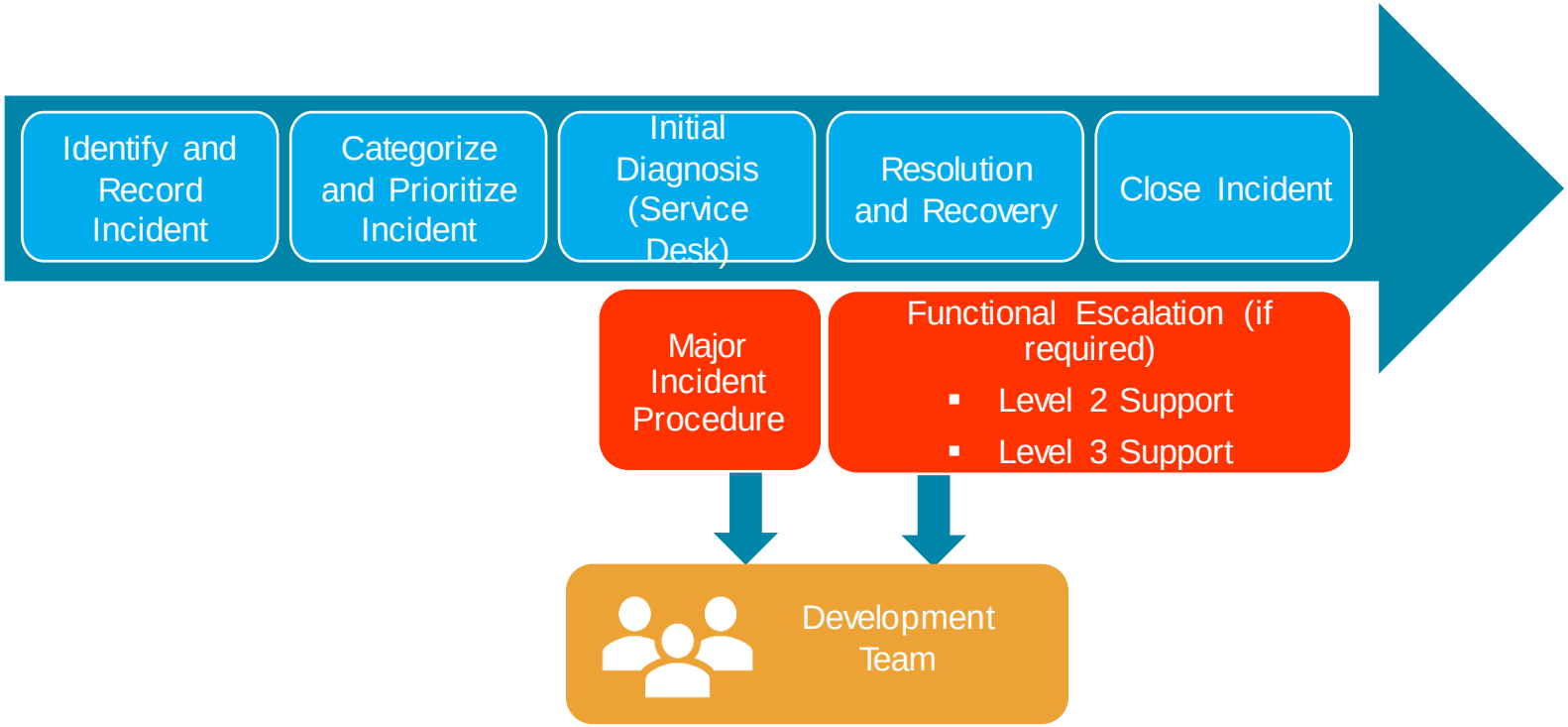
With **Blue-Green Deployment** you can deploy changes to the servers you are not using, destroying and rebuilding the servers as needed, and nobody will be affected if it goes wrong. Meaning you are no longer limited to deploying outside of normal hours to minimize the impact of down-time or having to worry about the impact of the change on end users.

When you are certain that the passive environment is working as required, you redirect user traffic to the new set of servers. Those servers become the new active environment, and you then use the previously active environment as a passive one to begin the process again.



# 12. Incident Management

**Incident Management** is an ITIL practice that restores normal service operation as quickly as possible in the event of an incident. It minimizes the adverse impact on business operations. It ensures that the best possible levels of service quality and availability are maintained.



# 12. Incident Management (Continued)

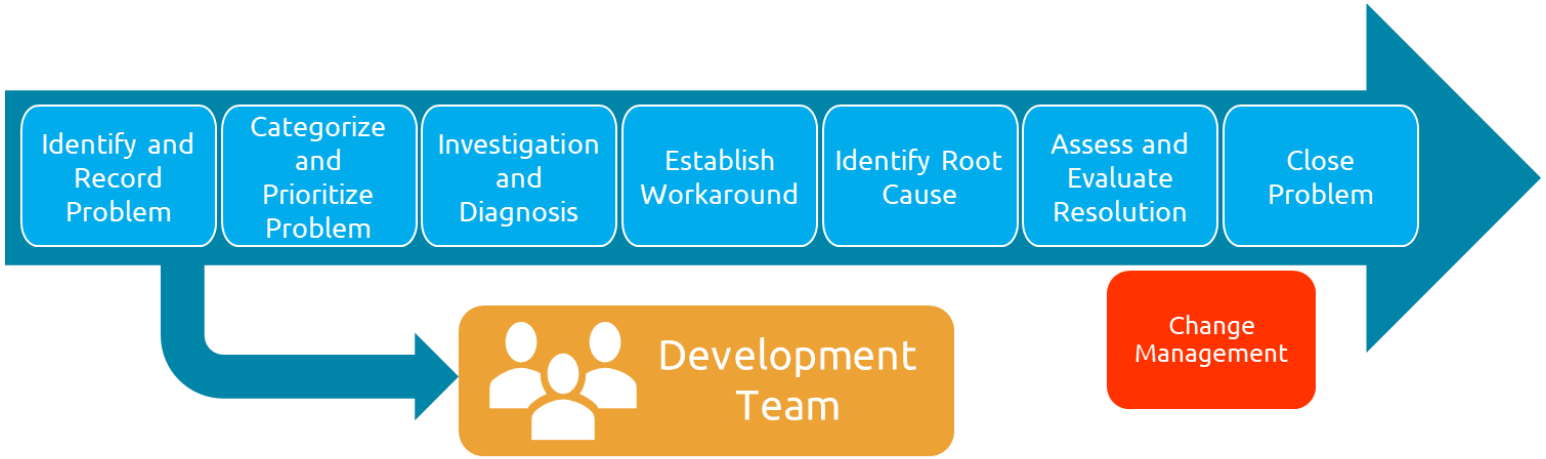
## Incidents vs. Problems

- It's important to note the difference between what ITIL classifies as an Incident vs. a Problem. As per ITIL, an **Incident** is "an unplanned interruption to a service, or a failure of a component of a service that hasn't yet impacted service." A **Problem**, however, is a condition of one or more incidents. Whether an incident becomes labeled a problem is a choice in ITIL based upon a variety of considerations.

| Incident | Problem |
| --- | --- |
| ▪ An unplanned interruption to an IT service or reduction in the quality of an IT service<br>▪ Failure of a Configuration Item that has not yet impacted service | ▪ The underlying cause of one or more Incidents<br>▪ The cause is not usually known at the time a problem record is created, and the Problem Management process is responsible for further investigation |

# 13. Problem Management & Kaizen

**Problem Management** is an ITIL practice which works to prioritize, identify and, if possible, resolve the root cause of one or more incidents in order to eliminate them or minimize their impact.




- The more accountable the Development Team is made when something breaks, the less likely they will be to break it to begin with. It decreases a silo mentality and encourages collaboration to involve them. And, being involved with Incidents and Problems also encourages the experimentation and learning that are central to the Third Way.

# 13. Problem Management & Kaizen (Continued)

## Problem Solving With Lean Kaizen

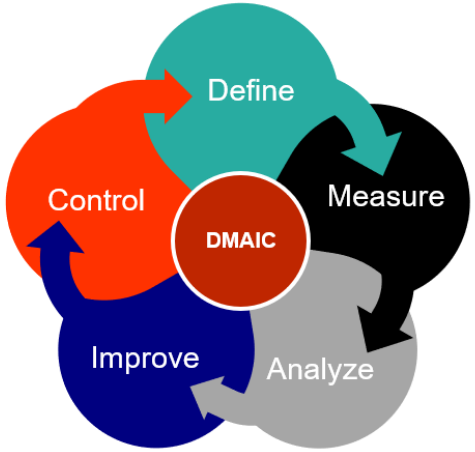
KAIZEN

改善

CHANGE FOR THE BETTER

**Kaizen** is a structured approach for solving problems that is about improving flow and processes incrementally, with an attitude or mindset that encourages everyone at every level of an organization to look for small ideas which, if possible, can be implemented easily and quickly. Kaizen should be part of the daily culture of an organization.

## The DMAIC Cycle For Improvement



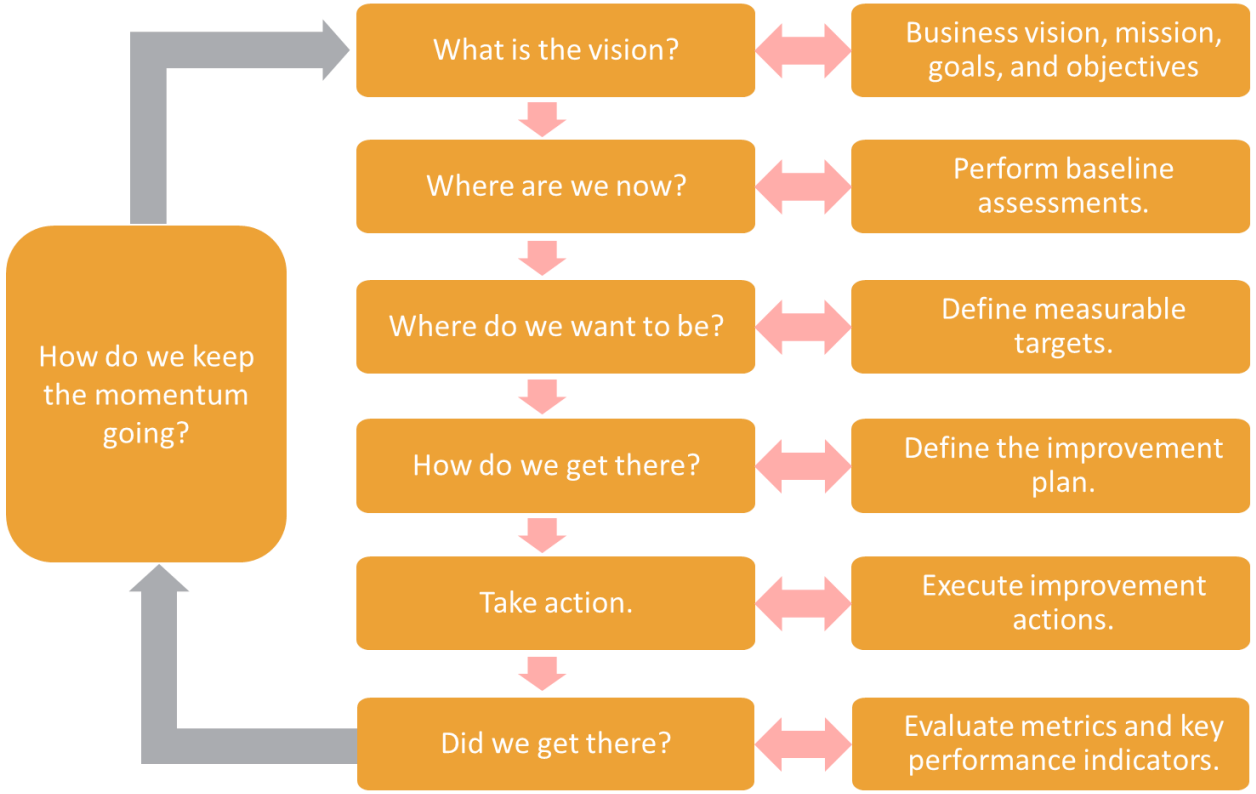

- **The DMAIC Cycle** provides guidance for continual improvement throughout five stages:

1. Define: Create understanding of the problem with everyone involved
2. Measure: Measure data and gather facts
3. Analyze: Identify root causes and define hypotheses
4. Improve: Prioritize and implement actions
5. Control: Ensure sustainability and stability of the improved process

# 14. Continual Improvement

**Continual Improvement** is essential as a solution to the business and IT value delivery problems. It is a way to keep pace with the value the business demands and the pace with which competition is now emerging and the world is changing.

