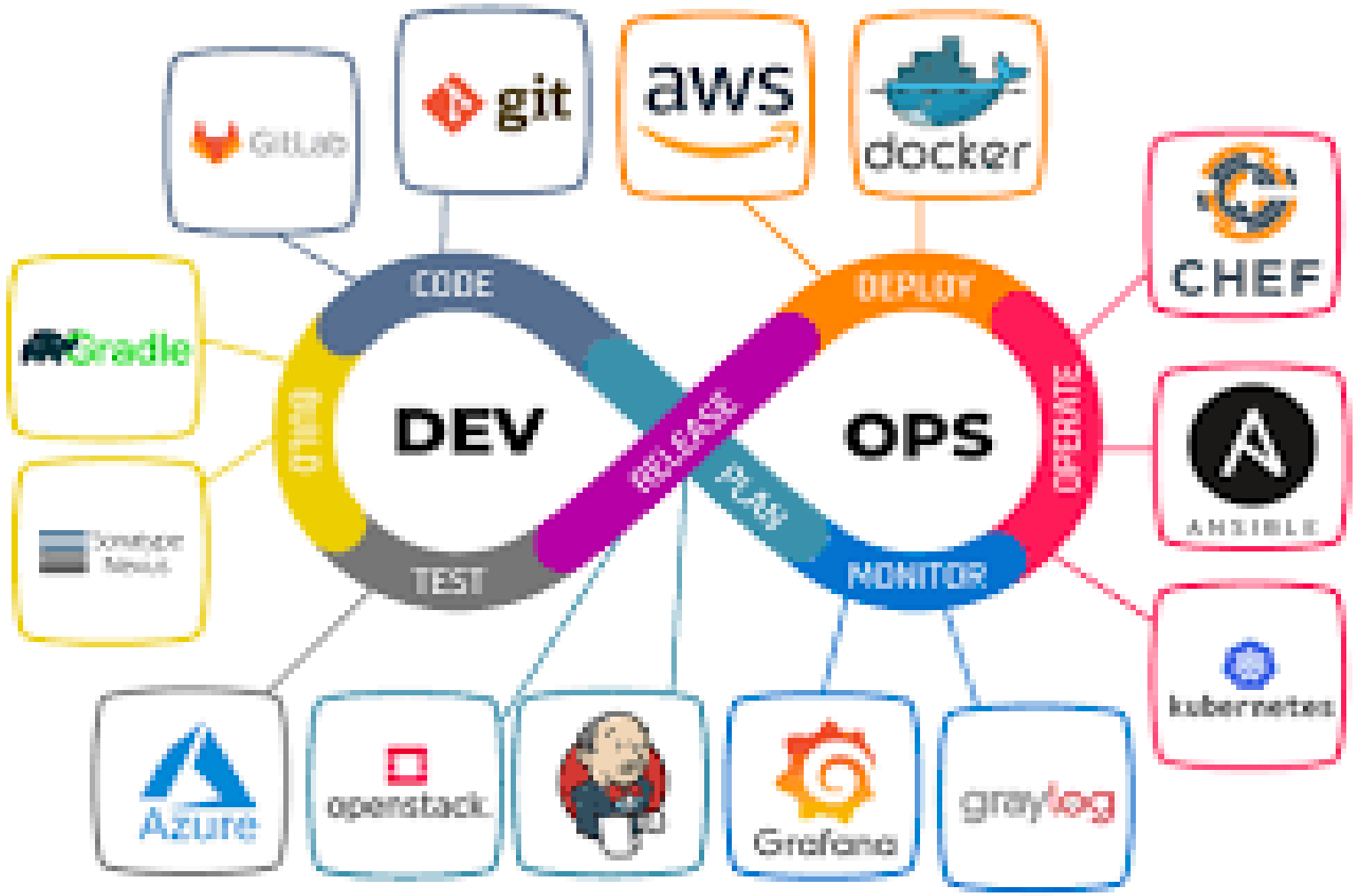
Introduction to DevOps



Mrs. Sujata Oak
Assistant Professor
Department of Information Technology
APSIT

Introduction to DevOps

Agenda

01 Evolution of software Develpoment over the years

02 What is DevOps?
Do we really need DevOps?

03 How does DevOps work anyways?
Challenges and problems out there

04 Devops Tools

Going to Market Faster: Most Companies Are Deploying Code Weekly, Daily, or Hourly

Business requires new features/changes

Business need new features/changes

Deployed in days/weeks

Not months

DEVELOPER

"Dev" is used as a shorthand for developers in particular, but in practice it is even wider and it means that "all the people involved in developing the product," that includes the product, QA and other disciplines.

Dev's job is to add new features

Ops's job is to keep the site stable and fast

OPERATIONS

"Ops" is a blanket term for system administrators, DBAs, Network engineers, Security professionals and various other sub-disciplines and job titles"

DEVELOPER

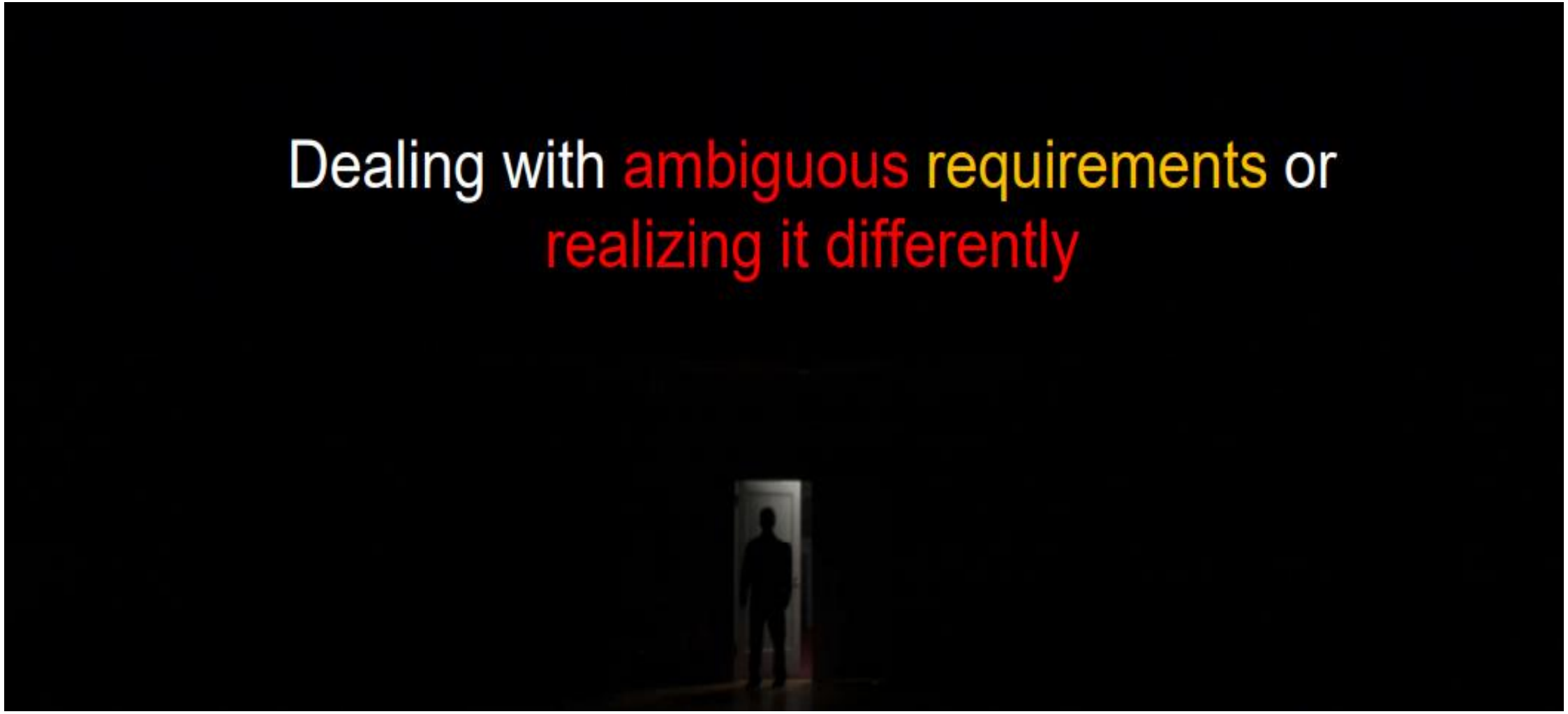
Ops's job is to enable the business (this is Dev's job too)

