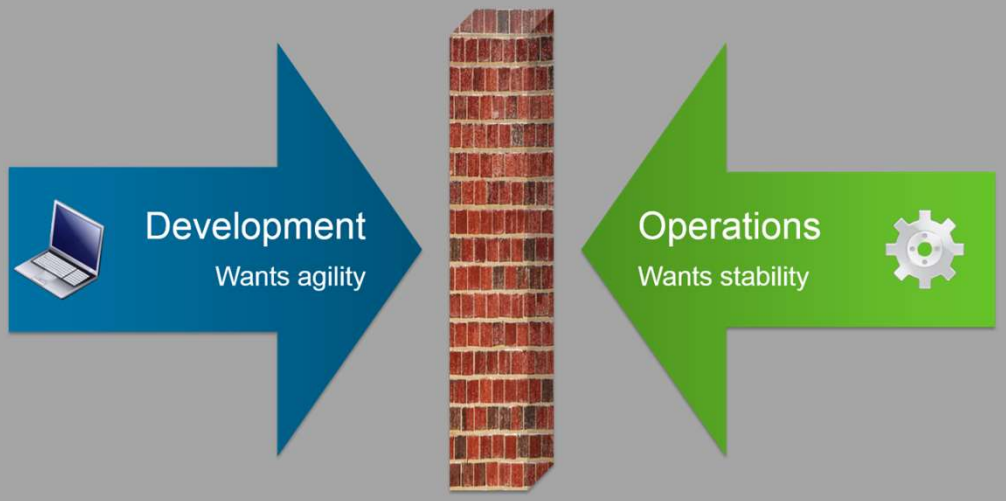

**S.M SOHAIB UR REHMAN**
Principal DevOps Manager, TPS Worldwide
SVP, Karachi Chapter at Agile Pakistan
Visiting Faculty: Karachi University UBIT

# Agenda

- Concept
- Challenges
- Solutions
- Opportunities

# Introduction

What and Why?



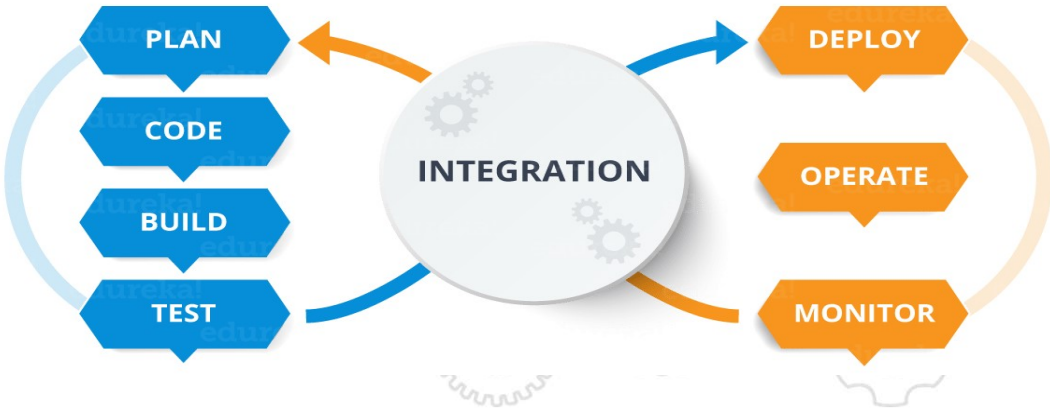

*"Devs are from Venus*
*Ops are from Mars"*

Steven Haines

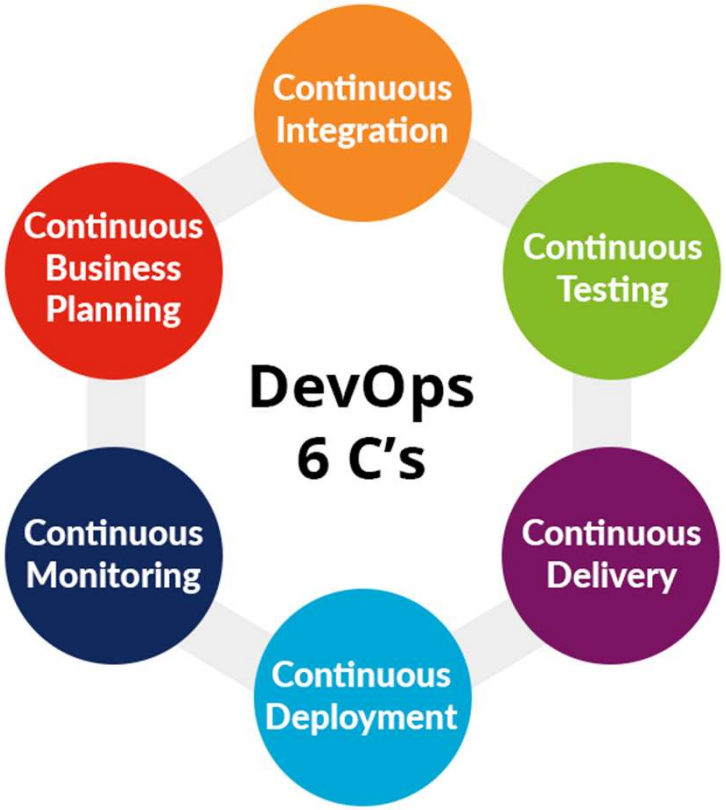

# DevOps

*"DevOps is a set of practices and cultural changes — supported by automation tools and Lean processes — that creates an automated software delivery pipeline, enabling organizations to deliver better-quality services and applications faster."*