It demands not just that you continually improve, but that you measure against that improvement and approach that improvement in a strategic way that focuses your attention on those constraints and areas where it will have the most impact.




Copyright © AXELOS Limited 2019. Reproduced under license from AXELOS Limited. All rights reserved.
(Figure 4.3 The Continual Improvement Model – ITIL® Foundation, ITIL 4 edition, 2019)

# 14. Continual Improvement (Continued)

## Continual Improvement Register

The **Continual Improvement Register** is a document which provides structure and visibility to continual improvement efforts, and which records and categorizes each opportunity for improvement by timeframe, effort level, urgency, impact and priority.

| Reference ID | Date Logged | Logged By | Effort Level | Timeframe | Description | Reason | Urgency | Impact | Priority |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

## Continual Improvement Supports Everything In DevOps

Continual Improvement should be implemented with the goal of improvement that supports the Full Stack and the Three Ways, with objectives that:

Align goals and priorities around delivering business value

Balance improvements to time and cost efficiencies with improvements to quality and effectiveness

# 15. Antifragility

Antifragility Goes Beyond Resilience

# 15. Antifragility

Embracing Failure As A Learning Tool


In 2011, Netflix set loose "Chaos Monkey", a software that simulates failure by randomly shutting down production servers.

NETFLIX

**THE SIMIAN ARMY**

This evolved into a collection of cloud testing tools to simulate other potential failures that the company could face in production. They call this collection of tools "The Simian Army.

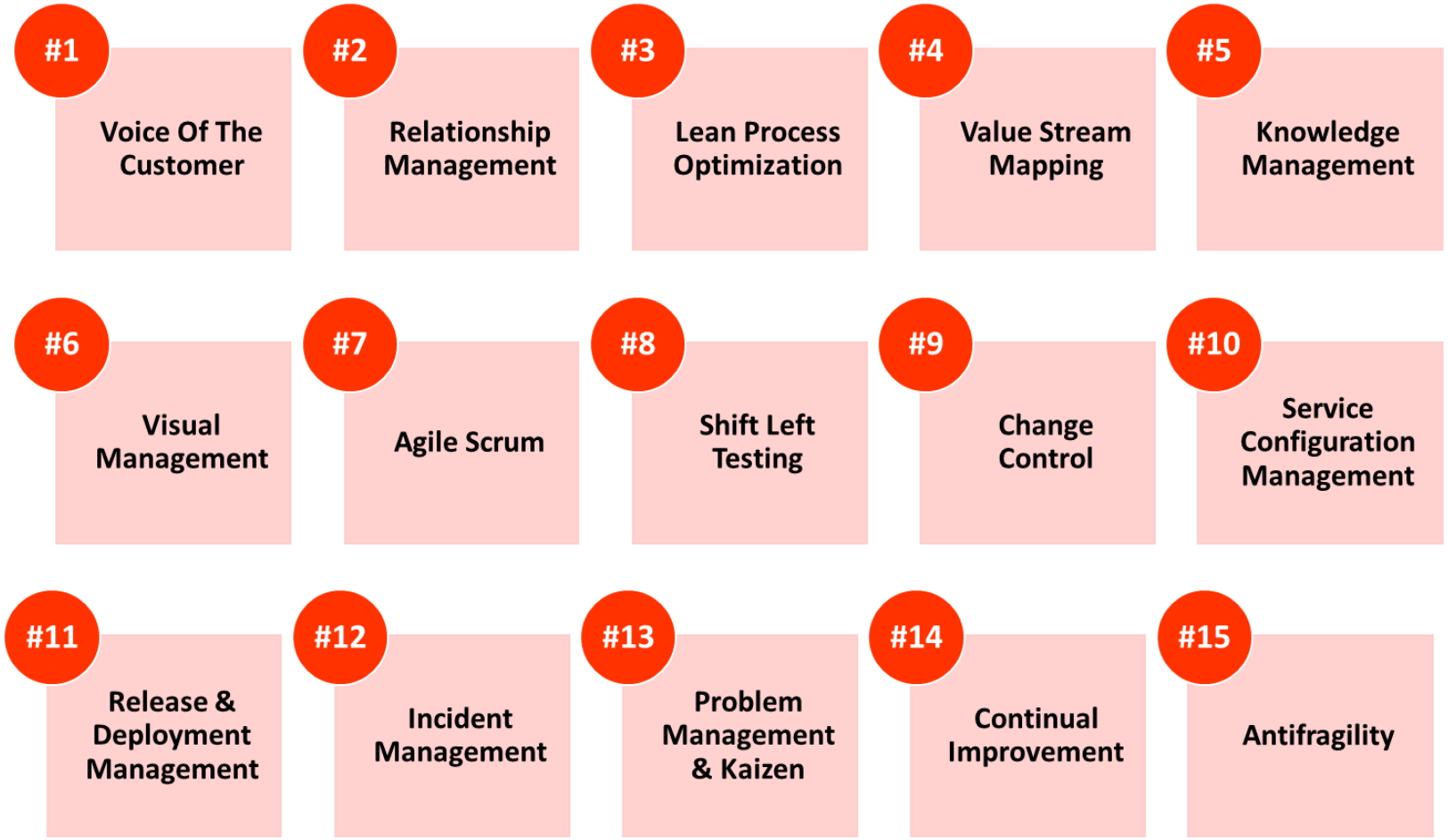
In 2016, they set loose Chaos Kong, a simulated DDoS strike that put Netflix's antifragility to the ultimate test.

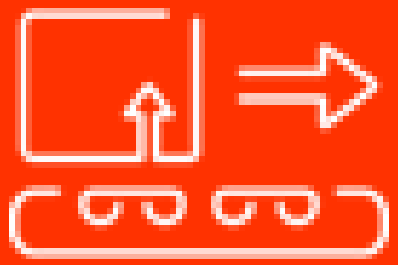
**Chaos Engineering** is an entire discipline that is all about achieving antifragility and ensuring stable performance even in the face of emergencies or unplanned incidents and outages.

> *"Chaos Engineering is the discipline of experimenting on a distributed system in order to build confidence in the system's capability to withstand turbulent conditions in production."*
>
> **The Principles Of Chaos Engineering**

Source: http://blog.cognitect.com/blog/2016/3/24/the-new-normal-embracing-failure-with-netflix-the-chaos-monkey-and-chaos-kong

# The 15 Essential Practices Of DevOps



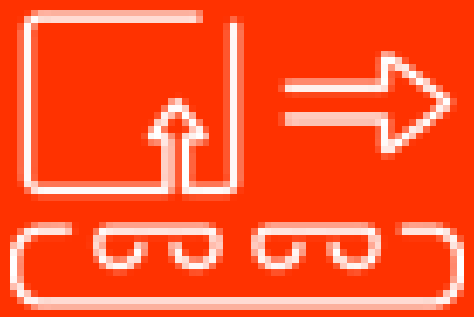


PeopleCert DevOps Fundamentals

# Module 5:
# The Full Stack – Technology & Automation

**Topics:**

- Automation For The Deployment Pipeline
- Cloud Technology & Virtualization
- Architecting For Continuous Delivery

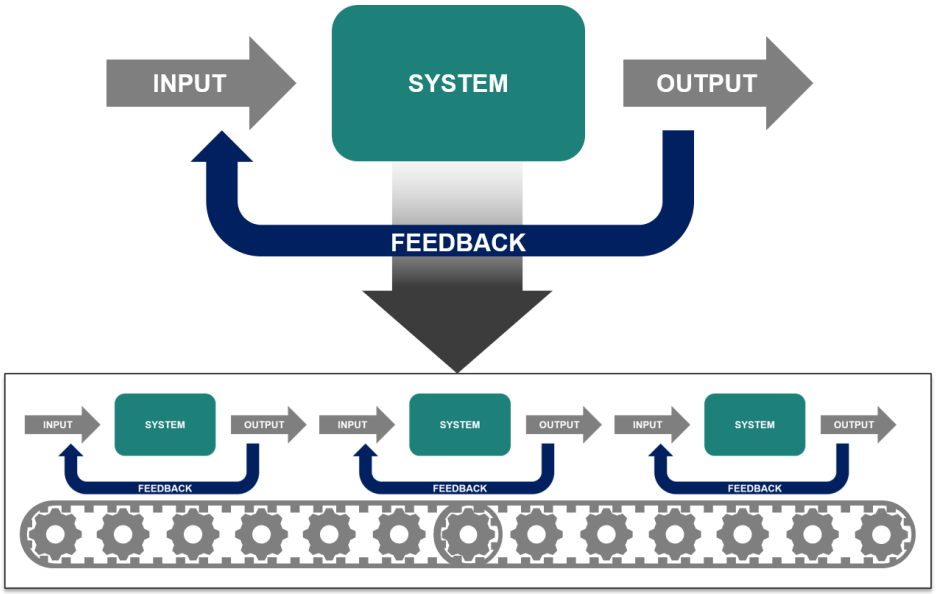


# Module 5: The Full Stack – Technology & Automation

## Automation For The Deployment Pipeline

# Understanding Automation

Automation is one of the central concepts of DevOps. In fact, a common misconception about DevOps is that it *is* automation. This perception comes from the fact that automation is crucial to the Three Ways. It is one of the best ways to create flow, ensure that feedback occurs, and allow for experimentation and learning.



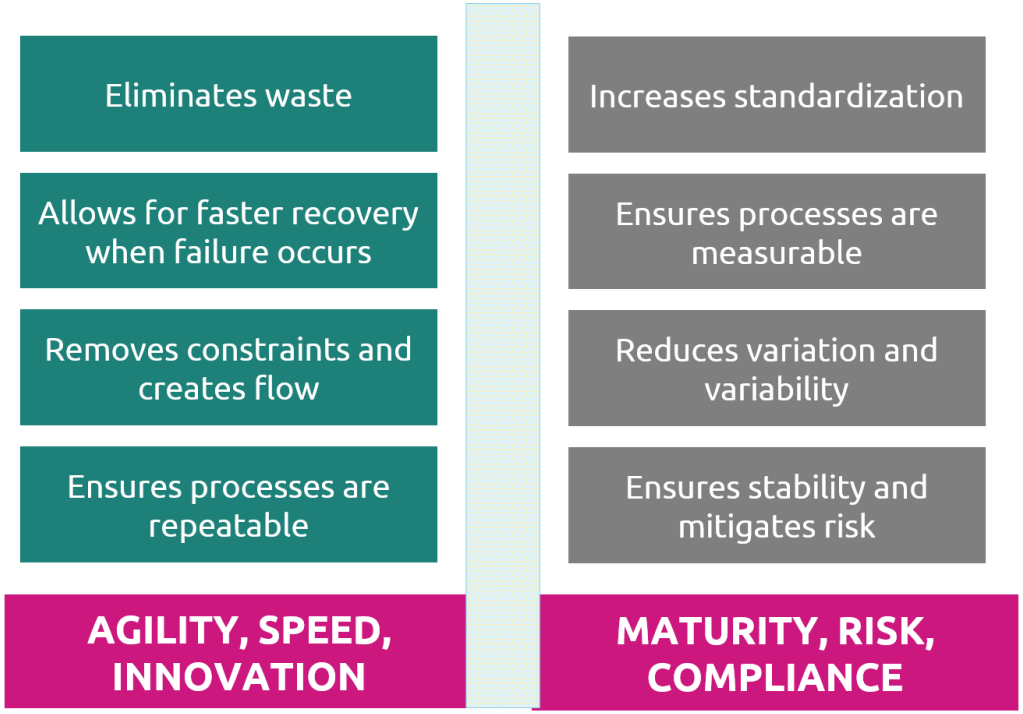

Automation is removing constraints and increasing standardization to create flow (i.e. the input to output and the system in between from left to right) and ensuring that consistent feedback loops are also in place that support flow.