OPERATIONS

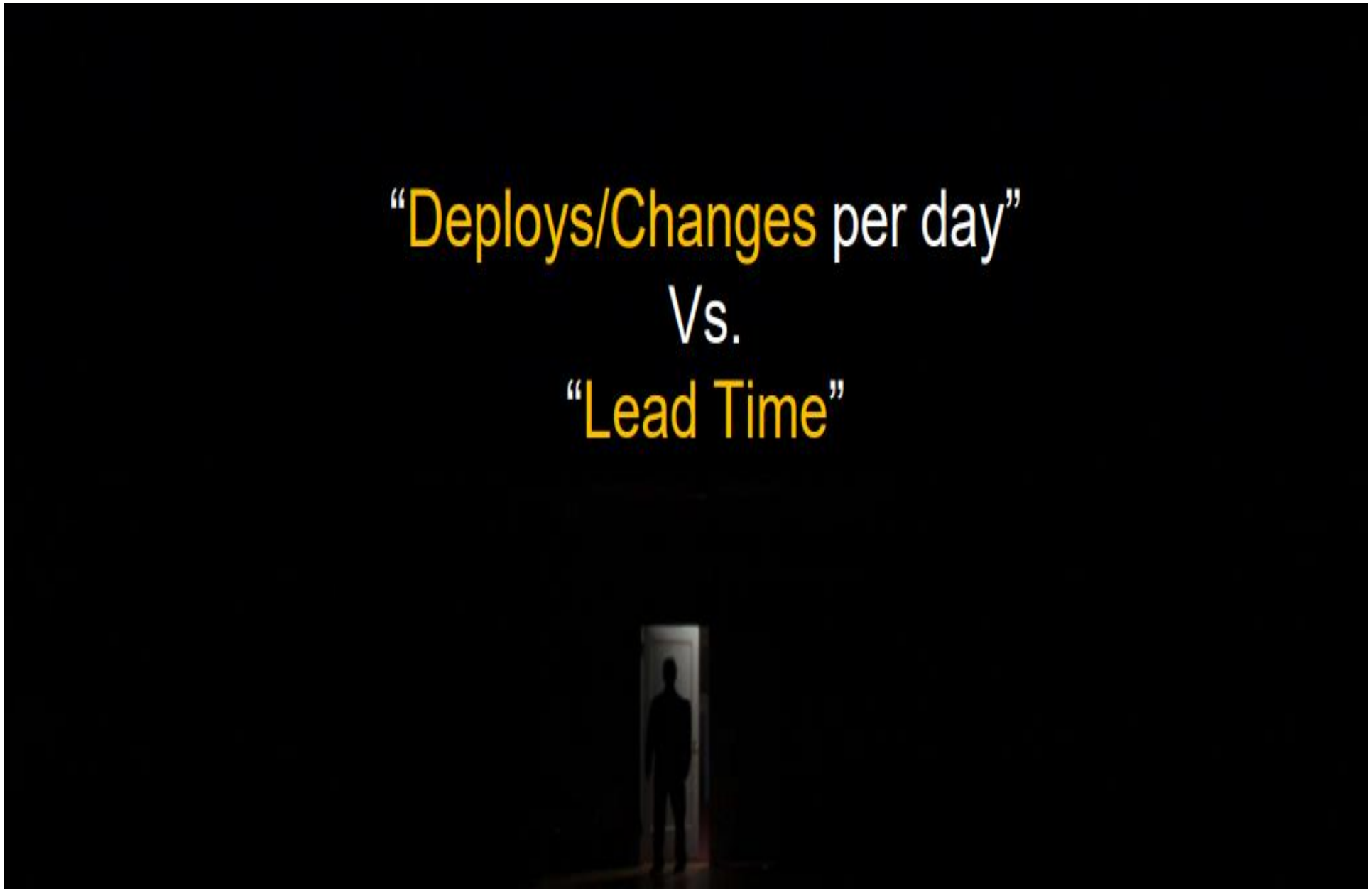
but code/features changes are integrated at the end of development.