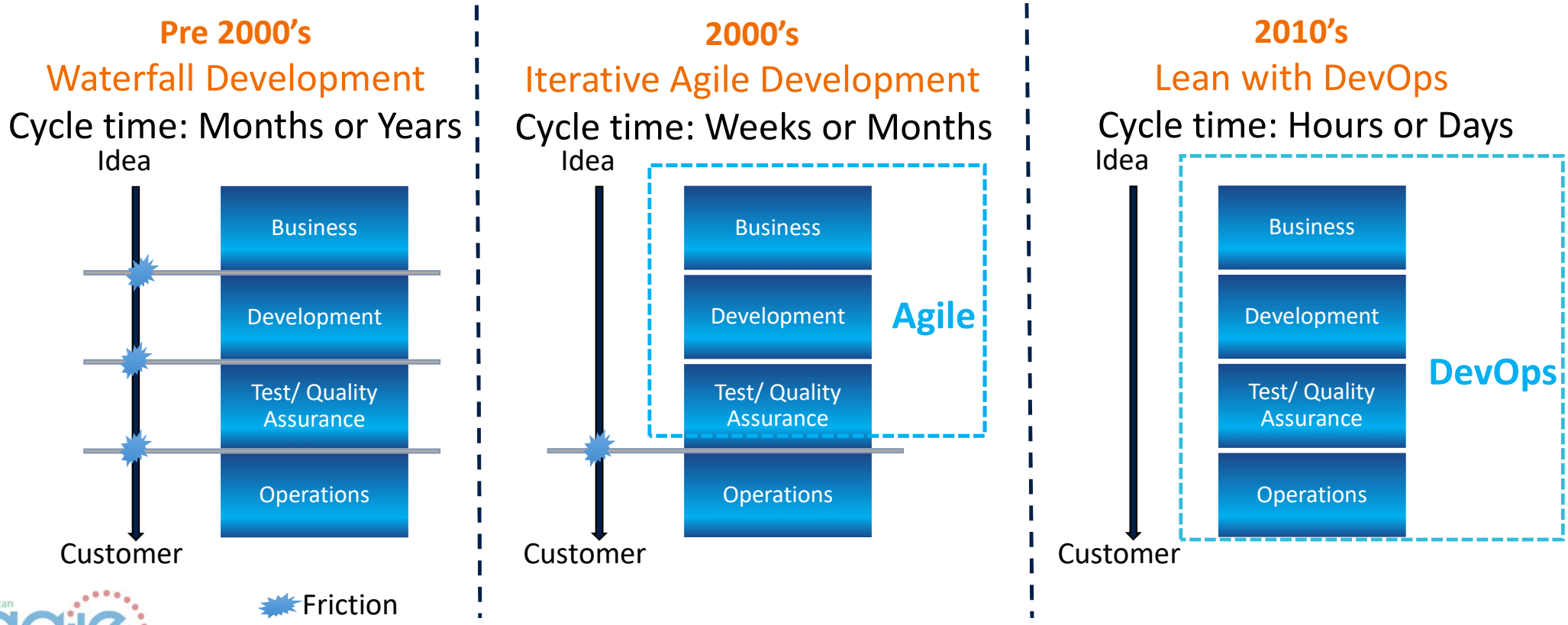
Forrester



# Elaborating DevOps

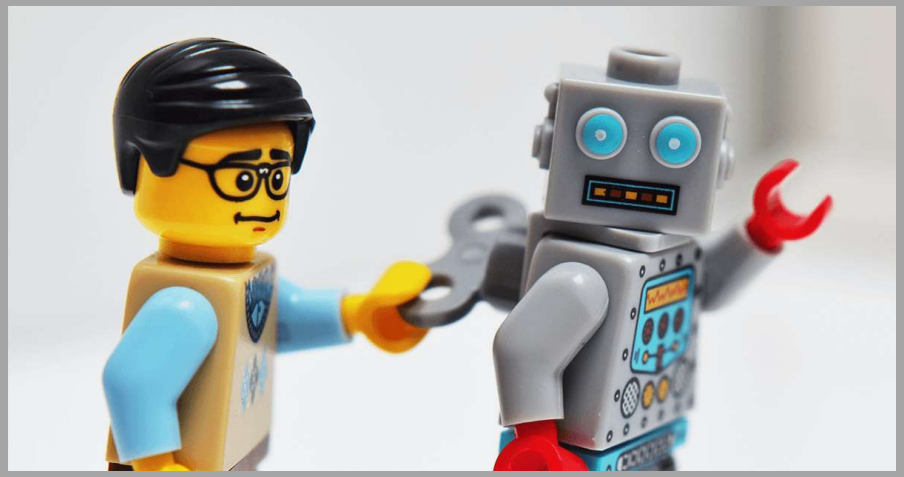


# Background

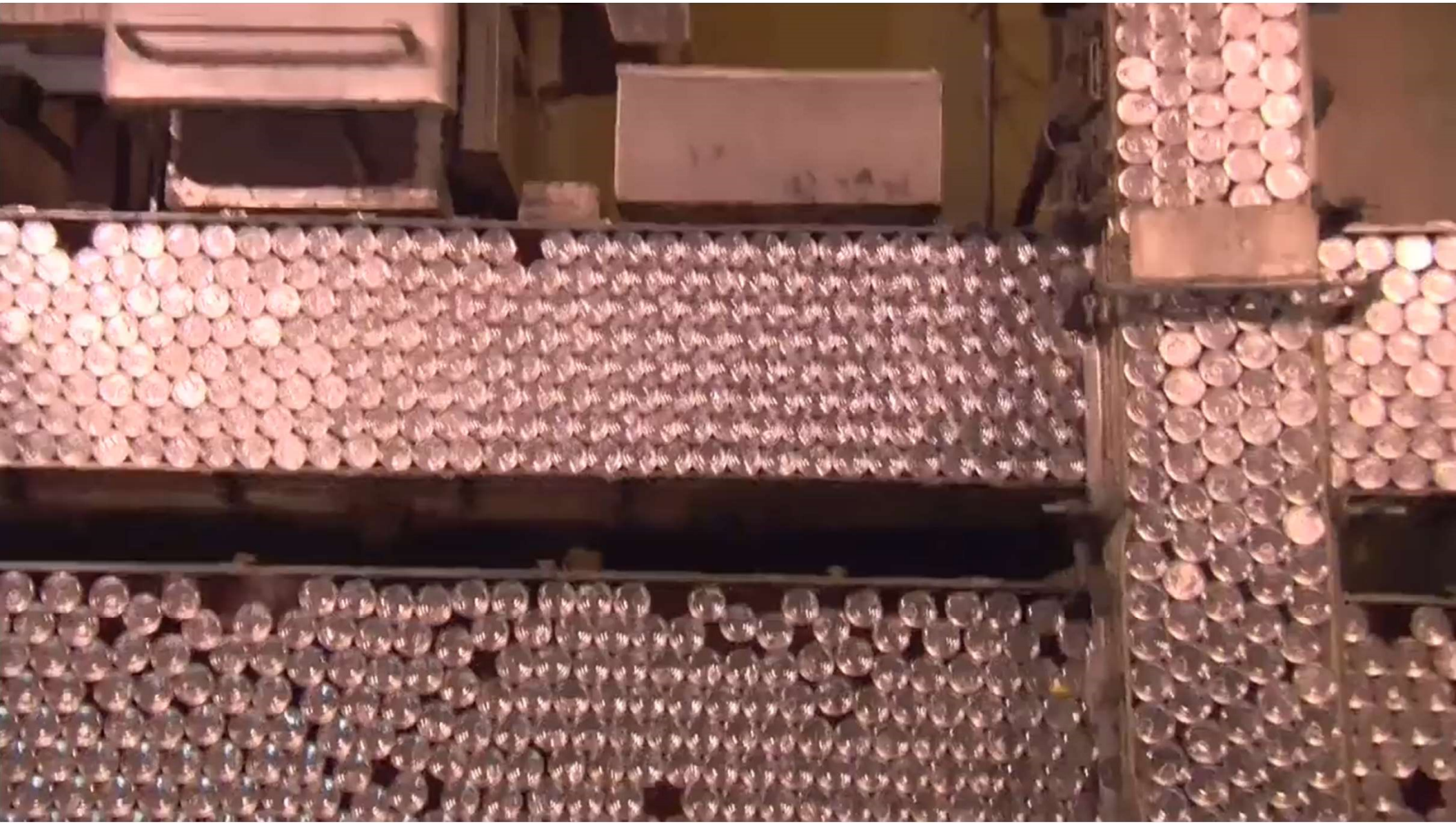

Pre 2000's
Waterfall Development
Cycle time: Months or Years
Idea
Business
Development
Test/ Quality Assurance
Operations
Customer
Friction
2000's
Iterative Agile Development
Cycle time: Weeks or Months
Idea
Business
Development
Test/ Quality Assurance
Operations
Agile
Customer
2010's
Lean with DevOps
Cycle time: Hours or Days
Idea
Business
Development
Test/ Quality Assurance
Operations
DevOps
Customer