The reason that automation creates standardization and removes constraints is because in order to automate, the work must first be done to ensure that the processes are automated the right way.

Automating bad processes will result in bad results. This is why culture and practices stacks are addressed first before adding the technology stack.

Source: *Service Automation Framework*, Jan-Willem Middelburg

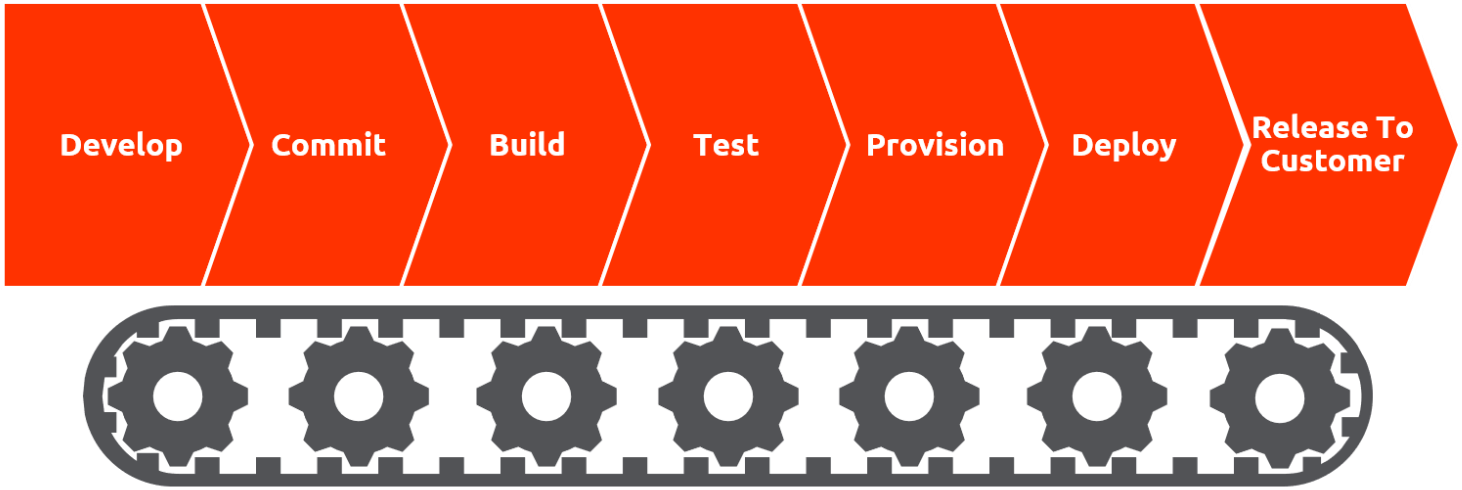
# Automation Means Flexibility & Stability




- Automation eliminates waste by removing human labor from those processes where it is not needed by minimizing handoffs and rework. It allows for faster recovery when failure occurs by standardizing and reducing variability in processes, making it easier to identify and correct errors and avoid passing them downstream. It removes constraints and creates flow, speeding up processes. And it ensures that processes are repeatable, again eliminating rework. All of these benefits help IT to go faster.

- Automation increases standardization and reduces variability in addition to repeatability. It also ensures that processes are measurable, which means that reports can be pulled more quickly and more data can be gathered to further improve processes, ensure stability and mitigate risk.

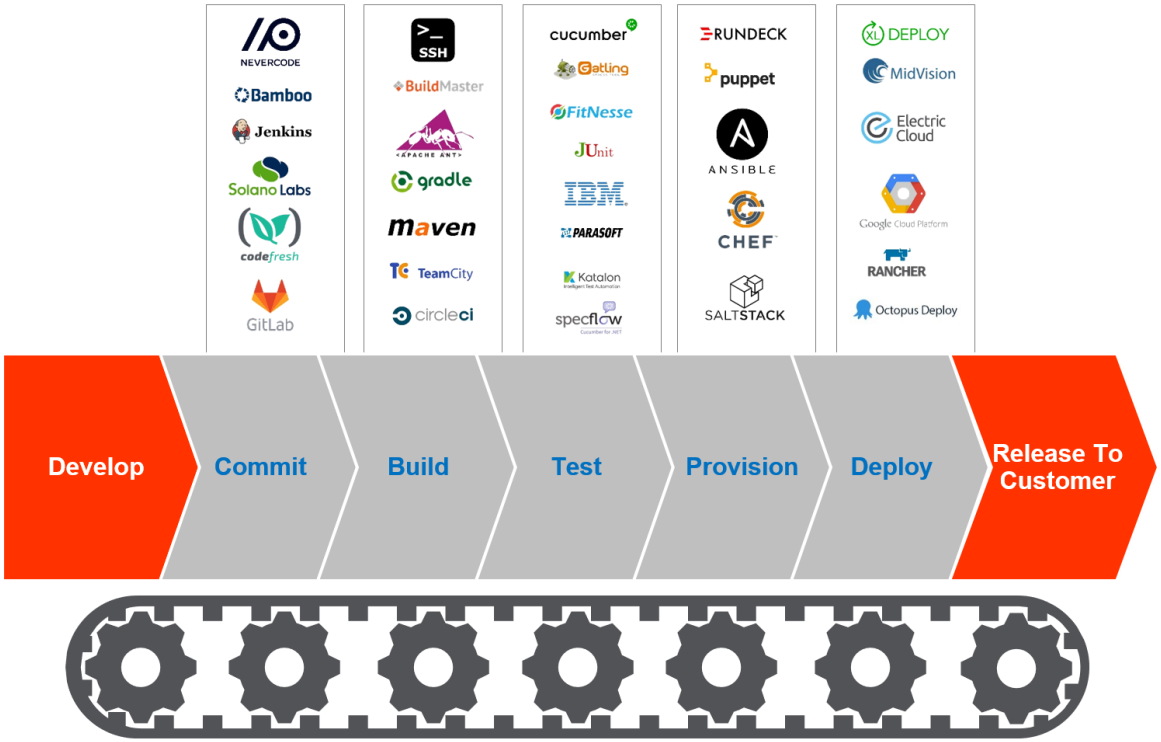
# Deployment Pipeline Automation




| | | | |
|---|---|---|---|
| Ensure the quality of code | Ensure the state of environments | Ensure that software is deployed correctly | Ensure stability and performance |

- Each stage of the pipeline provides a new opportunity to verify the quality of features and provide feedback. It provides transparency and visibility related to the flow of changes and work.