Integration was a long and unpredictable process

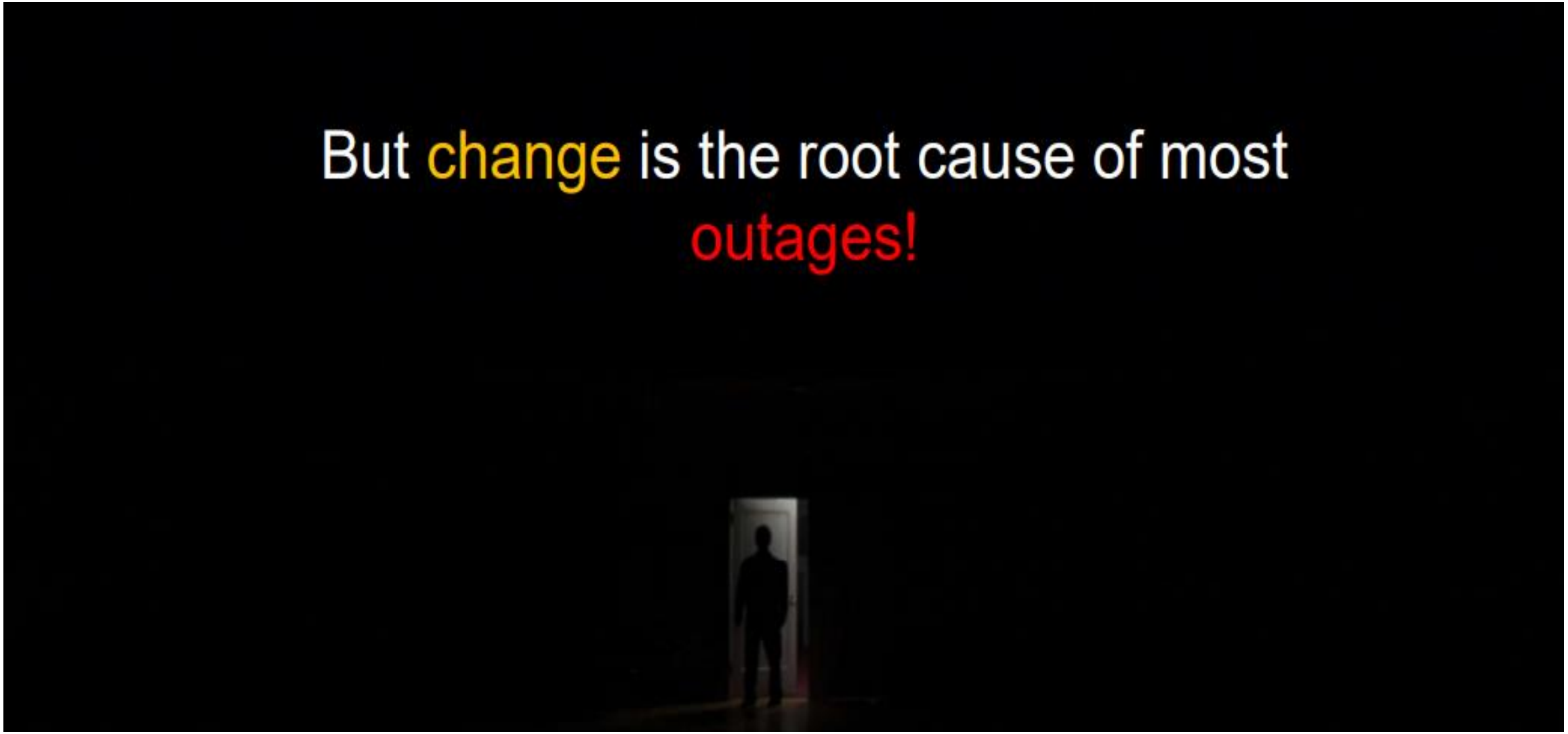
Lots of bugs are found at the end of testing phase

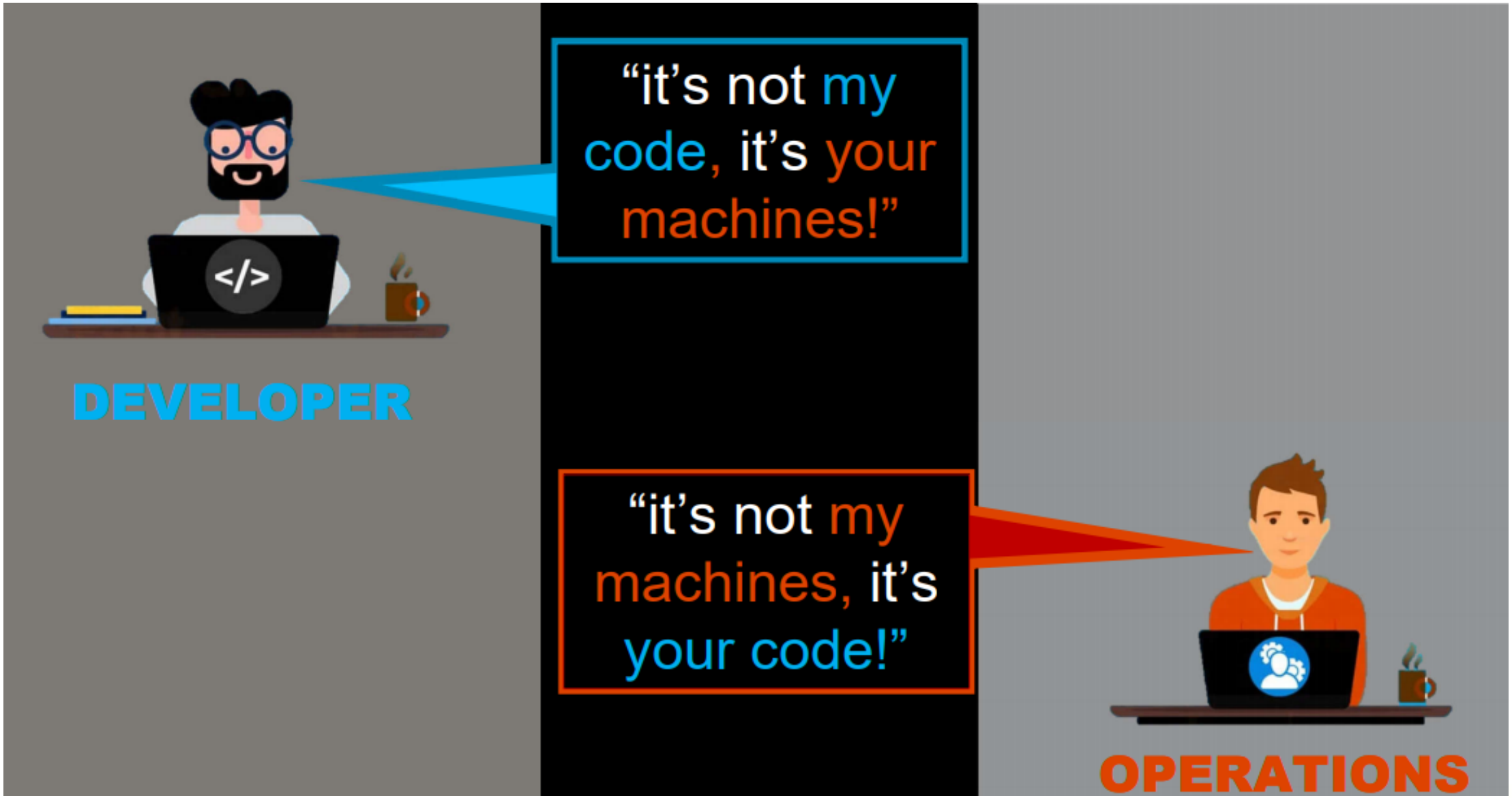

Dealing with ambiguous requirements or
realizing it differently

How long does it take for a committed code to move and run successfully in the Production Environment?