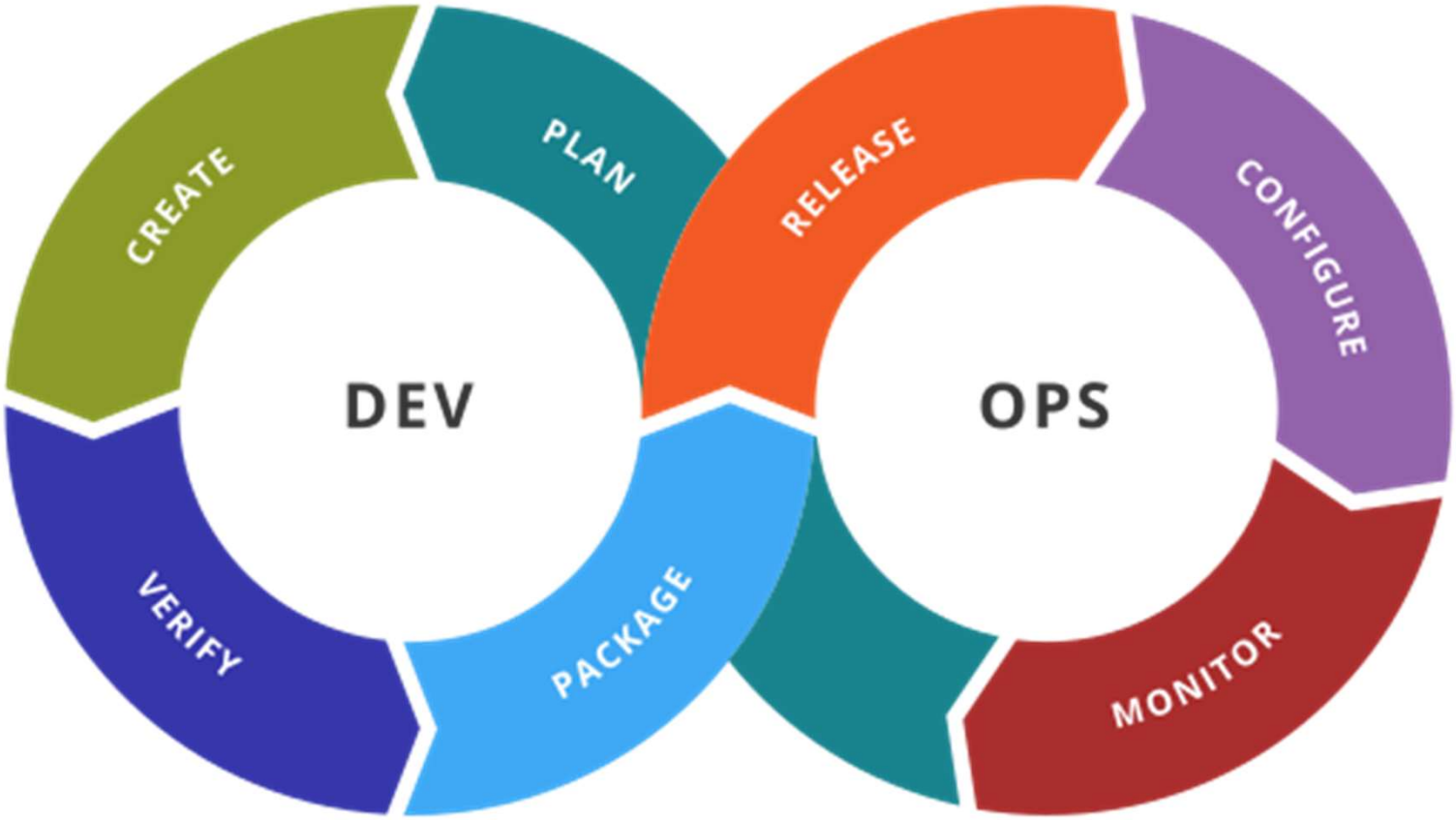
# Problem 1

Automating Software Delivery Pipeline



"If you automate a mess, you get an automated mess"





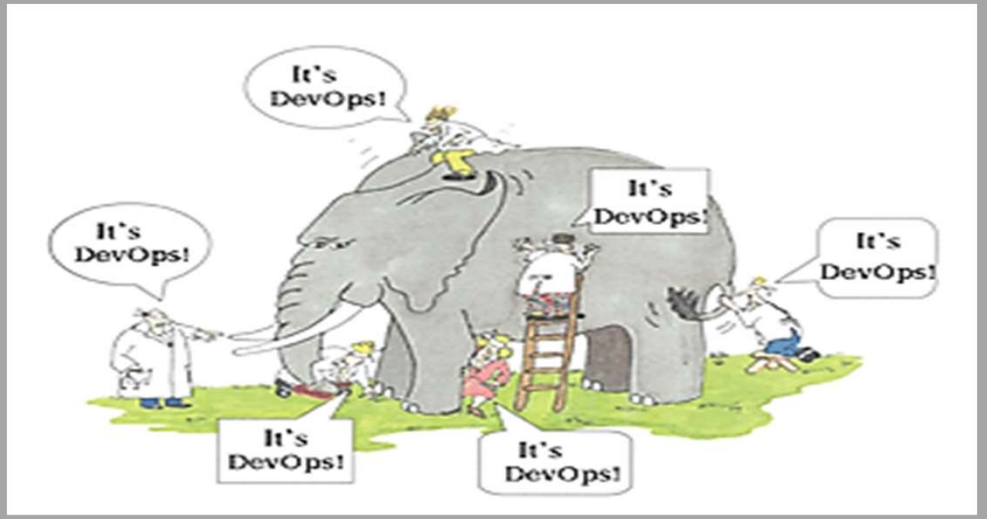

Automated Software Delivery Pipeline

# Problem Space

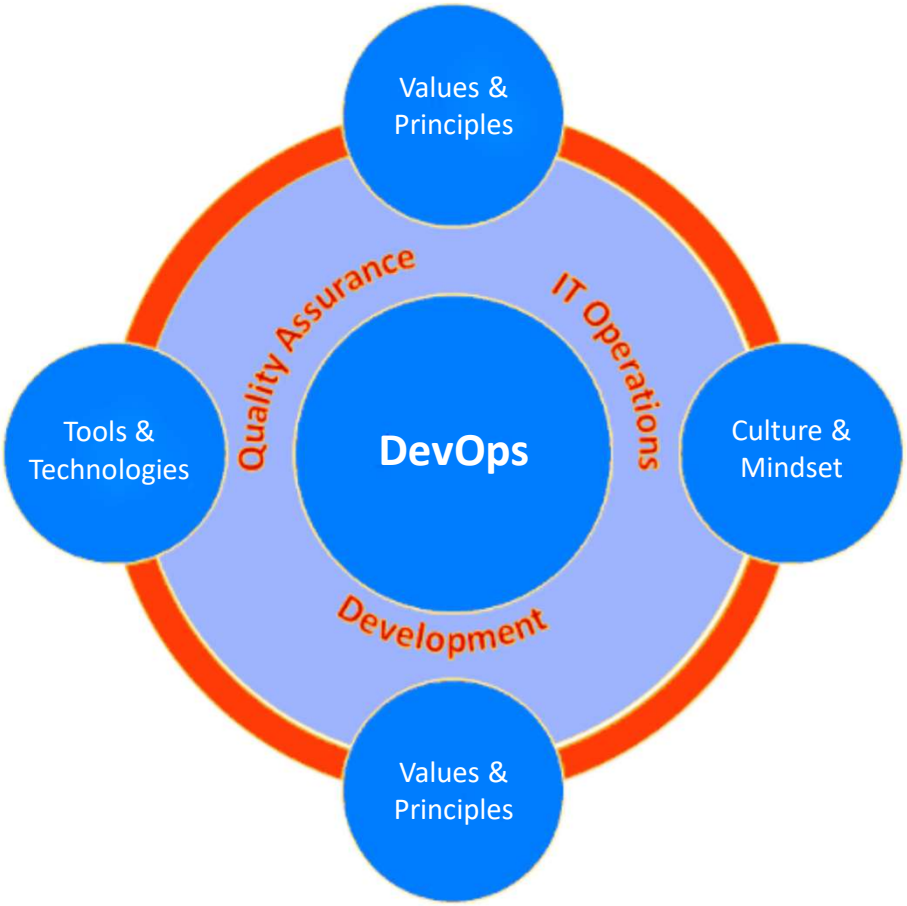
# Solution Space

# Problem 2

Aligning Various Perspectives




*"Any organization that designs a system will inevitably produce a design whose structure is a copy of the organization's communication structure"*