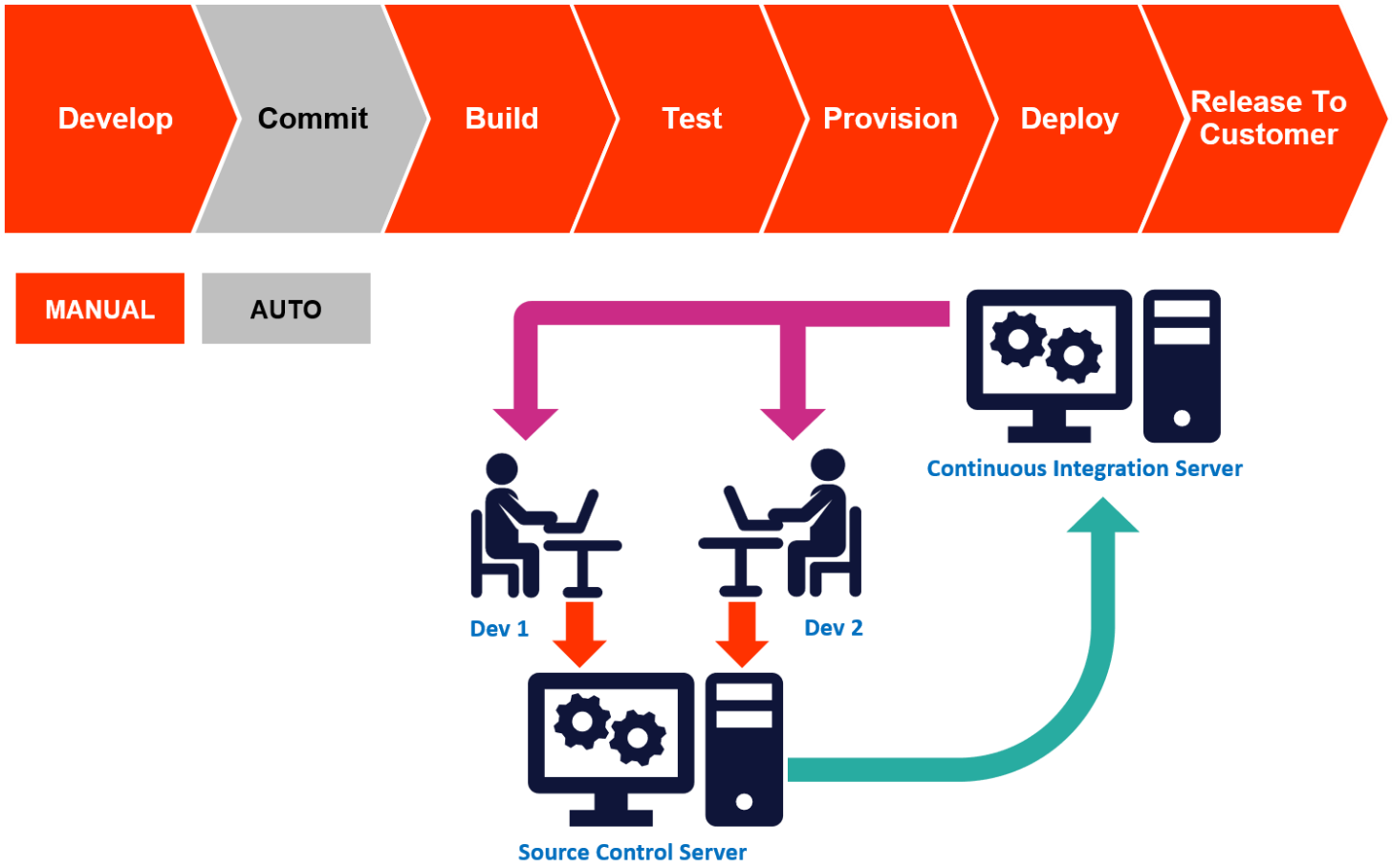
# Tools For A DevOps Toolchain



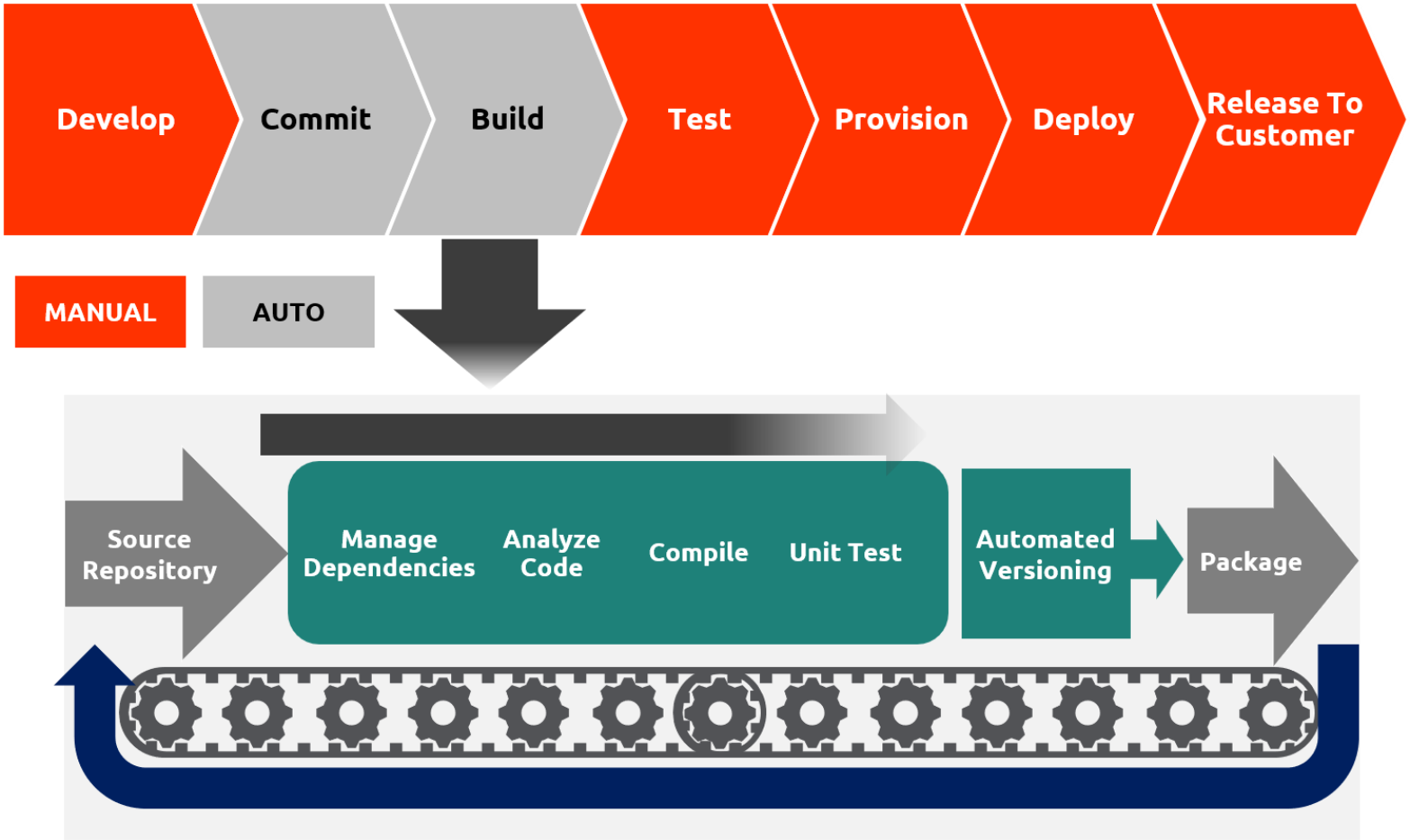

## Other Tools That Support DevOps

- Database Management
- Monitoring & Dashboards
- Release Management
- Workflow & Ticketing
- Service Configuration
- Collaboration
- Security
- Networking
- Change Control
- Orchestration