"Deploys/Changes per day"
Vs.
"Lead Time"

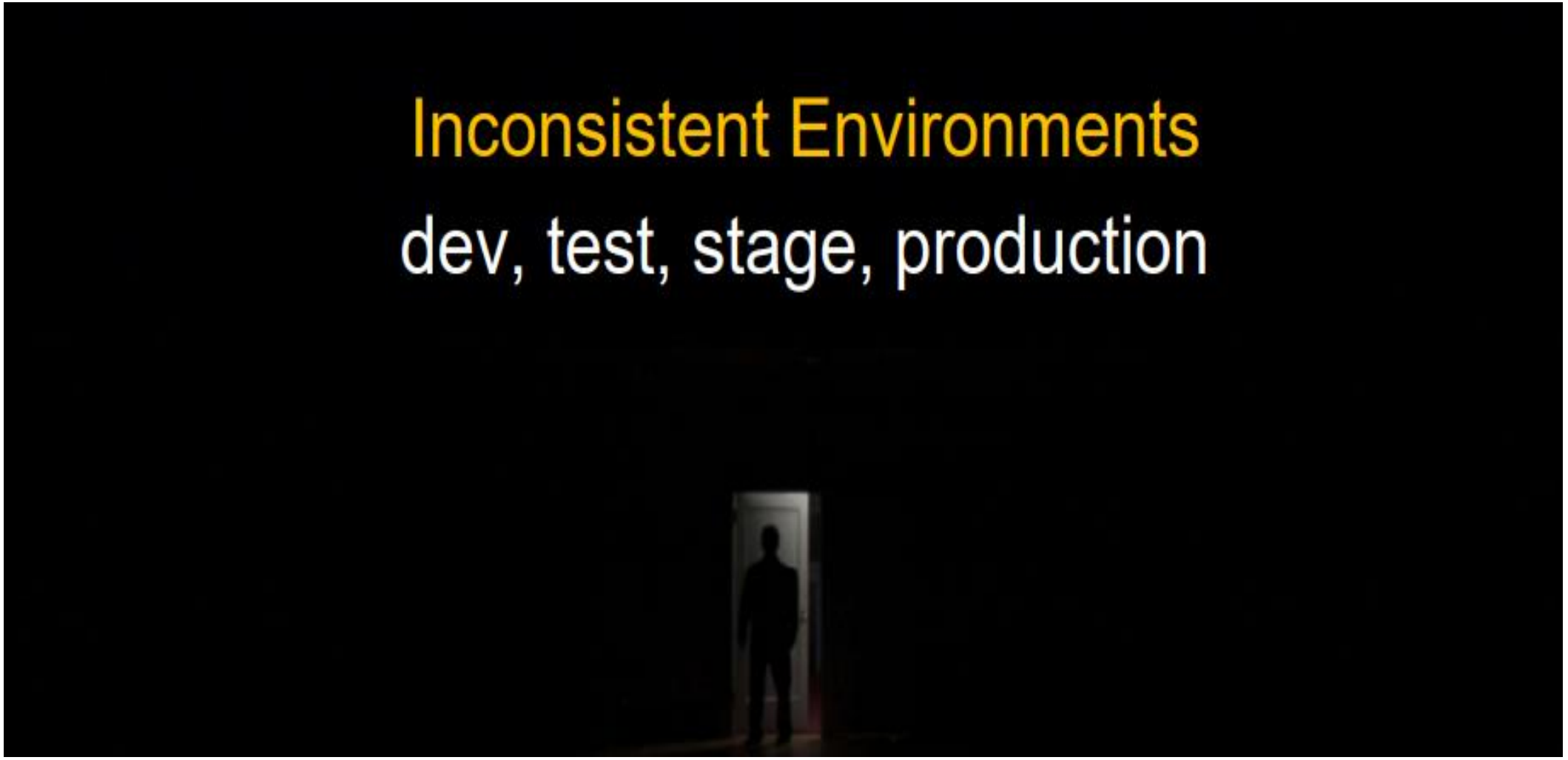
But change is the root cause of most outages!




"it's not my code, it's your machines!"
DEVELOPER
"it's not my machines, it's your code!"
OPERATIONS