# DevOps Perspectives

# Problem Space

**O** rganization

rientation

wnership

# Solution Space

rganization

rientation

wnership

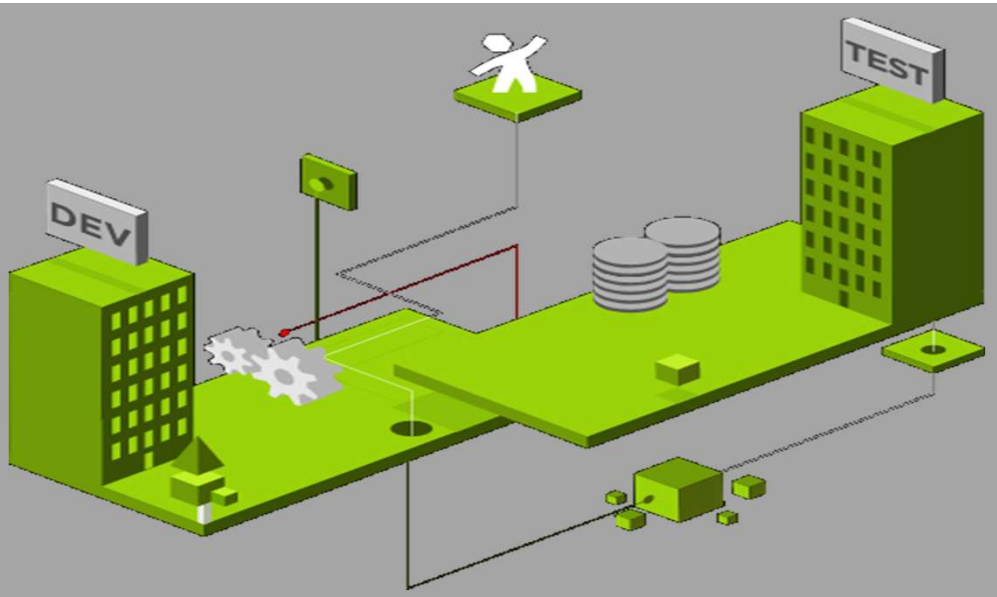
Use DevOps Recommended Patterns

Collaboration

Trust, Empowerment, Self-managed Teams

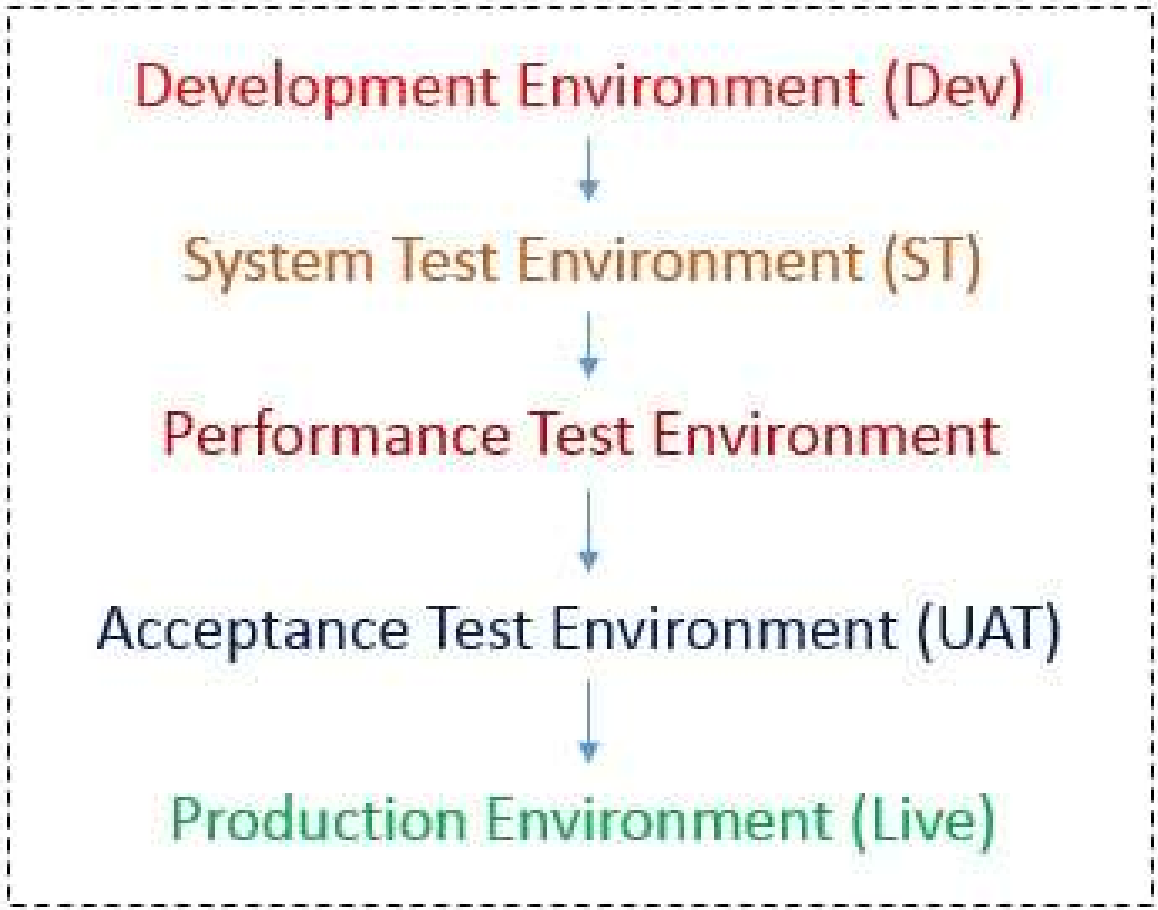
# Problem 3

Environments Management




*"Move fast with stable infrastructure"*

Mark Zuckerburg




Environments Management

# Problem Space

nfrastructure

ntegration

solation

# Solution Space

nfrastructure Plan, Budget, Codify

ntegration Auto-triggering, Rollback, Alerts

solation SREs, Synchronization, Config Management

# Problem Checklist

Automating Software Delivery Pipeline

Aligning Various Perspectives

Environments Management