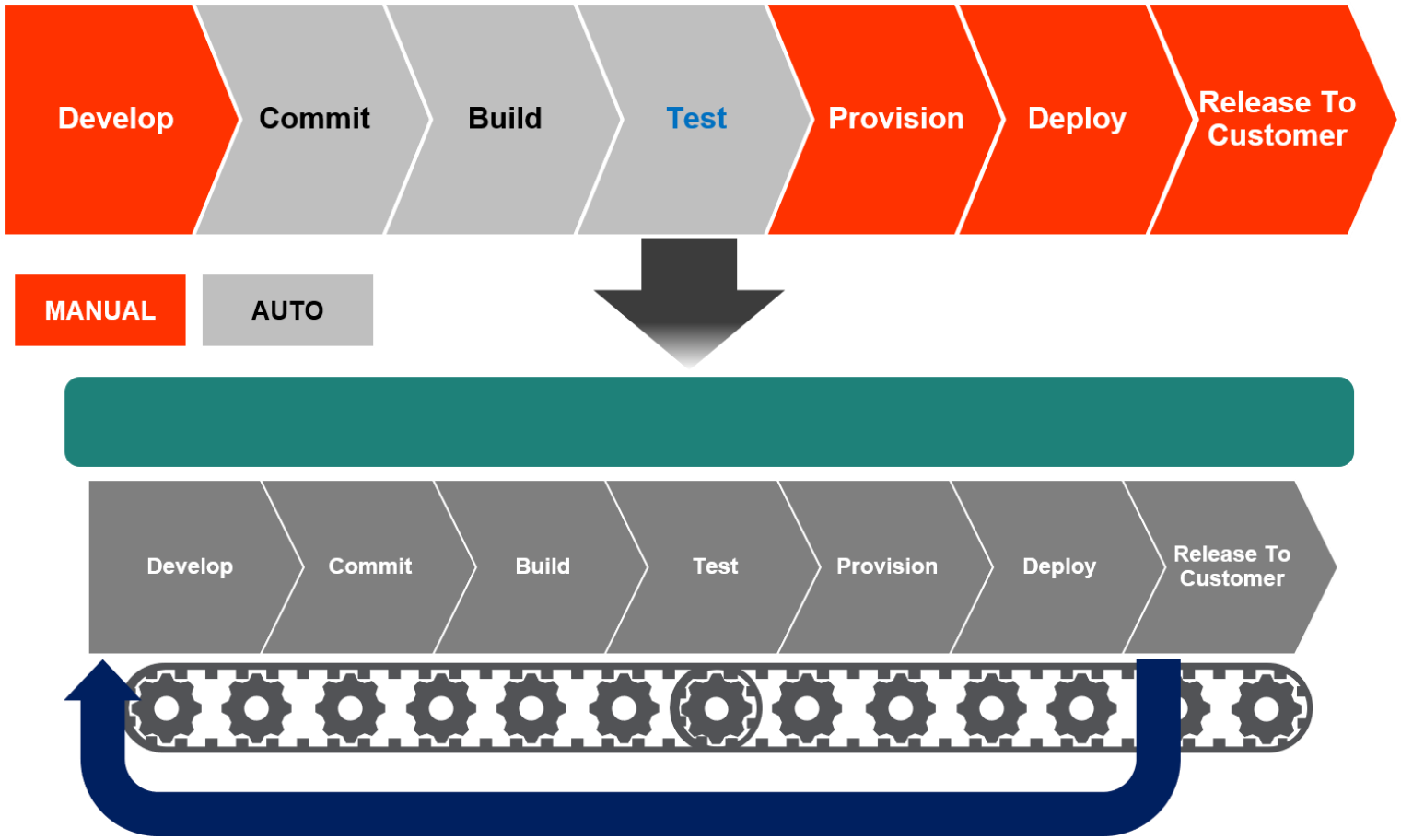
# Continuous Integration



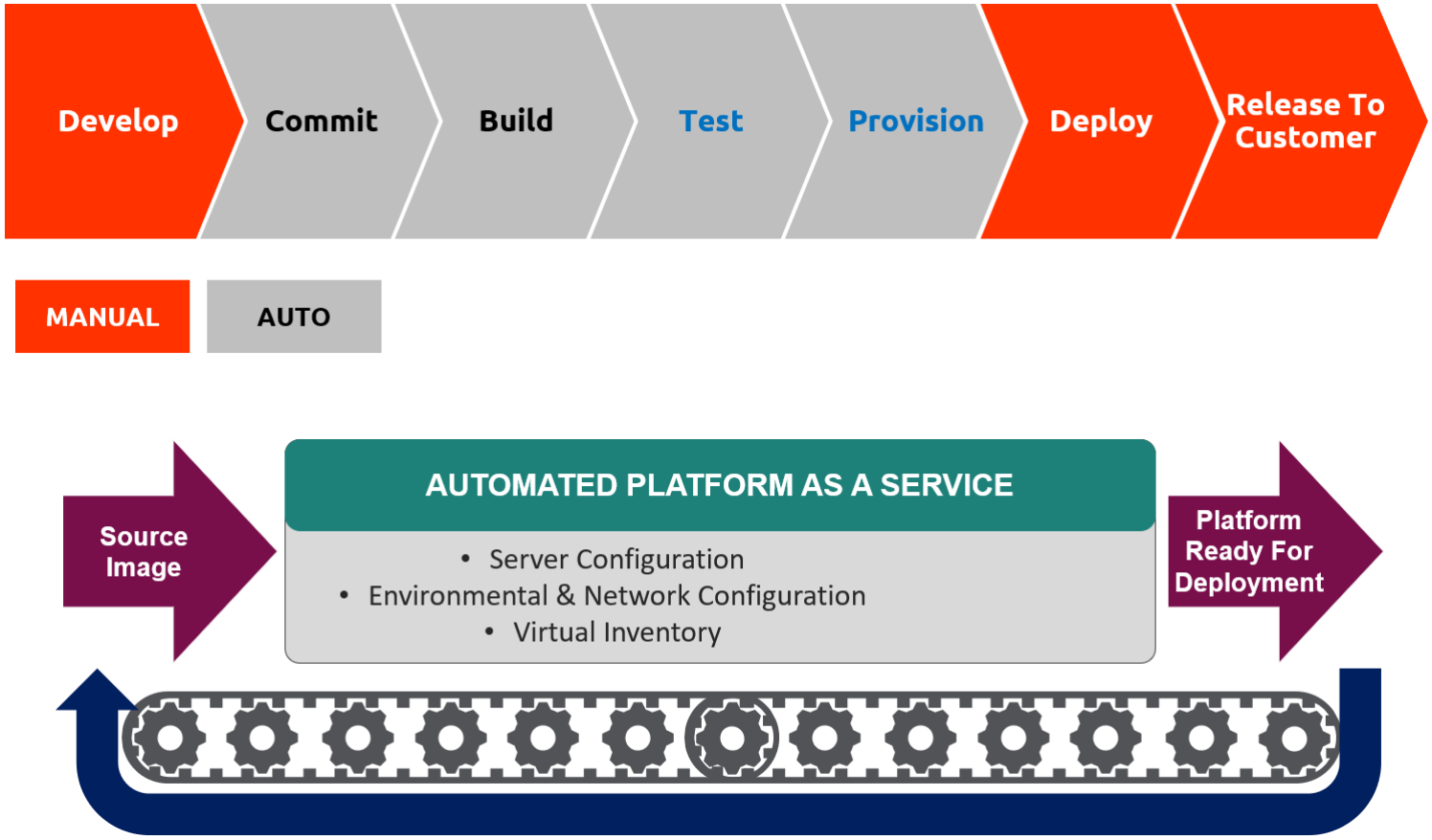
# Automated Build



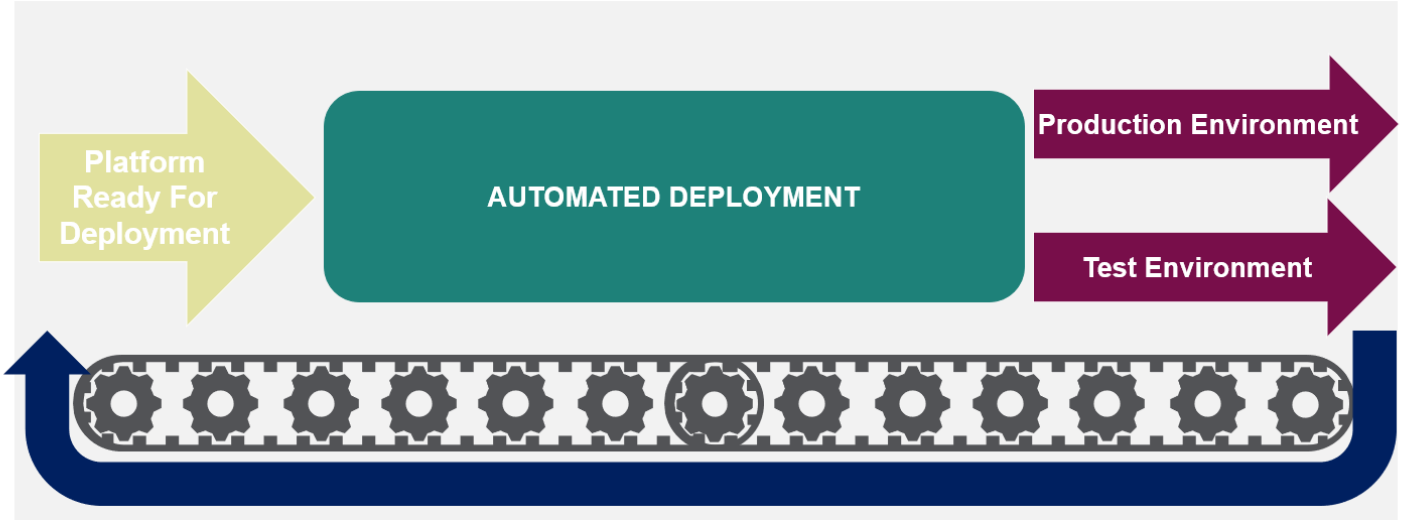
# Automated End To End Testing



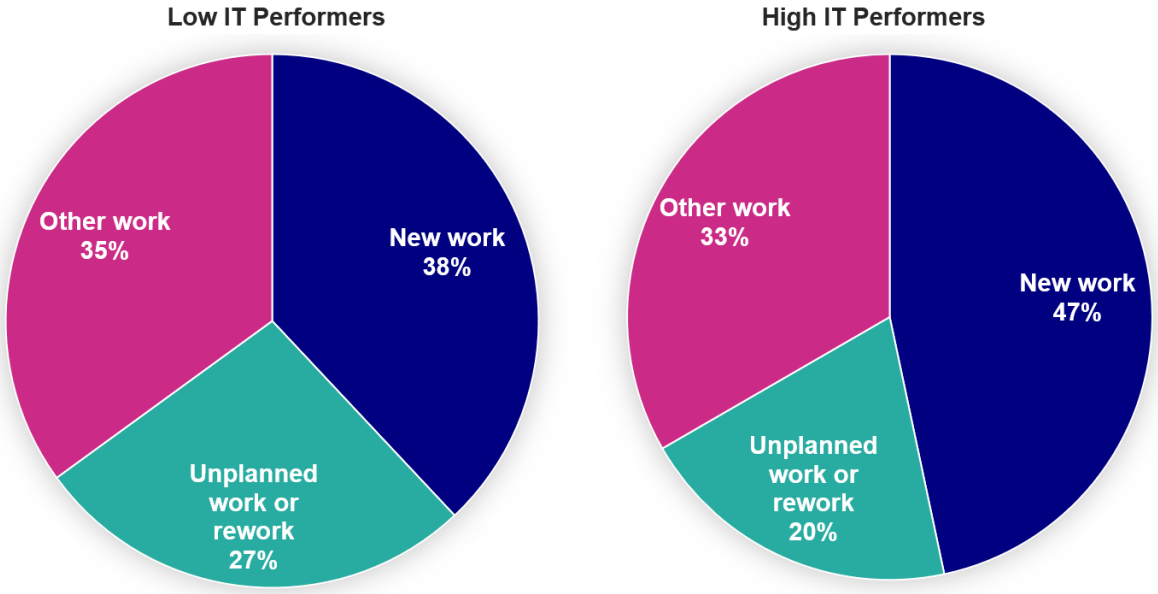
# Automated Provisioning



# Automated Deployment

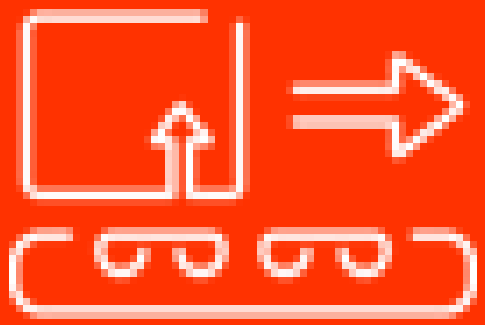
# Automation Shifts Focus




- The *2016 State of DevOps Report* found that low performers were spending 27% of their time on unplanned work or rework, and 35% of their time on other work such as meetings and routine maintenance. These buckets are generally made up of tasks that are routine, manual, and repetitive. Often, this work occurs and is slow due to bottlenecks or manual hand-offs.

- High performers, however, were able to spend 11% more of their time on new work, in part enabled by the cloud-enabled automation that is critical to DevOps. They were able to spend more of their time focused on tasks that could not be automated or eliminated – tasks that are non-routine, cognitive in nature, that require human creativity, skill, experience, judgment or problem solving. They were able to spend more time innovating.

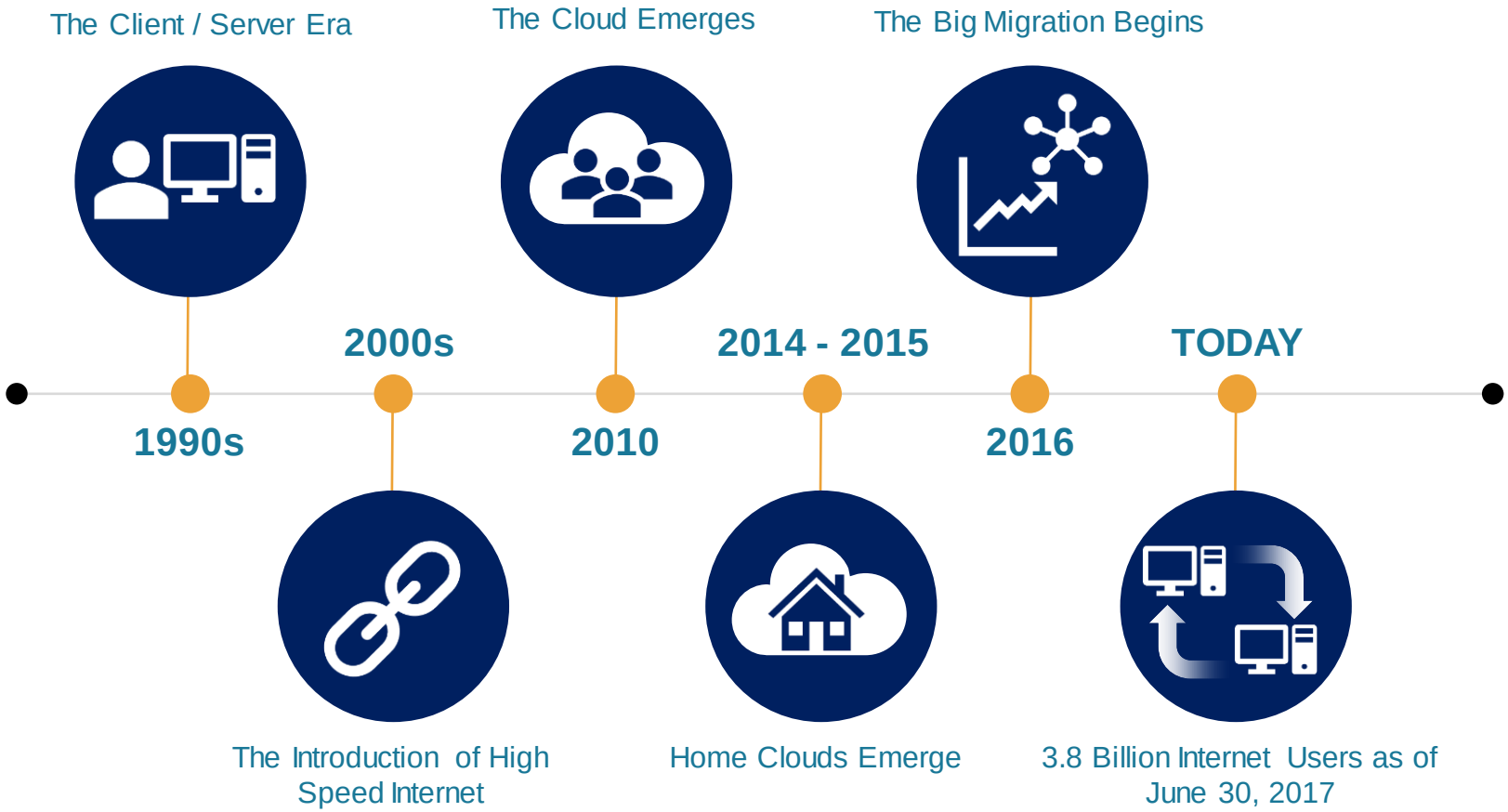
Source: *2016 State Of DevOps Report*, Puppet Labs



# Module 5: The Full Stack – Technology & Automation

Cloud Technology & Virtualization

# The Emergence Of Cloud Technology



# Characteristics Of Cloud Computing


With a cloud service, users can provision resources and capabilities like server time and network storage automatically, without human interaction.


Resources are accessible via a network and through standard mechanisms, including a wide variety of mobile devices and workspaces that might be required by users.

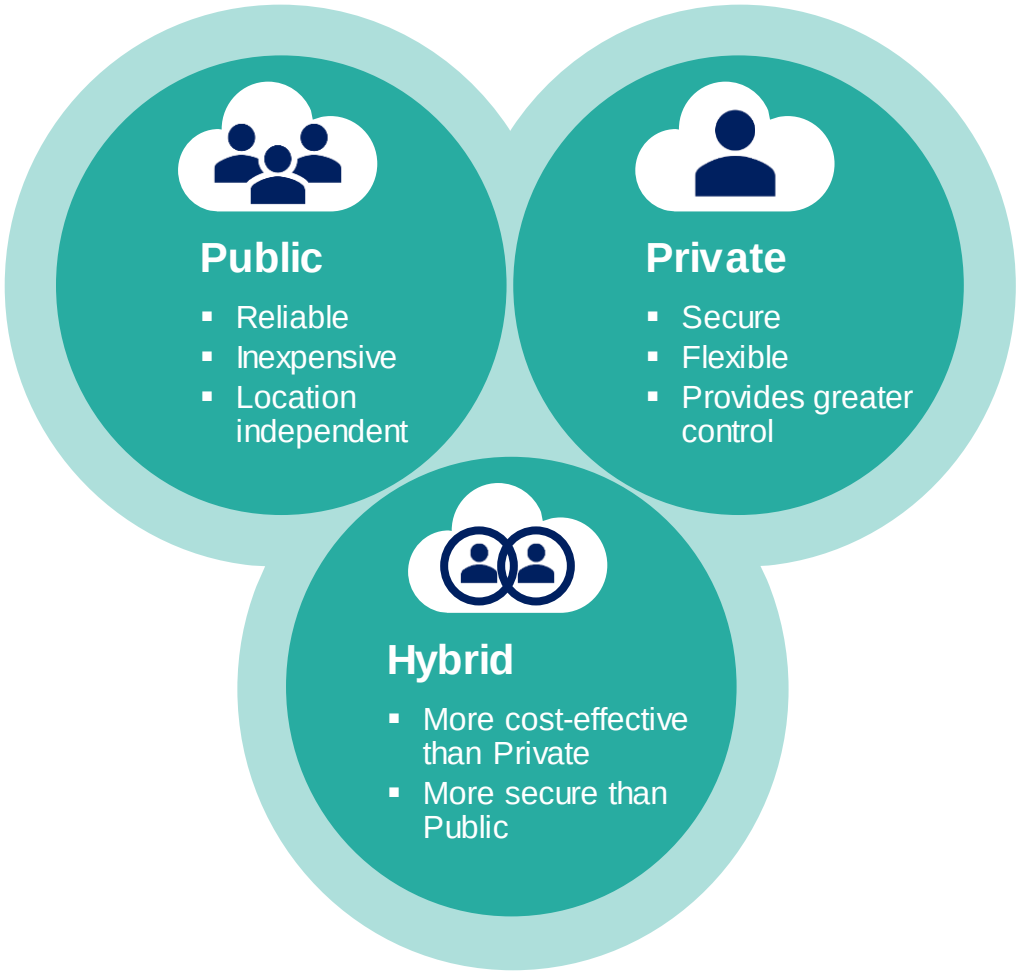

A cloud service can be available to multiple customers using the same physical resources – known as a multi-tenant model, which is achieved by assigning and re-assigning resources dynamically according to user demand.


Resources are provisioned and released with the elasticity to quickly scale outward and inward based upon user demand.


Resource usage is controlled and optimized by leveraging an appropriate metering capability. Usage is then monitored, measured and billed according to utilization – it is pay for use.