Inconsistent Environments

dev, test, stage, production



DEVELOPER

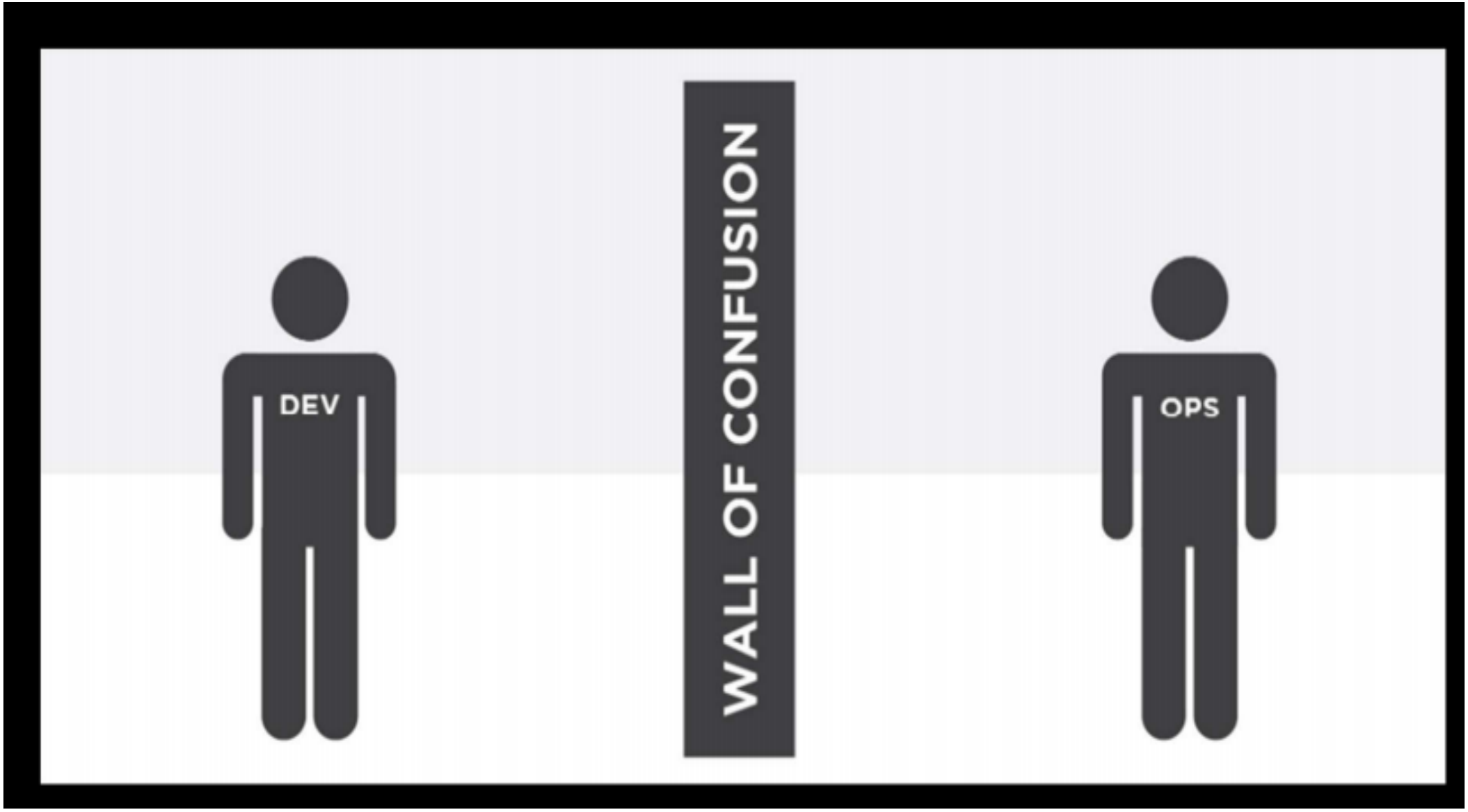
» Developers introduce changes

» They try to implement every new feature introduced

» Change is the enemy for Operations

» Changes can lead to instability

OPERATIONS


DEV
WALL OF CONFUSION
OPS

DEVELOPER

Sends out artifacts based on requirements

Day 1: During Infrastructure setup

Manually hacks the scripts received and changes the config files to reflect changes in production which could potentially lead to an issue.

OPERATIONS

Day 2: Loss of work

DEVELOPER

All the artifacts are fine, the error is because of some other issue. QA was given wrong setup

Developer gave faulty artifacts

OPERATIONS

Day 3: After Delivery

There are some database anomalies, code is not fit to be deployed on live server.

TEST ENGINEER

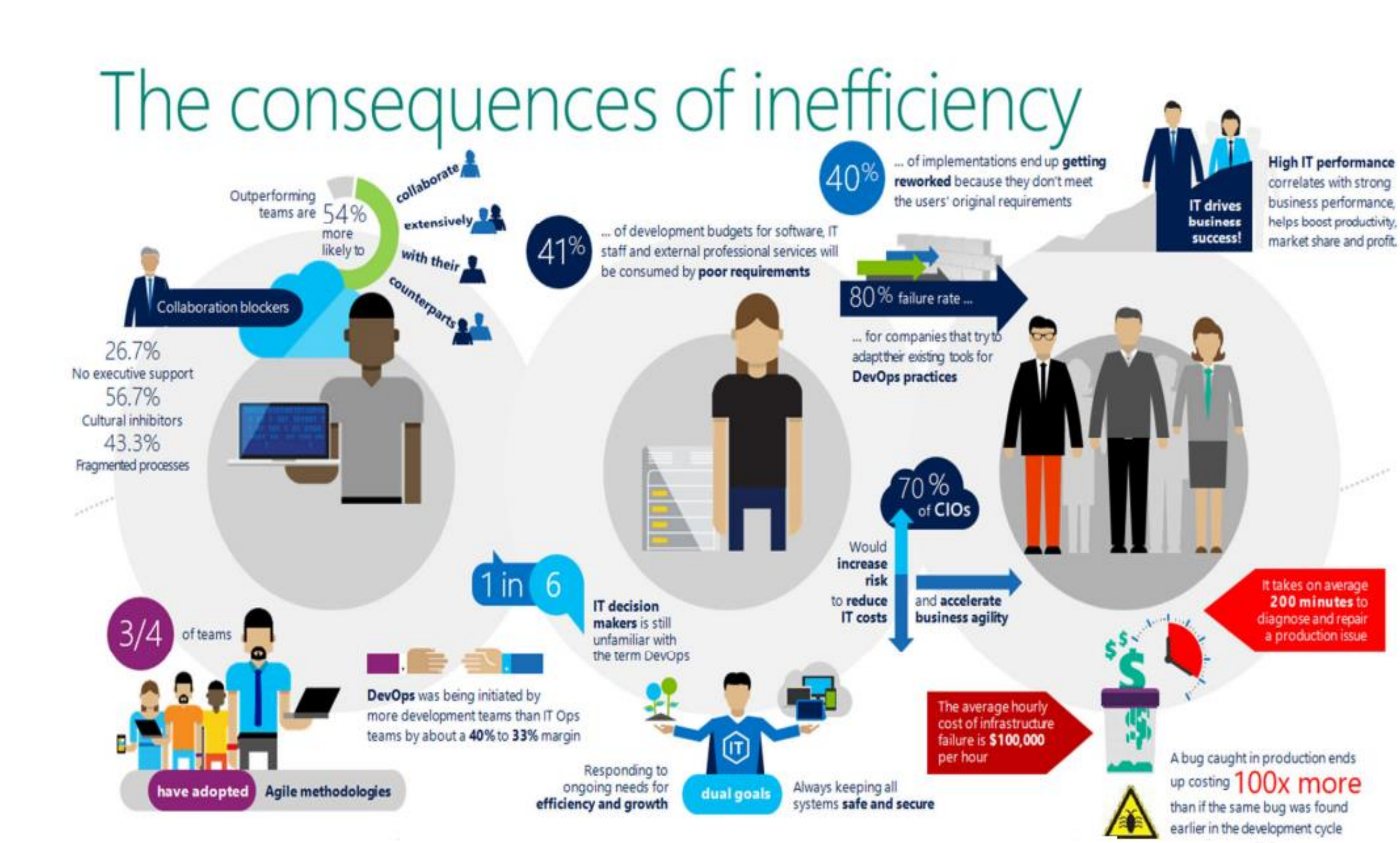
I checked at my end, seems like the database deployed on staging server by operations team is running an older version.

DEVELOPER

I was given wrong artifacts also constant changes in the code can be one of the reasons for failure.