# Opportunities

Faster and Reliable Development and Delivery

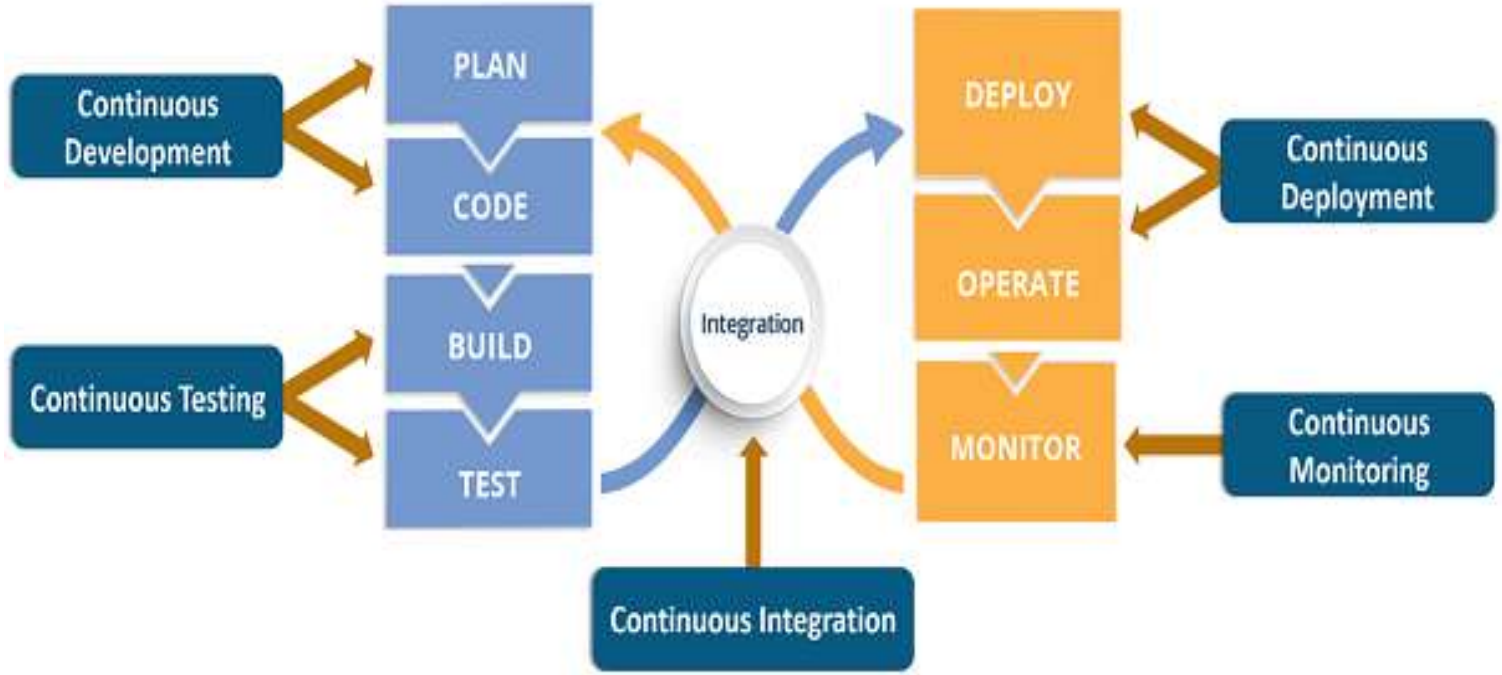
*“You build it, you run it*

Werner Vogels

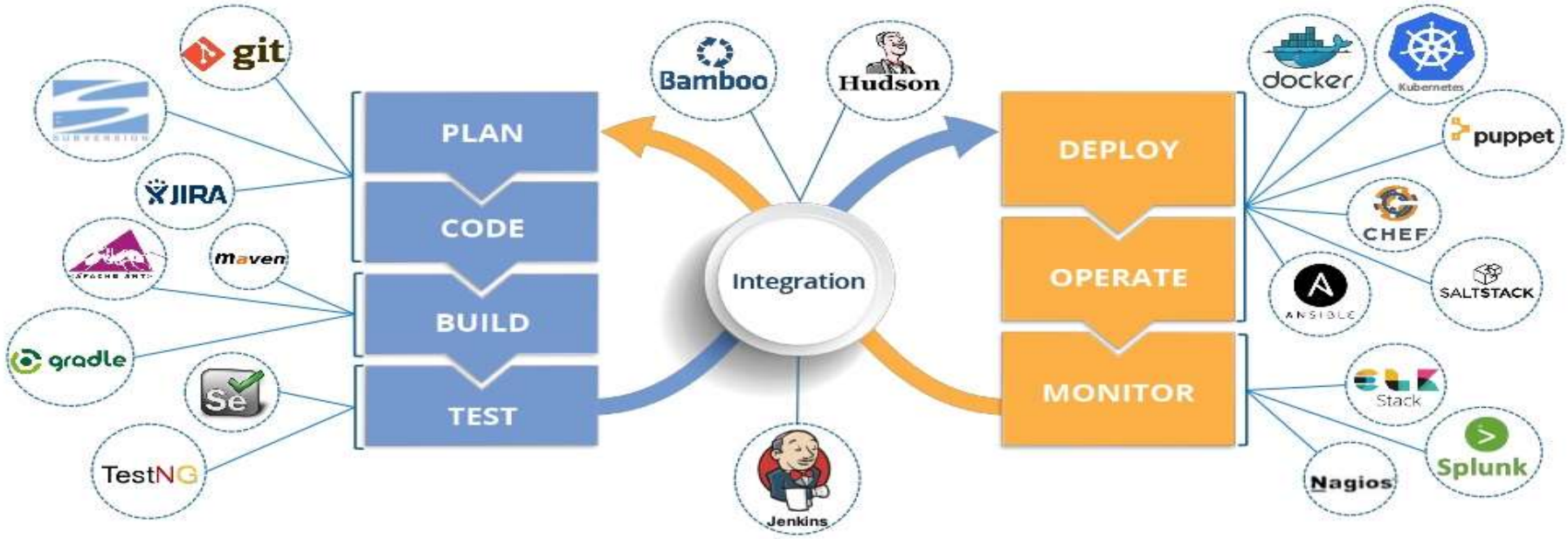
# Reasons


Quicker Deployment Cycles
Increase in Product Quality
Faster Innovation Cycles
Reduction in Suspension of Operations


# Life Cycle Phases in DevOps

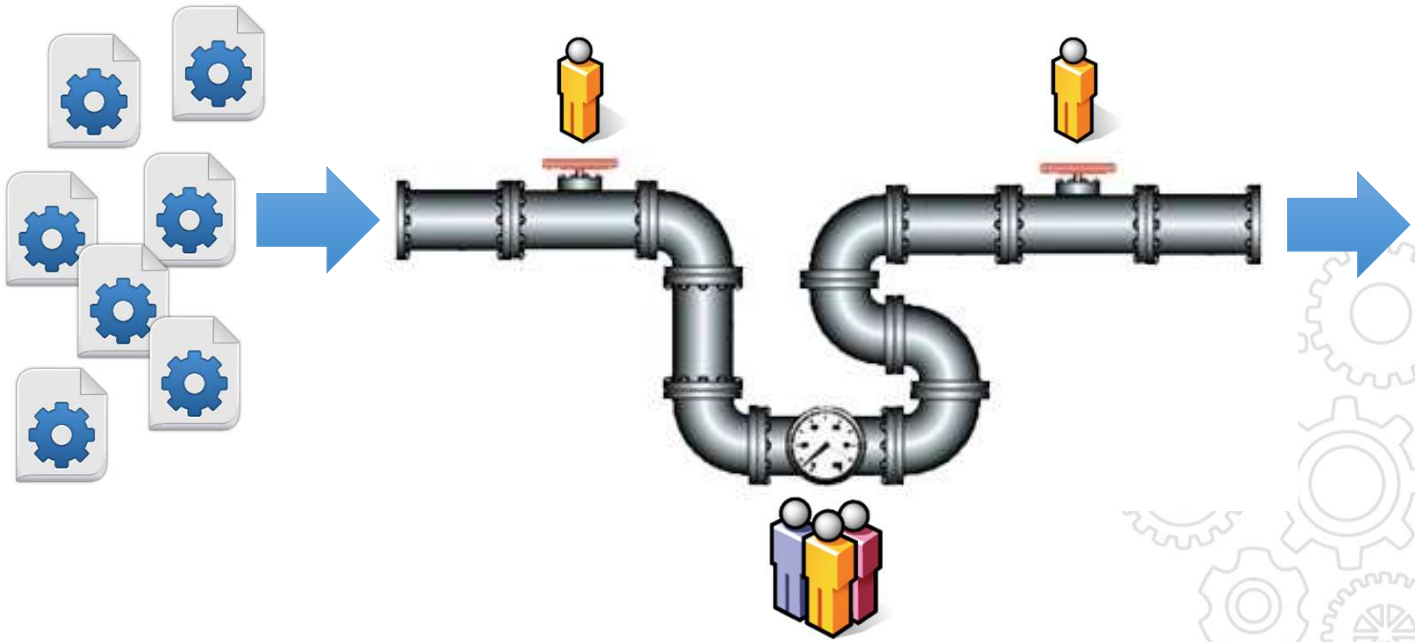



# Popular DevOps tools



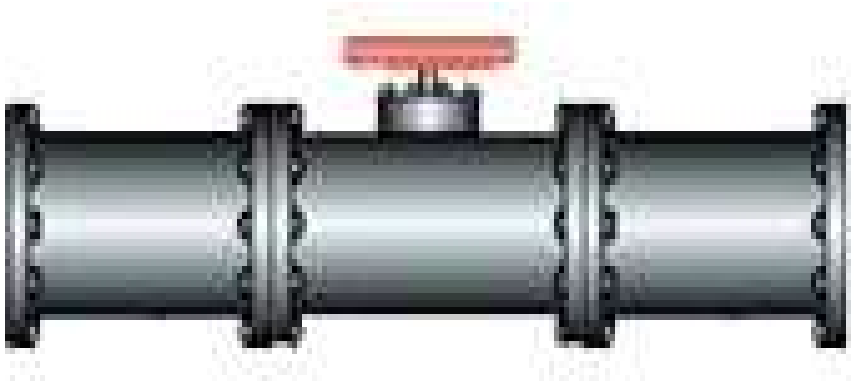

# DevOps

More collaboration and automation between the development and operations teams = larger pipeline




# DevOps

Remove the hand offs – streamline the process, challenge everything that doesn't add value.




How often do the high performing organizations deploy their code now a days?