Source: https://www.ibm.com/blogs/cloud-computing/2014/01/cloud-computing-defined-characteristics-service-levels/

# Cloud Computing Deployment Models




Public Cloud Computing Providers

Microsoft Azure

rackspace

Google Cloud Platform

hp Cloud Services

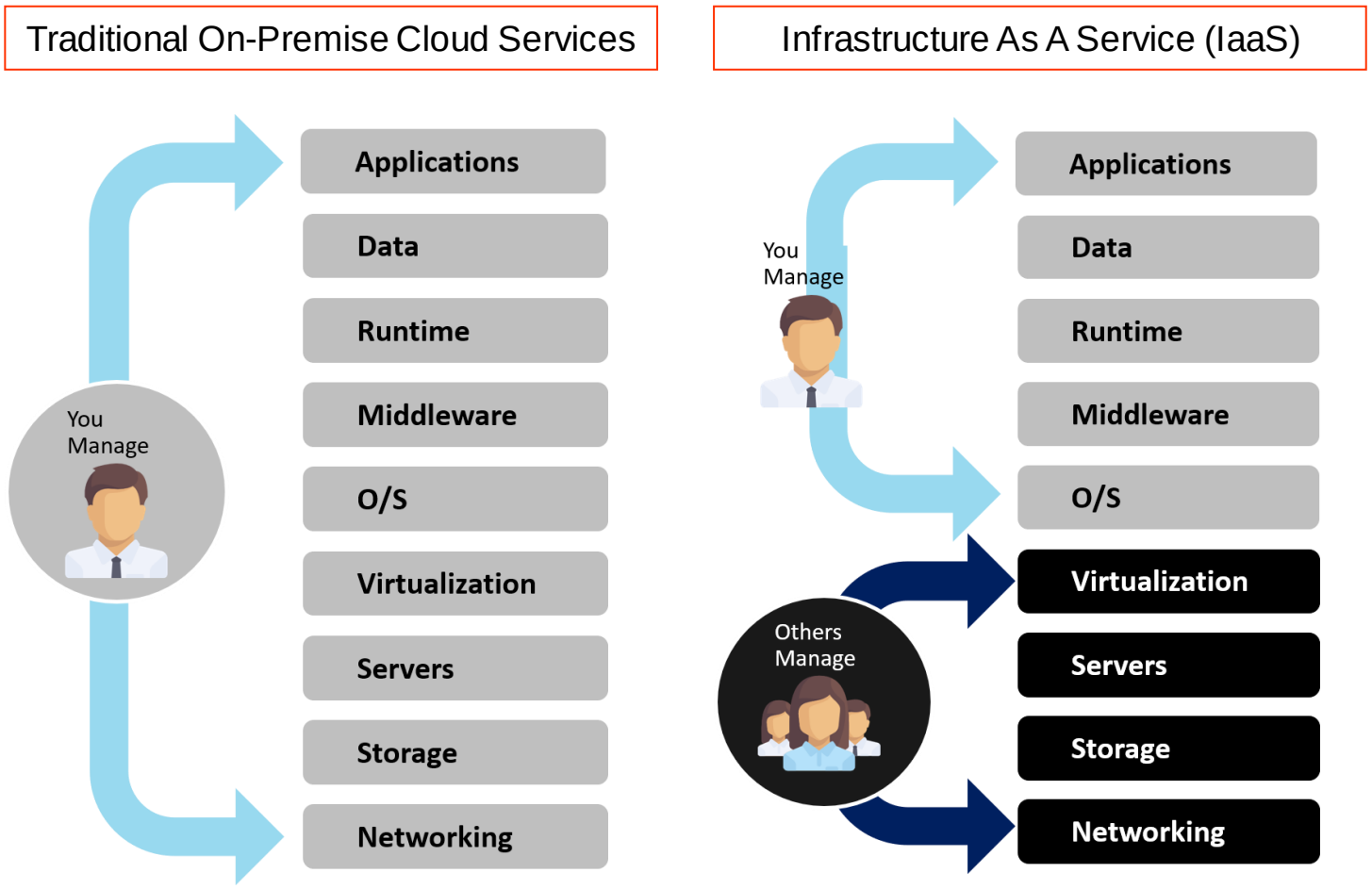
amazon web services

IBM Cloud

DELL

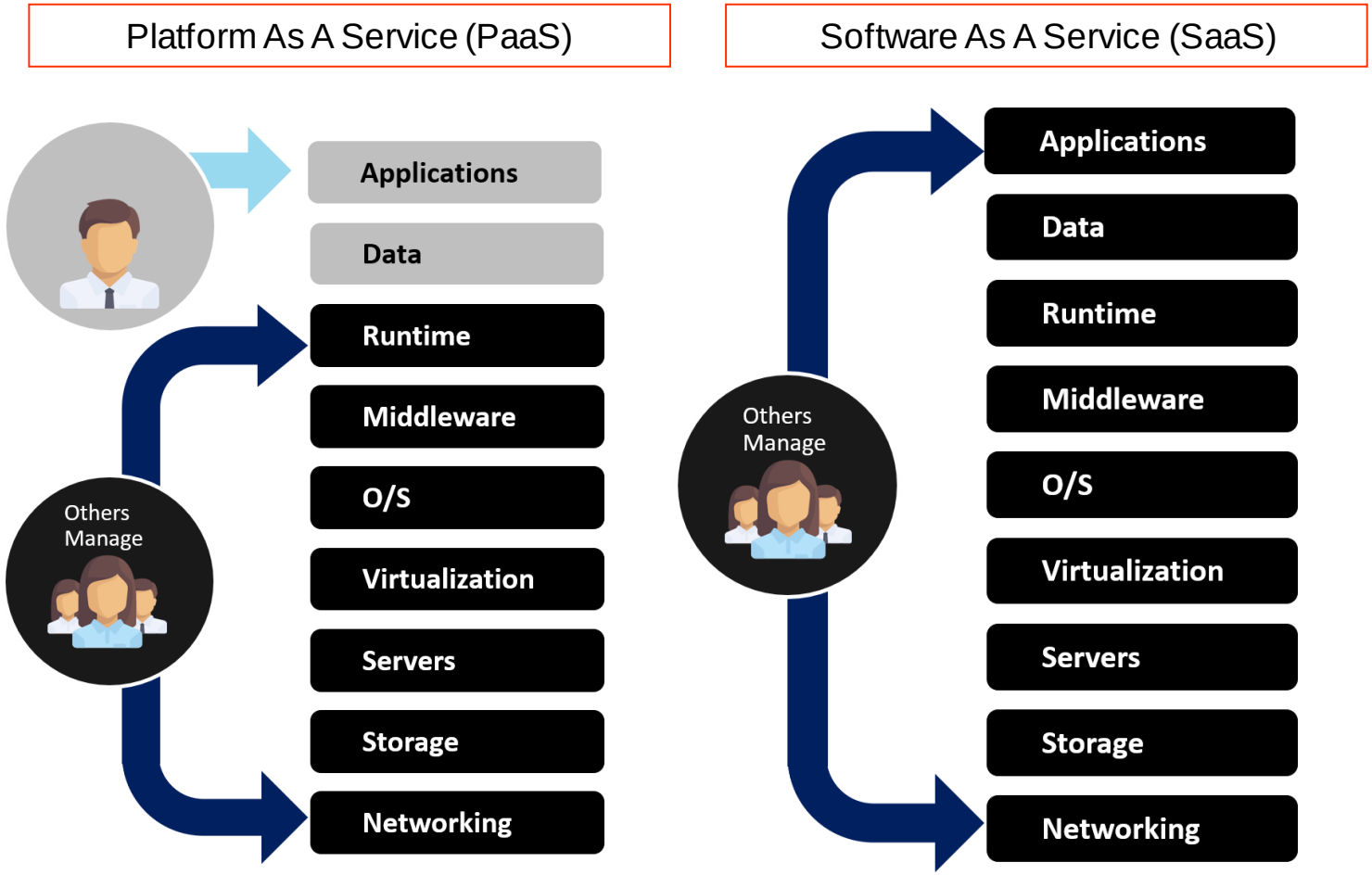
Source: https://www.ibm.com/blogs/cloud-computing/2014/01/cloud-computing-defined-characteristics-service-levels/

# Cloud Computing Service Models



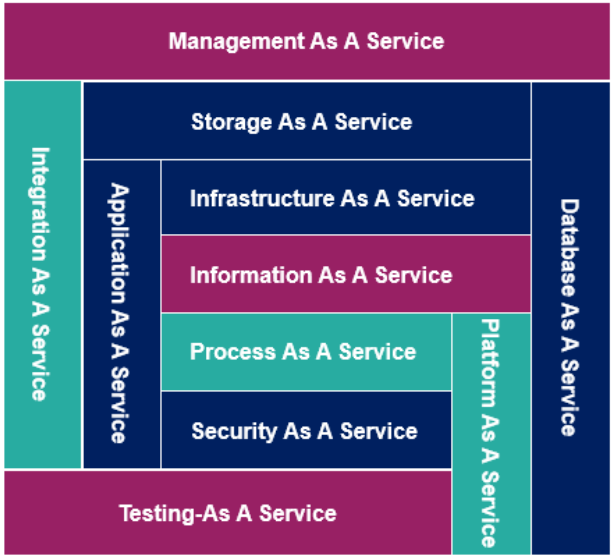

Source: https://www.ibm.com/blogs/cloud-computing/2014/01/cloud-computing-defined-characteristics-service-levels/

# Cloud Computing Service Models




Source: https://www.ibm.com/blogs/cloud-computing/2014/01/cloud-computing-defined-characteristics-service-levels/

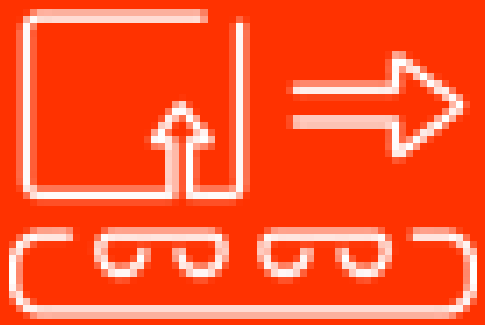
# Everything As A Service & Everything As Code





The four Cloud Computing Service Models are not the ONLY service models that exist. There are many examples of those four models being tweaked to outsource or manage other specific components of the cloud. This is known as "Everything As A Service". There are a wide variety of options that exist that can support or compliment DevOps and that will continue to evolve moving forward.

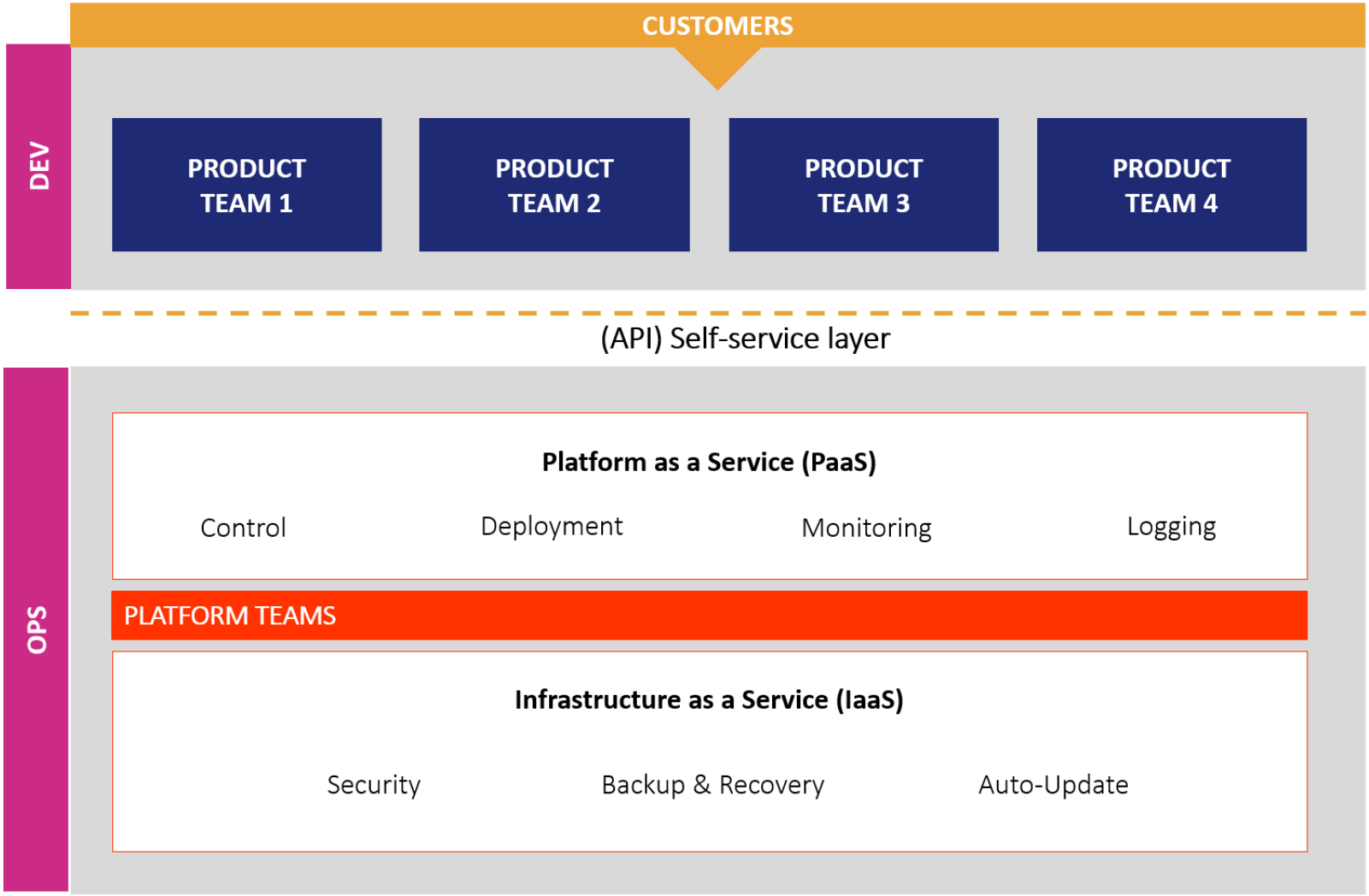
Increasingly, software is defining everything. Everything is becoming code, becoming virtualized, becoming abstract as opposed to physical. This allows for increased flexibility, increased connectivity, and increased speed of service. Nothing has to be physically stored with cloud and with software defining everything. It opens up a wealth of new possibilities for IT and enables the balance of speed and reliability that DevOps represents.