OPERATIONS

The consequences of inefficiency

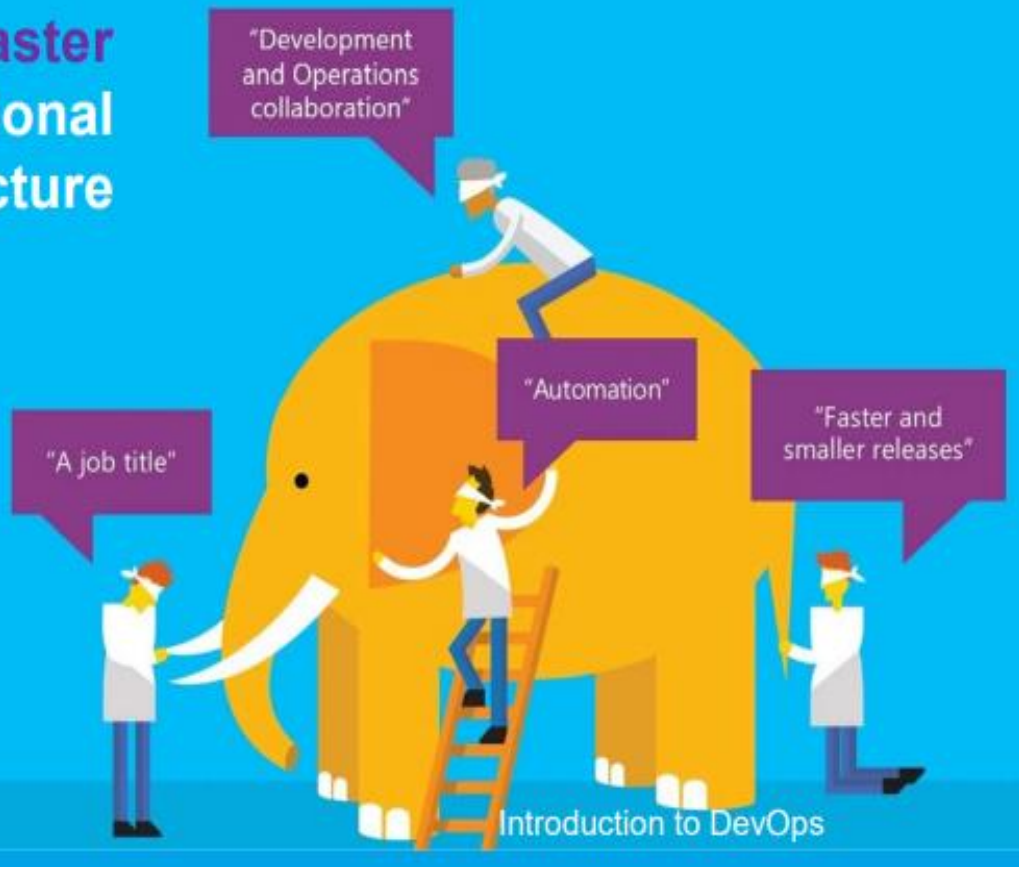


WHAT IS DEVOPS

DevOps is the combination of **cultural philosophies, practices,** and **tools** that increases an organization's ability to deliver applications and services at **high velocity.**

Evolving and improving products at a **faster pace** than organizations using traditional software development and infrastructure management processes.

This speed enables organizations to better serve their customers and compete more effectively in the market.