# Benefits

Impact and success stories

*“You build it, you run it”*

Werner Vogels


# 2019 DevOps Report

**Deployment Frequency**

- 75X higher

**Lead Time for Changes**

- 440x faster

**Mean Time to Recover**

- 96x faster

**Change rate failure**

- 5x lower

# 1 DEVOPS IN TRADITIONAL HIGH TECH: HP LaserJet Firmware Group

**400-800 developers**
around the world at any given time

{ } Supporting
**10 million lines of code**

**Only 2 releases**
a year

## Problem:

Majority of developer time spent importing existing firmware into products

No time to write new features

## What did they do?

- Eliminated code branching
- Automated testing
- Implemented a "stop the line" culture

## Results:

Commits per day:
**1 → 100**

Regression testing time:
**6 weeks → 24 hours**

% time spent writing new features:
**5 → 40**

## Lessons learned:

1. Slow development cycles stifle creativity and new feature development
2. Long feedback cycles make it harder to prevent mistakes in the future

# 2 IBM Success story

| Environments | Manual | | Automated | | Gain |
|---|---|---|---|---|---|
| | Deploy | Tests | Deploy | Tests | |
| Test | 4 hours | 80 hours | 20 min | 3 hrs | 96% |
| Staging | 8 hours | 4 hours | 40 min | 15 min | 92.5% |
| Production (25 environments) | 200 hours | 4 hours | 3 hours | 5 min | 98.5% |
| **Gain** | | | 98% | 96% | 98.5% |

# 3 DEVOPS IN BANKING: Lloyds Banking Group

**FACT**

**Lloyds Banking Group can trace its history back to 1765 – no company is too old to implement a DevOps model!**

Fintech companies are disrupting the financial services industry – reaching younger customers, developing new payment methods, and quickly reacting to changes to the market. Traditional banks don't want to be left behind.

**Goal:**

Bring new or updated applications to market more quickly while also complying with government regulations, not damaging legacy applications, and improving testing quality.

**What they did:**

- Automate testing and repetitive tasks
- Remove waste from processes
- Stabilize testing environment
- Leverage 3rd party DevOps tools throughout the continuous integration, delivery, and deployment pipeline

**Result:**

**Testing cycle time:**
**9 days → 4 days**

**200% faster time to market**

# Career path

## Demand and skills

*“DevOps engineers are in high demand and this will not change in the future. They enjoy high salaries and some of the best conditions in the High-Tech industry.”*