# Module 5: The Full Stack – Technology & Automation

Architecting For Continuous Delivery

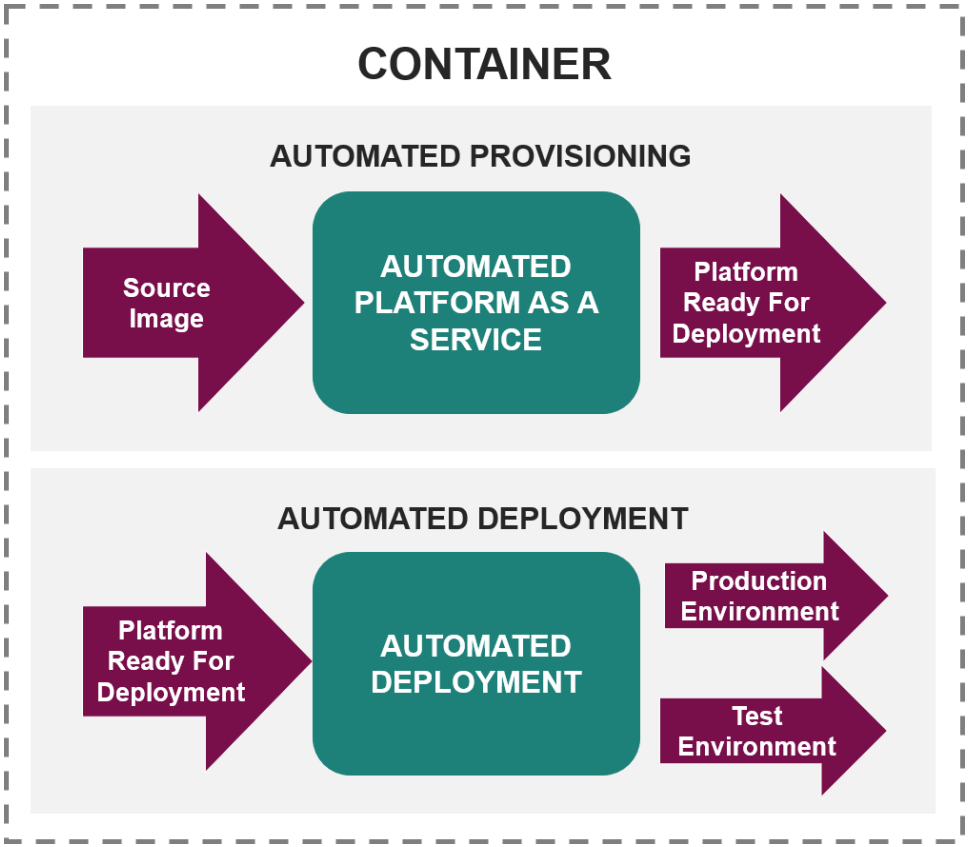
# Product & Platform Teams



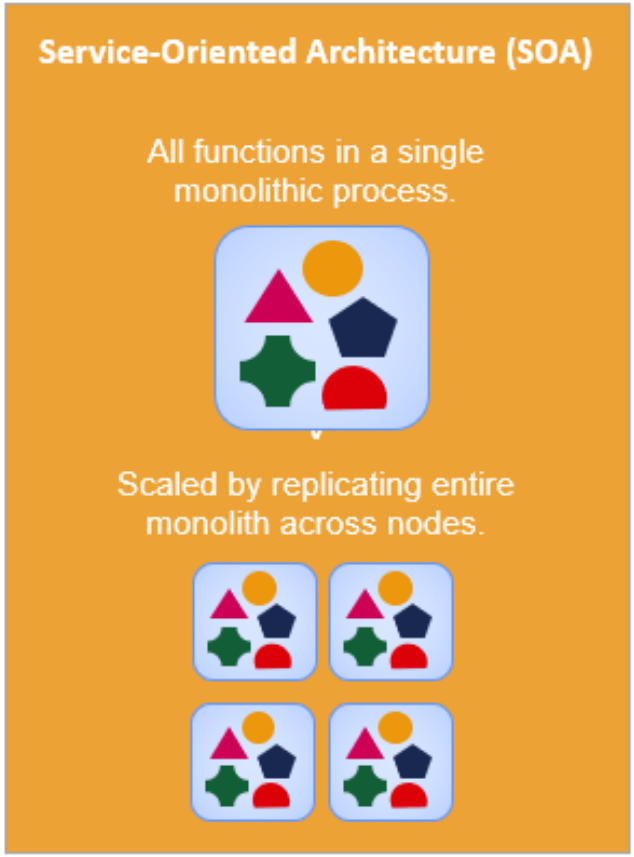
# Containerization

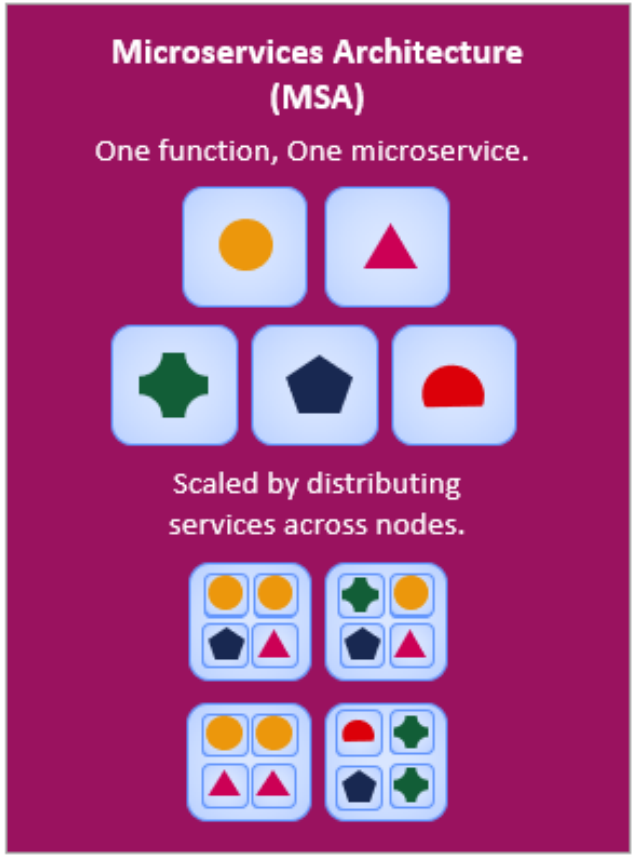
- Containers are a solution to the problem of how to get software to run reliably when moved from one computing environment to another. This could be from a developer's laptop to a test environment, from a staging environment into production and perhaps from a physical machine in a data center to a virtual machine in a private or public cloud.

- Put simply, a container consists of an entire runtime environment: an application, plus all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package. By containerizing the application platform and its dependencies, differences in OS distributions and underlying infrastructure are abstracted away.



# Microservice Architectures

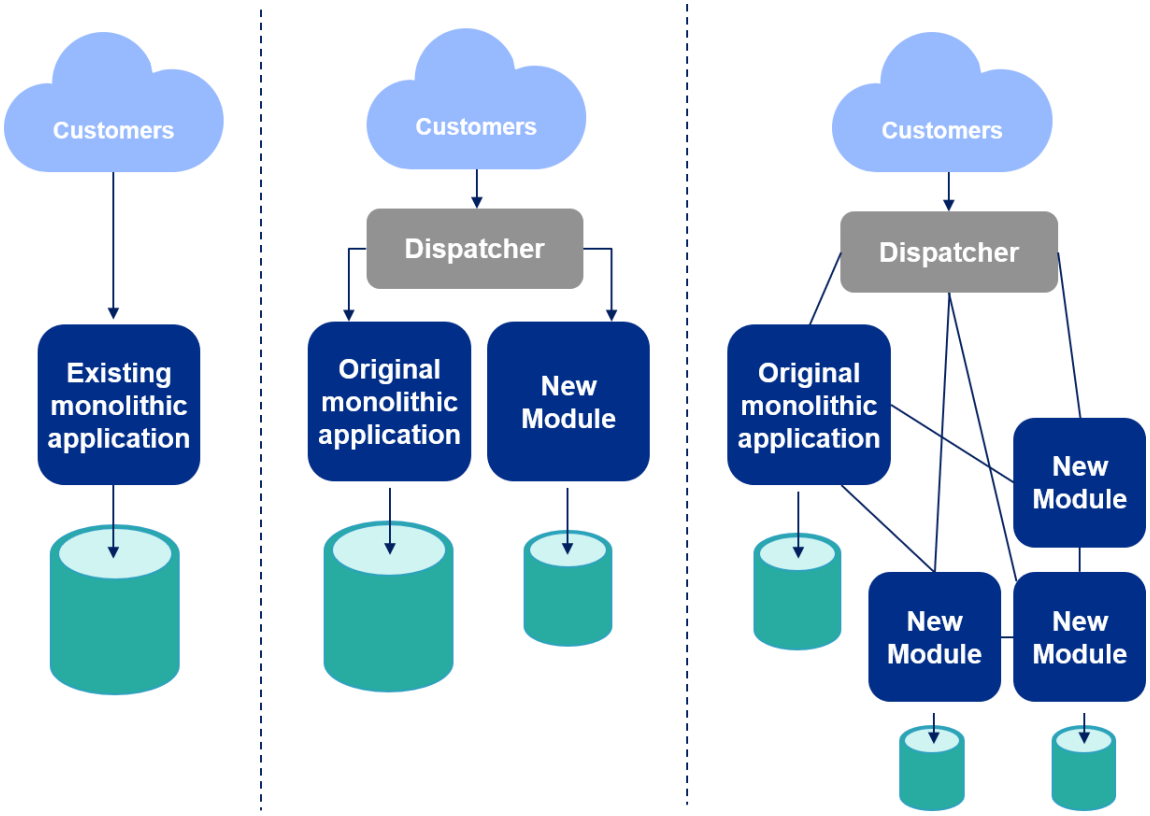



**For more information, check out this video:**
*What Are Microservices?:* https://www.youtube.com/watch?v=CKL3fV5UR8w

Source: https://martinfowler.com/articles/microservices.html

# From Legacy To Smaller Services

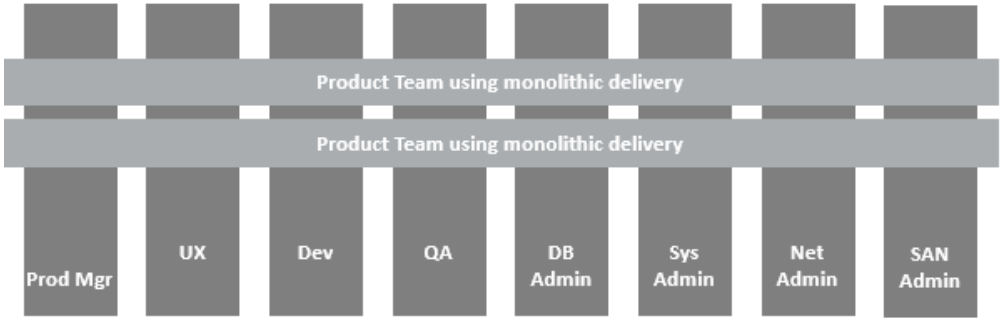


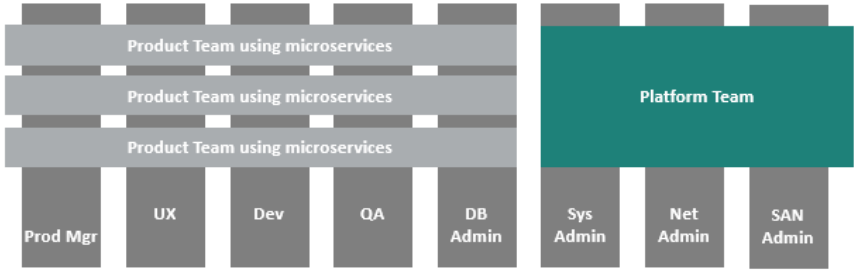

A **Strangler Application** is a monolithic application which is "strangled" over time by the iterative introduction of microservices to seamlessly replace its specific features and functions.