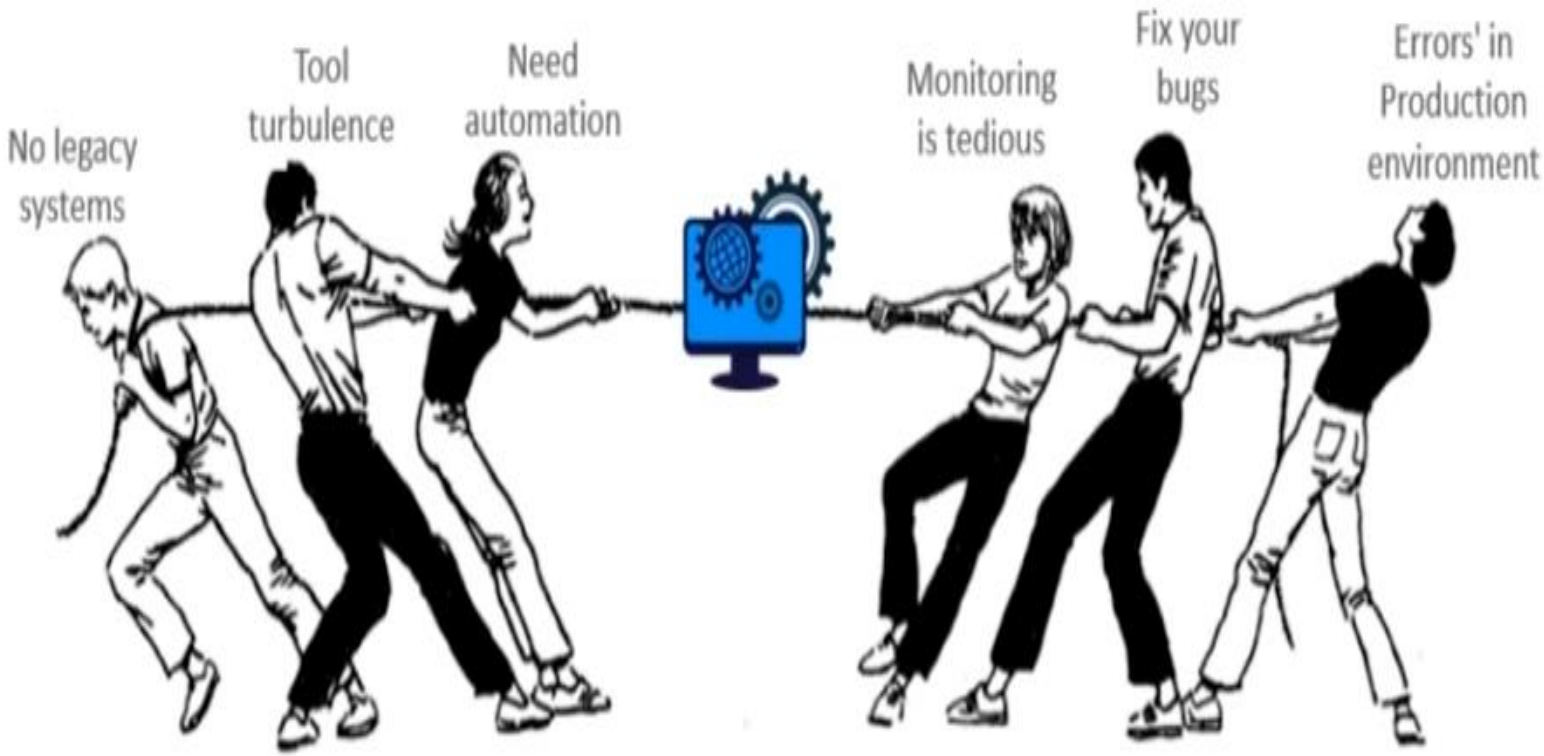
Collaboration

Communication

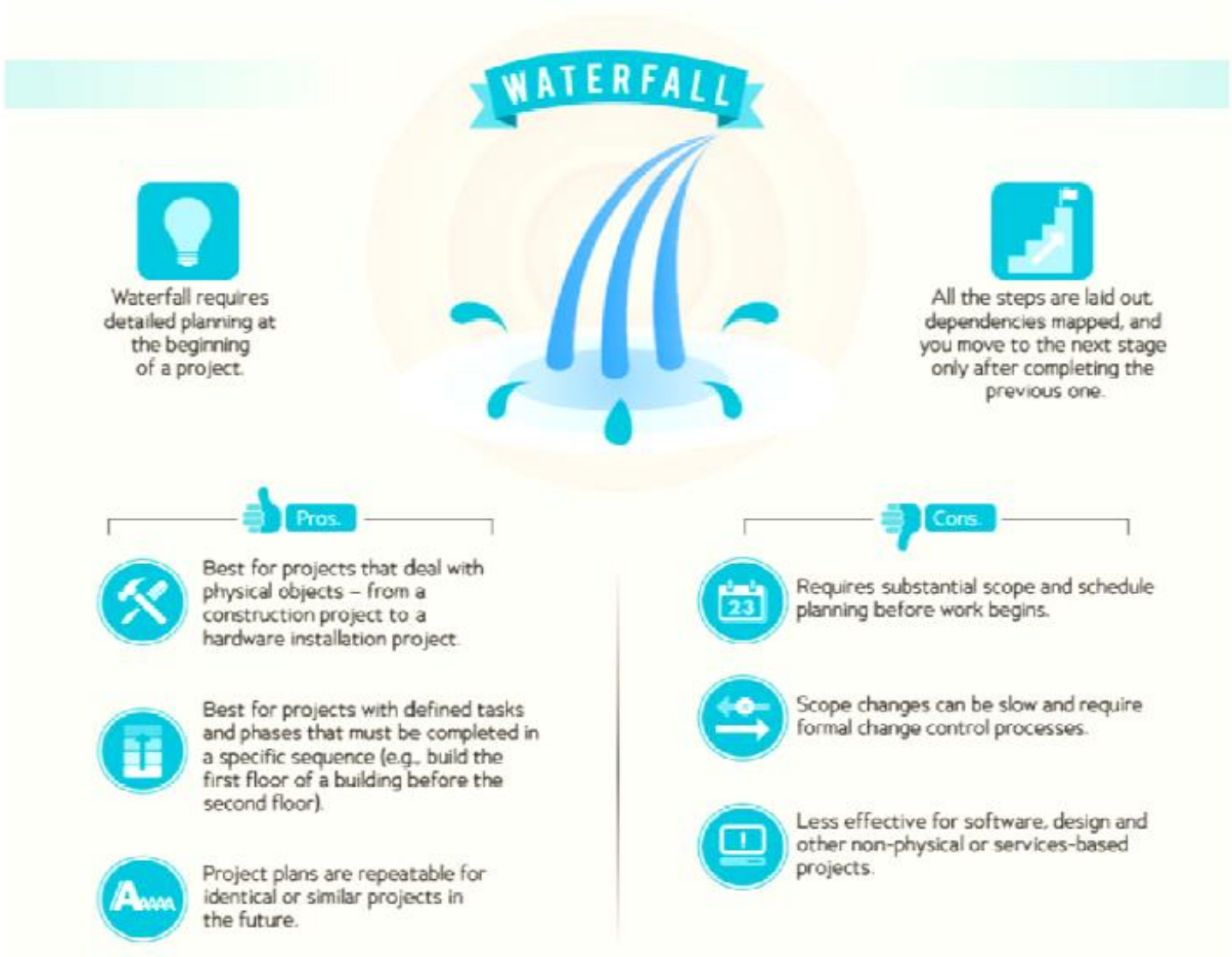
Integration

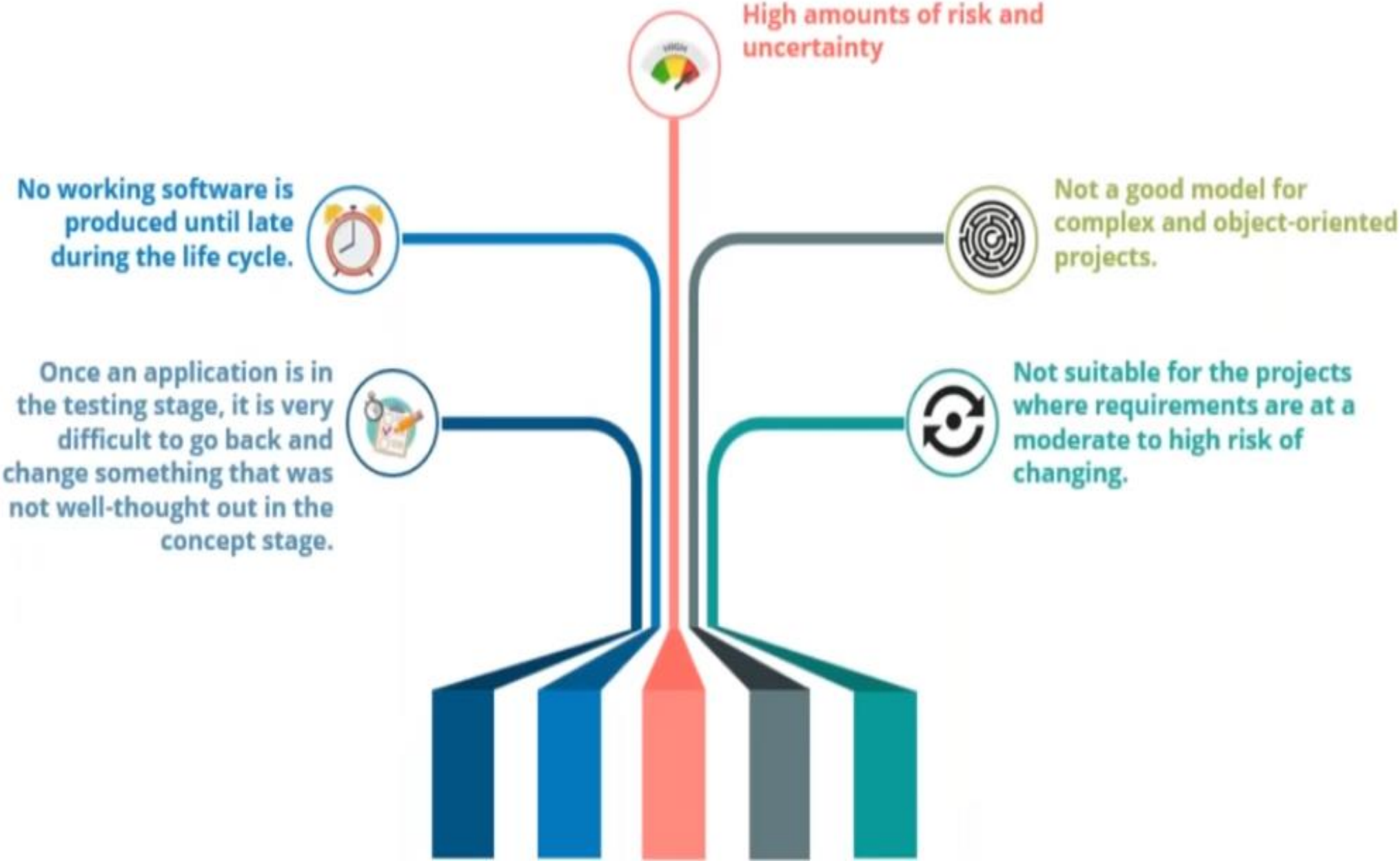

Ops who think like Devs
Devs who think like Ops

Primary factor leading to challenges during software development is the *Silo* between **development** & **operations**.