Shahar Gotshtat


# Why to adopt devops as a career?


# DevOps as a Career

**156,209**

DevOps Engineers jobs currently available

**#2**

DevOps Engineers ranking in 50 best jobs in Glassdoor survey

**57%**

Open source expertise are focused on DevOps skills


**$104,158**

Average salary for a DevOps Engineer in US

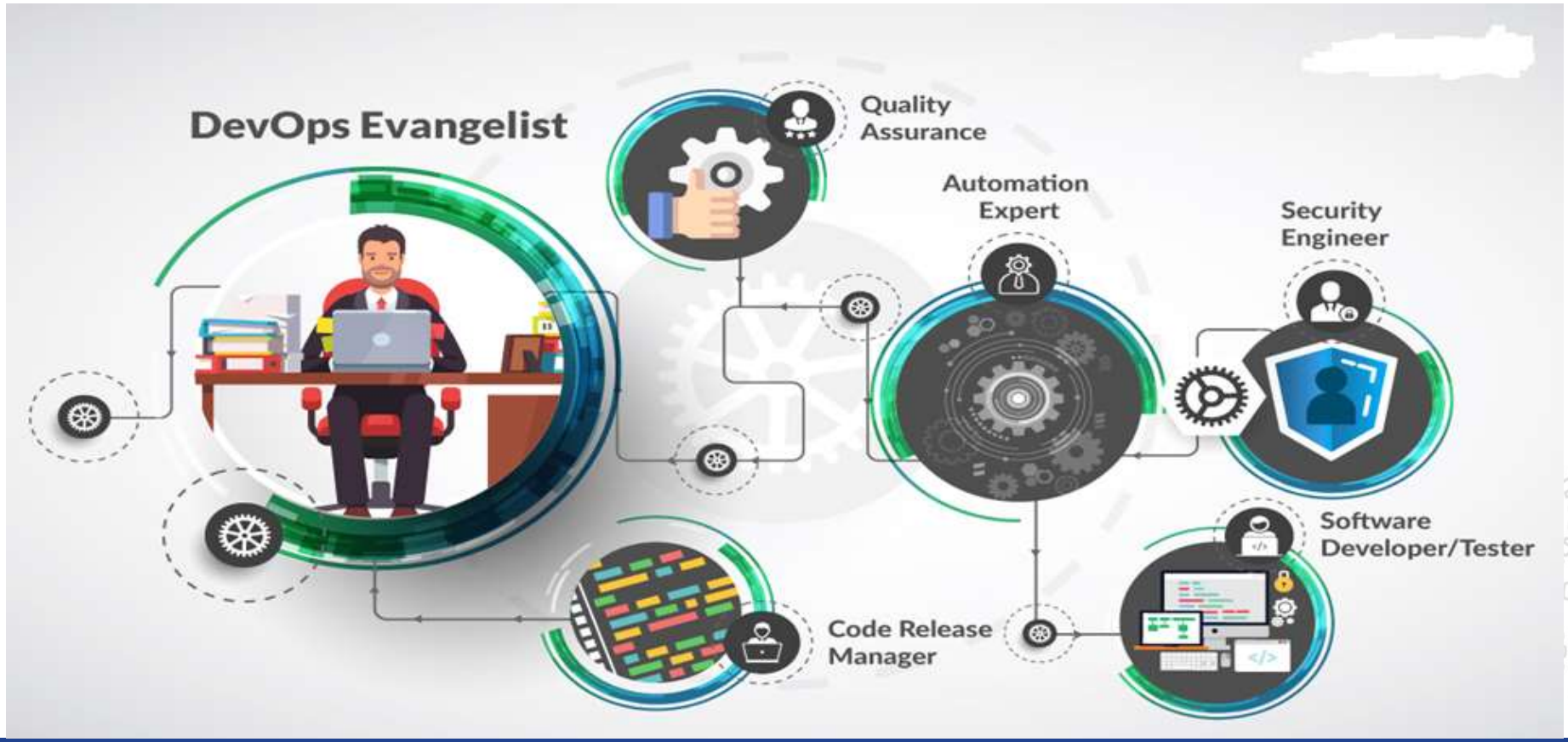
**60%**

Hiring Managers are looking to fill DevOps Engineers position

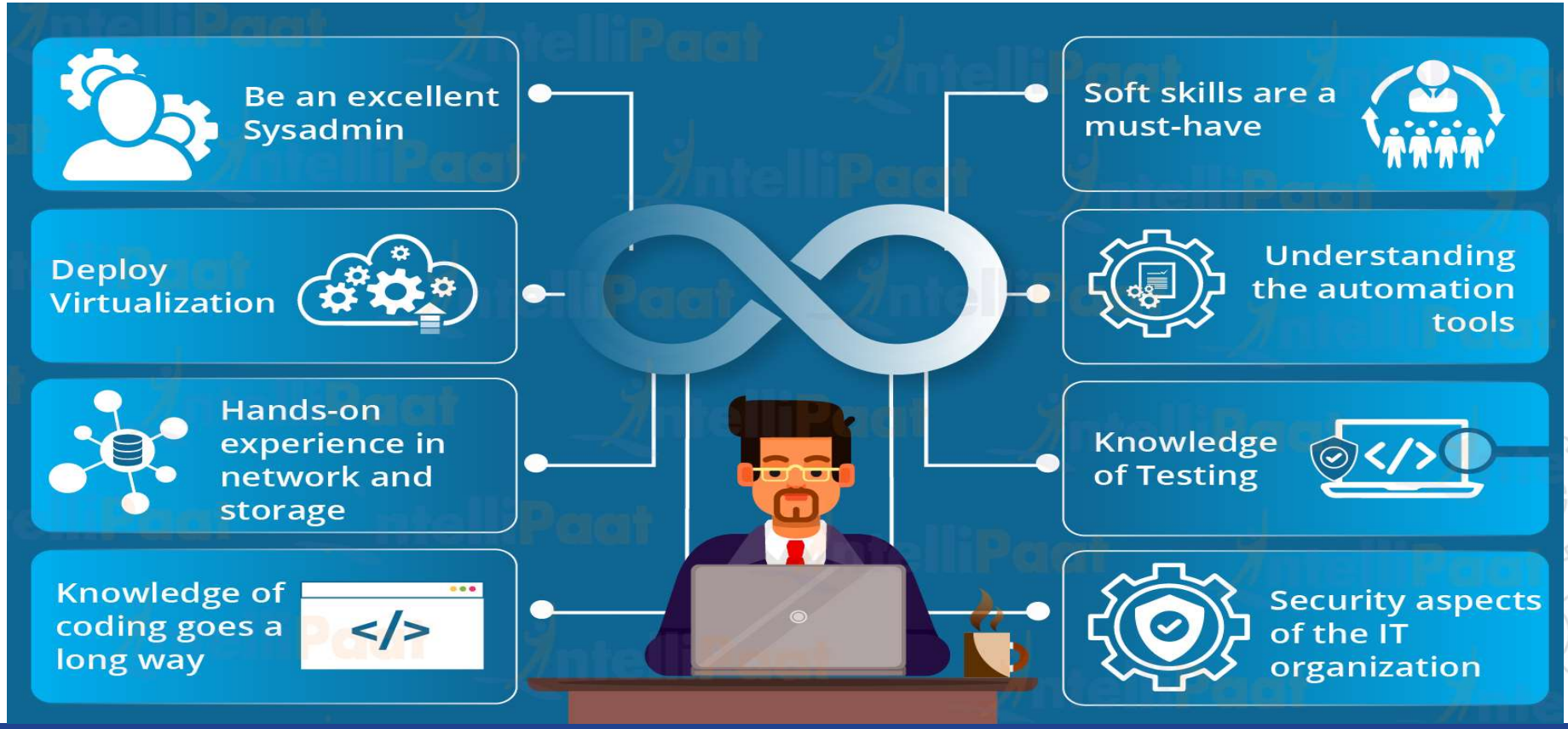
**42%**[36]

Open source jobs study want to add DevOps skills across their hiring portfolio

# Job responsibilities of a DevOps engineer




# Knowledge & Skills You Need As A DevOps Engineer



# Questions?

# Learning Objectives

**Define DevOps** according to software industry thought leaders

Discuss important **components** of DevOps

Define **CAMS** and **CALMS** acronyms

Relate **Lean Management** to DevOps

# What is Lean Software Development (LSD)?

Lean Software Development (LSD) is an agile framework based on optimizing development time and resources, eliminating waste, and ultimately delivering only what the product needs. The Lean approach is also often referred to as the Minimum Viable Product (MVP) strategy, in which a team releases a bare-minimum version of its product to the market, learns from users what they like, don't like and want to be added, and then iterates based on this feedback.

## Lean Values in DevOps

Break down historic silos

Improve **collaboration** between
**Development** and **Operations**

Streamline and **improve work** in key ways

Cultural focus: **Deliver value** to customer

In order to achieve these aims in DevOps, the idea is to break down historic silos and improve collaboration between development and operations teams to streamline and improve work in some key ways. It used to be that separate teams wrote the code, tested the code, deployed the code, and then maintain the code during a product's lifecycle. The idea behind DevOps is to break down all those barriers and get everyone collaborating from the get-go on the same team. Ultimately, DevOps is about people, which is really what Lean management approaches are about as well. It's about creating a culture of focusing on delivering value for the customer

# Key DevOps Acronyms

| CAMS | CALMS |
| --- | --- |
| Culture | Culture |
| Automation | Automation |
| Measurement | Lean |
| Sharing | Measurement |
| | Sharing |

Back in 2010, two major DevOps thought leaders, Damon Edwards and John Willis, coined the acronym CAMS, C-A-M-S, which stood for Culture, Automation, Measurement, and Sharing. Later, Jez Humble, another thought leader in the DevOps community, who's authored a number of books, decided to add Lean to the mix, creating the acronym CALMS.

## Defining DevOps

"DevOps is about **humans**. DevOps is a set of practices and patterns that turn **human capital** into high-performance **organizational capital**."

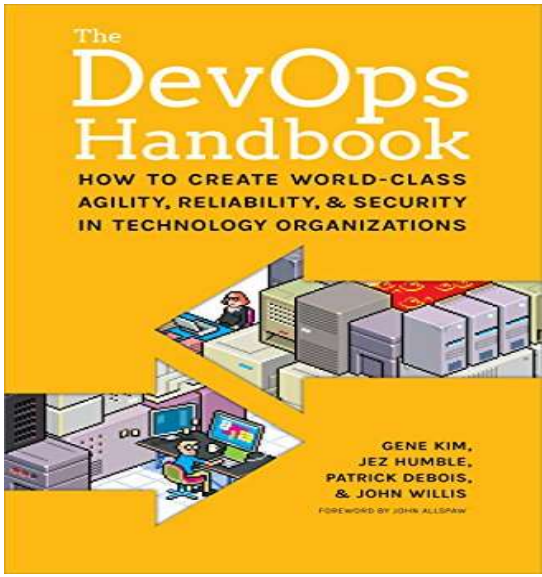
– *John Willis*

**Director of Ecosystem Development, Docker**

# Another DevOps Definition

"DevOps is the emerging professional movement that advocates a **collaborative** working **relationship** between Development and IT Operations, resulting in the fast flow of planned work (i.e., high deploy rates), while simultaneously increasing the **reliability**, **stability**, **resilience** and **security** of the production environment."

*– Gene Kim*



## A Full Lifecycle

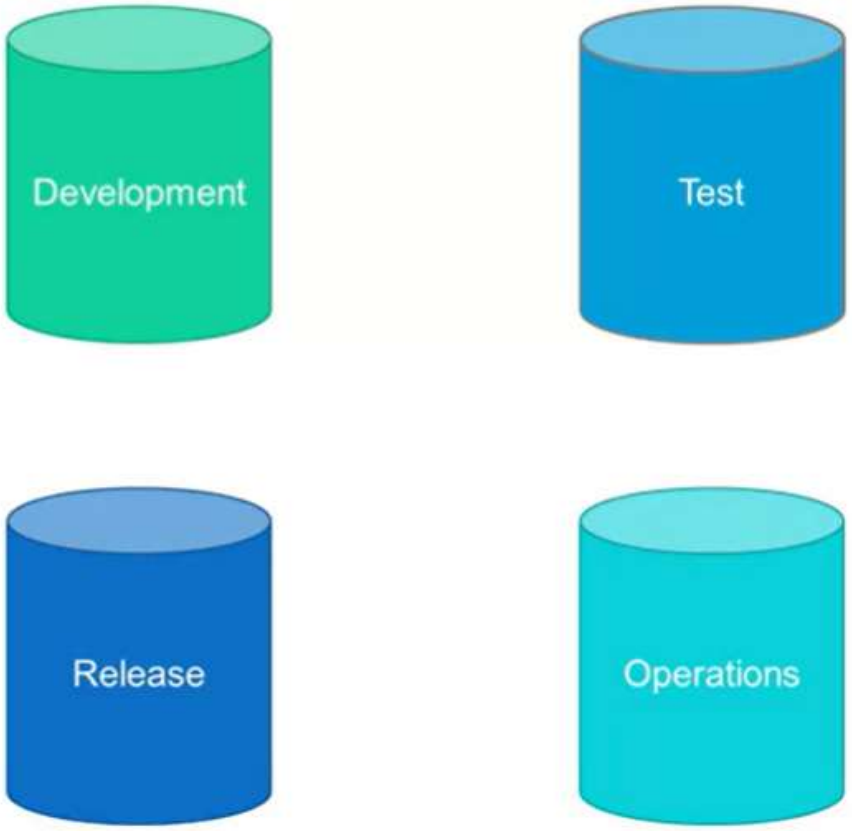
**Building**, **Deploying**, **Operating** and **Supporting**

**DevSecOps** = security teams incorporation

Entire **organization** needs to transform!