Source: http://continuousdelivery.com/implementing/architecture

# Microservices & Teaming

Microservices enable the evolved teaming into Product and Platform Teams. Traditionally, organizations have operated in silos around specialists, which requires many handovers, tickets, meetings, etc. to deliver a complete product. By this point, it should be clear that silos are undesirable. Product Teams built around monolithic applications help speed up delivery, but ultimately they are still organized around the silos.



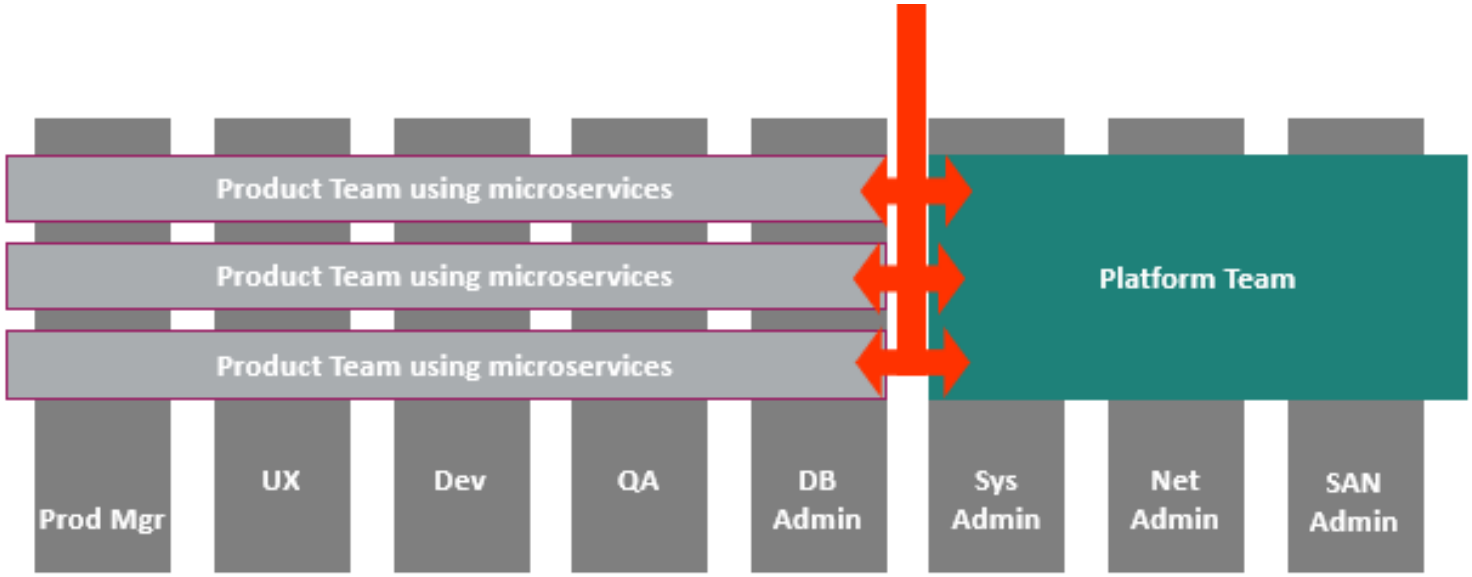




DevOps reorganizes these silos using microservices. There is a Platform Team, the team responsible for creating and maintaining a platform everything runs on. And this moves management from their silos and puts them in charge of Product Teams, teams that own particular products rather than specialized tasks. These teams own products throughout their entire lifecycle.

Source: *Cloud Trends, DevOps, and Microservices*, Adrian Cockcroft

# Application Programming Interfaces (APIs)

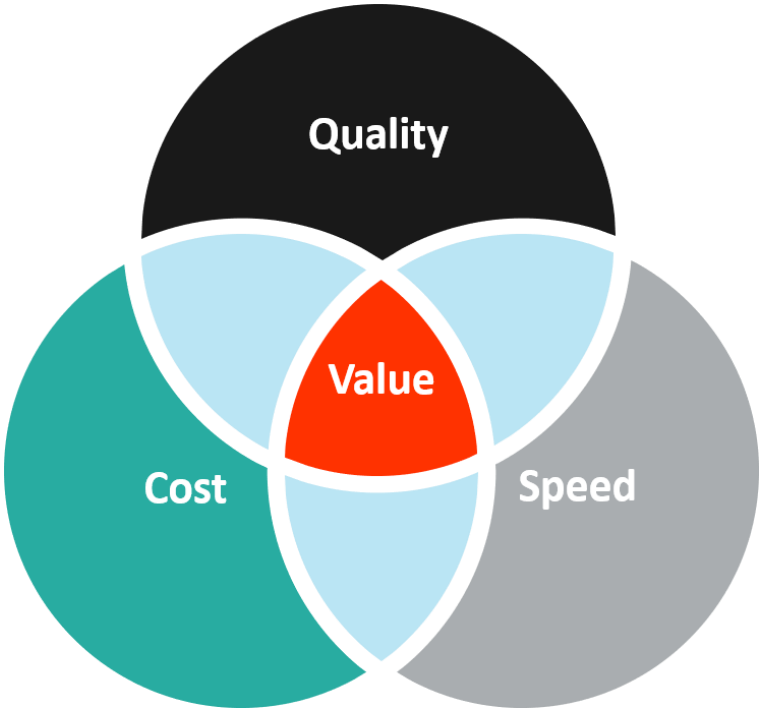
**Application Programming Interfaces** provide a set of clearly defined methods of communication between the Product and Platform teams. They simplify the relationship between the teams by standardizing and automating many of the routine functions that the teams interact to complete.



# PeopleCert DevOps Fundamentals

## Wrap Up & Review

# DevOps Delivers Value




**Improved Quality:**

- Architecture that is both stable and flexible enough for change
- Increased service availability and reduced errors
- Integrated data enabling collaboration and feedback

**Increased Speed:**

- A shared set of priorities and a realistic plan to achieve them
- A reduced backlog of work and a sustainable pace
- Faster flow enabled by optimized and integrated IT management tools

**Reduced Costs:**

- Reduction in the costs of keeping things running through standardization and the elimination of non-value added work
- Elimination of technical debt and redundant tools enabling automation and increased efficiency
- Increased cross-training and emphasis on generalized specialization enabling less dependency on expensive external resources

# Further References

## Subject Matter Experts

- Gene Kim (@realgenekim, realgenekim.me)
- James Turnbull (@kartar, jamesturnbull.net)
- John Willis (@botchagalupe, blog.docker.com/author/johnwillis)
- Nicole Forsgren (@nicolefv, nicolefv.com)
- Patrick Debois (@patrickdebois, www.jedi.be)
- Alan Shimel (@ashimmy, stillsecureafteralltheseyears.com)

## Online Communities & Blogs

- IT Revolution (www.itrevolution.com)
- DevOps.com (devops.com)
- DevOpsGuys (www.devopsguys.com/blog)
- Code As Craft (codeascraft.com)
- The Agile Admin (theagileadmin.com)
- XebiaLabs Blog (blog.xebialabs.com)
- Dev2Ops (dev2ops.org)

## Videos, Webinars & Other Media

- DevOps Café Podcast (devopscafe.org)

# Further References

## Case Studies & Research

- The annual State of DevOps Report (puppet.com/resources/whitepaper/state-of-devops-report)
- IBM DevOps Case Studies (www.ibm.com/ibm/devops/us/en/casestudies)

## Publications

- *Lean Enterprise* by Jez Humble, Joanne Colesky & Barry O'Reilly
- *The DevOps Handbook* by Gene Kim, Jez Humble, Patrick Debois & John Willis
- *The Phoenix Project* by Gene Kim, Kevin Behr & George Spafford
- *Continuous Delivery* by Jez Humble & David Farley

## Conferences

- DevOpsDays (www.devopsdays.org)
- DevOps Enterprise Summit (events.itrevolution.com)
- DockerCon (2017.dockercon.com)
- Velocity Conference (conferences.oreilly.com/velocity)
- PuppetConf (puppet.com/puppetconf)
- Pink18 (www.pinkelephant.com/en-US/Pink18/home)

PeopleCert DevOps.

# Fundamentals Exam Information

# Exam Tips

**Before the exam:**

- Review the material
- Ask questions in the Q&A that follows
- Use the sample paper to familiarize yourself with the question types that will be on the exam and to test your score
- Review all answers you got wrong and look through the material for more information on the answer
- Don't engage in negative talk with other students before the test, eliminate negative thoughts and have a positive attitude towards your test taking at hand

**During the exam:**

- RELAX!
- Read the test instructions carefully
- Read the question stem, thing of the answer and then check the answers to find the BEST one that suits your answer
- Read all of the options before choosing
- Read through test questions at least once
- Do not spend too much time in questions that you don't know; answer the ones you are sure of the answer first and then get back to the ones you are not sure of what the answer is
- ANSWER ALL questions, even if you have to choose an answer at random

# Exam Structure

| Delivery | Computer (web proctored or classroom) |
|---|---|
| Type | **40** Multiple Choice Questions (MCQ)<br>Each question is awarded **one (1) mark** |
| Duration | **1 hour (60 minutes)**<br>For non-native speakers or candidates with a disability, an additional 30 minutes of extra time is allowed. |
| Pass Mark | **70% (28 marks out of 40)** |
| Invigilator / Supervisor / Proctor | Yes<br>Physical or Online Proctoring |
| Open Book | No<br>No materials are allowed in the examination room |

# Questions

PeopleCert Values Your Feedback

**Like the course?**
**Have something to say?**

Watch for a course evaluation coming to your inbox soon!

# Credits

- The History Of DevOps:
  https://www.youtube.com/watch?v=o7-IuYS0iSE
- What Is DevOps?:
  https://www.youtube.com/watch?v=bYQhlpa9eoc
- Gene Kim Defines The 3 Ways Of The Phoenix Project:
  https://www.youtube.com/watch?v=nUOXDEvplRc&x=mobref
- Continuous Delivery is a Team Sport:
  https://www.youtube.com/watch?v=9XmvFvdTObY
- Drive: The Surprising Truth About What Motivates Us:
  https://www.youtube.com/watch?v=u6XAPnuFjJc
- Spotify Engineering Culture: Part 1:
  https://www.youtube.com/watch?v=4GK1NDTWbkY
- What Are Microservices?:
  https://www.youtube.com/watch?v=CKL3fV5UR8w
- https://devops.com
- https://leankit.com/
- https://trello.com
- https://www.freepik.com/free-photos-vectors/infographic
  Infographic vector created by vectorpocket - www.freepik.com
- https://www.istockphoto.com/
- https://www.flaticon.com/
- https://www.gettyimages.com/

# Thank You!

For latest news and updates follow us.

**linkedin.com/company/peoplecert-group/**
**twitter.com/peoplecert**
**youtube.com/channel/UCLBidKZS9Xk08f_PM5Edtfg**
**facebook.com/peoplecert.org**