Some of the key components of that definition or that it's not just about technology,
it's not just about process, and it's not just about people,  it's really the combination of all three that make DevOps come to life. It's about a full life cycle of delivering value, building, deploying, operating, and supporting.
the industry has also acknowledged the need to incorporate security teams into that definition. Sometimes even seeing the phrase DevSecOps.

## From 4 Silos into 1 Team




Traditionally, a lot of organizations have a development team,
test team, release team,
and operations team, security team,
and business teams, and what DevOps aims to do is to break down those silos and make the entire team accountable for all of those things instead of just handing them off.

# DevOps Principles: The Three Ways

## Introducing The Three Ways

1. Systems thinking
2. Amplifying feedback loops
3. A culture of continuous experimentation and learning

Developed by Gene Kim and Mike Orzen

## Systems Thinking

Emphasizes **performance** of entire system

**Collaboration** across functional lines

Focuses on IT-enabled **value streams**

The first way, systems thinking really emphasizes the performance of the entire system as opposed to the performance of a specific silo work group or team. This is a really important elements of DevOps.

DevOps is about collaboration across functional lines, really breaking down those silos and focusing on the value streams that IT enables.

## Amplifying Feedback Loops

**Feedback Loop:**

A process that allows for reflection on its own output before determining the next steps that need to be completed

The outcomes of the second way include understanding and responding to all customers, internal and external, shortening and amplifying all feedback loops, and embedding knowledge where we need it.

# Ways to improve feedback loops

- One way is to build automated tests into the pipeline so developers can get feedback early and often.

- Another way is to embed operations engineers into the development teams. That way the team will learn from others.

- Quality Dashboard

It showed the health of our automated test scripts that ran every night, and if a certain threshold was met, in our case, the percentage of tests that failed, than the dashboard would turn red. We would halt the release until we were back within our acceptable quality range. This is an example of a feedback loop. Our teams could easily see early and often if we had quality issues, and could deliver that feedback to the developers.

## A Culture of Continual Experimentation and Learning

**Create** that culture!

Encourage **risk-taking** & **failing forward**

Affirm that **repetition** in practice is a **prerequisite to mastery**

The outcomes of the third way are about making sure that you're allocating time for that improvement work.

You're rewarding teams for taking risks.

I think the third way is the most challenging for most organizations.

# DevOps Principles: The Seven Principles and Seven Wastes of Lean

1. Eliminating waste

- Which is making sure that you're not coding more features than needed.
- You're minimizing handoffs, and really not producing anything that is of low or no value.
- This also means making sure that you're making decisions at the right time.
- Sometimes, organizations will delay decisions, waiting on additional information, and worrying about being wrong.

# DevOps Principles: The Seven Principles and Seven Wastes of Lean

2. The second principle is building quality in

Deming quote.

"Quality is everyone's responsibility".

- This principle is about making sure that you have quality built into the product and into the process.

# DevOps Principles: The Seven Principles and Seven Wastes of Lean

3. The third principle is creating knowledge,

Deming quote.

"Quality is everyone's responsibility".

- This principle is about making sure that you have quality built into the product and into the process.

# DevOps Principles: The Seven Principles and Seven Wastes of Lean

4. The fourth principle, is differing commitment

- Make Decisions at the right time
- After analysis and considerations
- Delay decisions until you have more info

Because it allows you the time to gather more information before committing to something

# DevOps Principles: The Seven Principles and Seven Wastes of Lean

5. Deliver Fast

- Ensure feedback received early and often
- Allow for course correction
- Smaller batches allow you to deliver faster

# DevOps Principles: The Seven Principles and Seven Wastes of Lean

6. Respect People

- Lack respect for people in your Culture?
- Lean and DevOps falls apart

People are at the center of DevOps and Lean. If you aren't practicing that then it typically falls apart. I believe passionately that people are an organization's number one asset. Often though, organizations don't behave that way. When I first started getting exposure to DevOps and Lean, I realized that fundamentally it's about respect for people.

# DevOps Principles: The Seven Principles and Seven Wastes of Lean

7. Optimize The Whole

- This again is a systems thinking approach.
- You want to make sure that you're not doing local sub-optimization,  and that you are optimizing for the entire system.

# A3 Problem Solving Framework

| A3 Steps | PDCA Cycle |
|---|---|
| Background | Plan |
| Problem Statement/Current Condition | |
| Goal/Target Condition | |
| Analysis | |
| Proposed Countermeasures | Do |
| Implementation Plan | Check |
| Follow Up Actions | Act |

What's really funny regarding this framework, is that people are always asking, why is it called an A3?

There is not much to it
actually, really,A3 just refers to a European paper size,
that's equivalent to an 11 by 17 inch size paper.

# A3 Problem Solving Framework

**Step 1: Set the Background**

Include statement of how problem directly impact business outcomes

**Step 2: Current Condition and Problem Statement**

Current reality = where things stand today

**Step 3: Develop the Goal**

Target state you are trying to achieve

**Step 4: Performance Analysis**

Identify root cause, there may be multiple factors

**Plan Phase: Step 1-4**

spend time on the plan phase! Great planning leads to successful outcomes

# A3 Problem Solving Framework

**Step 5: Brainstorm**

How do you intend to reach the target condition?

Come up with a hypothesis on how countermeasure can help reach goal

# A3 Problem Solving Framework

**Step 5: Brainstorm**

How do you intend to reach the target condition?

Come up with a hypothesis on how countermeasure can help reach goal

**Countermeasures**

Decision criteria : effort Vs. impact

# A3 Problem Solving Framework

**Step 6: Implementation Plan**

Enable you to check results

Confirms impact on current conditions

For example, we will create 15 additional automated test scripts within two weeks, and it's assigned to our QA manager.

# A3 Problem Solving Framework

**Step 7: Update**

**Update "Standard work" based on steps taken**

Then based on the results, step seven is when you're going to update what's called standard work, based on the outcome of the steps that you took.

For example, if you automate it those test scripts, and it reduced your production incident count,
then you will want to check those into a library, and have those become your standard automated scripts going forward.

# A3 Problem Solving Framework

This A3 problem solving framework, can then be used as an ongoing tool, to create a culture of continuous improvement, or continuous learning.

# A3 Problem Solving Framework

This A3 problem solving framework, can then be used as an ongoing tool, to create a culture of continuous improvement, or continuous learning.

# Revision

Your organization has decided to kick-off a DevOps transformation and has asked you to help with defining the term for a communication that's being sent to the entire organization. How would you define DevOps?

1. "DevOps is about humans. DevOps is a set of practices and patterns that turn human capital into high-performance organizational capital."
2. DevOps is about creating a team and naming it the DevOps team.
3. DevOps is NoOps.

# Revision

Imagine that you are leading the Customer Mobile App Team at your company. You continue to have production defects after a release. Which of the 3 Ways would most likely help you remedy this problem?

1. Systems Thinking
2. Amplify Feedback Loops
3. Creating a culture of experimentation and continuous learning

# Revision

As the manager of the QA team, you continue to receive escalations from development teams saying that they are unable to move fast because they are always waiting on your team to finish running tests. What type of waste is happening in this situation?

1. Defects
2. Handoffs
3. Delays
4. Relearning or revisiting decisions
5. Extra features
6. Partially done work
7. Task switching

# Revision

A team in your organization says they are using the A3 framework to solve a problem with a high number of defects happening after releases. They aren't having a lot of success and when you look at the A3, they haven't populated the first four sections of the framework. What suggestion would you make to them? (Select the 1 best answer.)

1. To spend the time to make sure they clearly understand the problem. The first 4 sections are part of the Plan phase and are the most critical to spend the most time on and to complete.
2. That it's o.k. as long as they have the Do, Check and Act sections populated.
3. Breeze through the plan phase to more quickly arrive at the critical "do" phase.