Evolution of software Development over the years

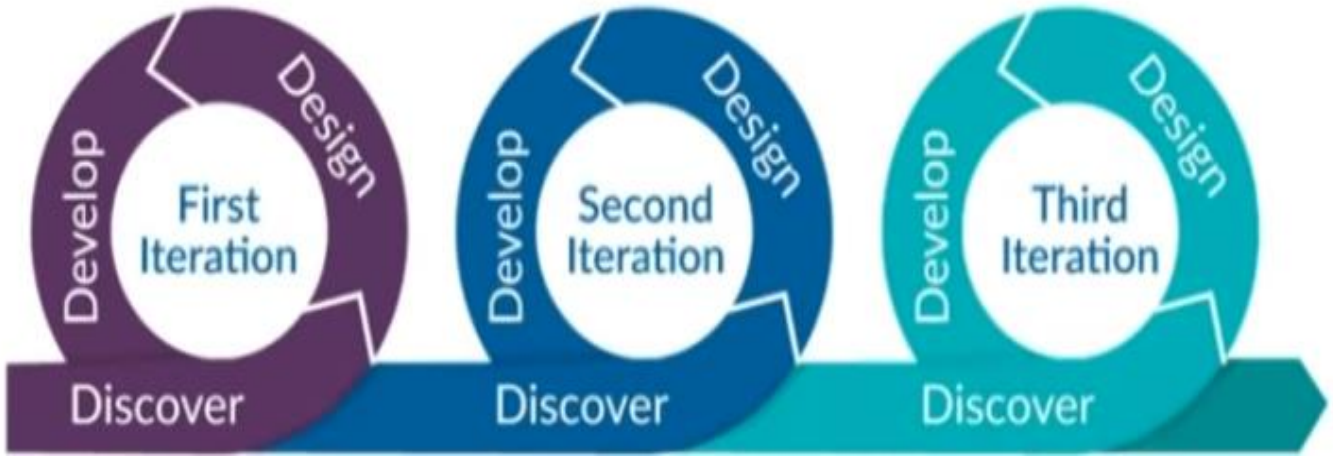



High amounts of risk and uncertainty
No working software is produced until late during the life cycle.
Not a good model for complex and object-oriented projects.
Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
Not suitable for the projects where requirements are at a moderate to high risk of changing.

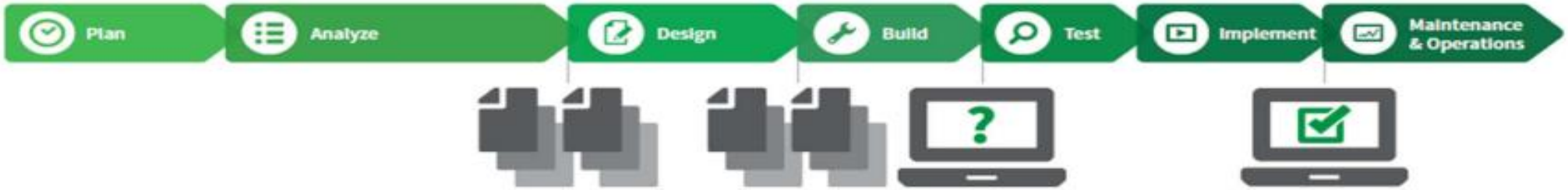
In the Agile Methodology each project is broken up into several 'Iterations'

All Iterations should be of the same time duration (between 2 to 8 weeks)

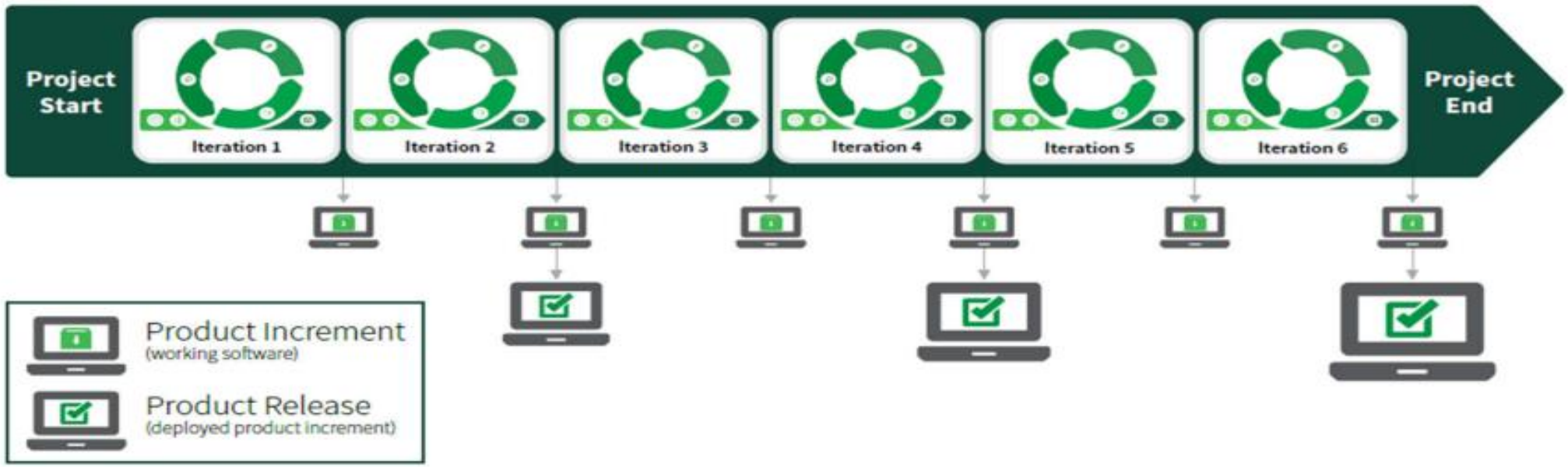
At the end of each iteration, a working product should be delivered



Traditional Waterfall (System Development Lifecycle)



Agile Iterations

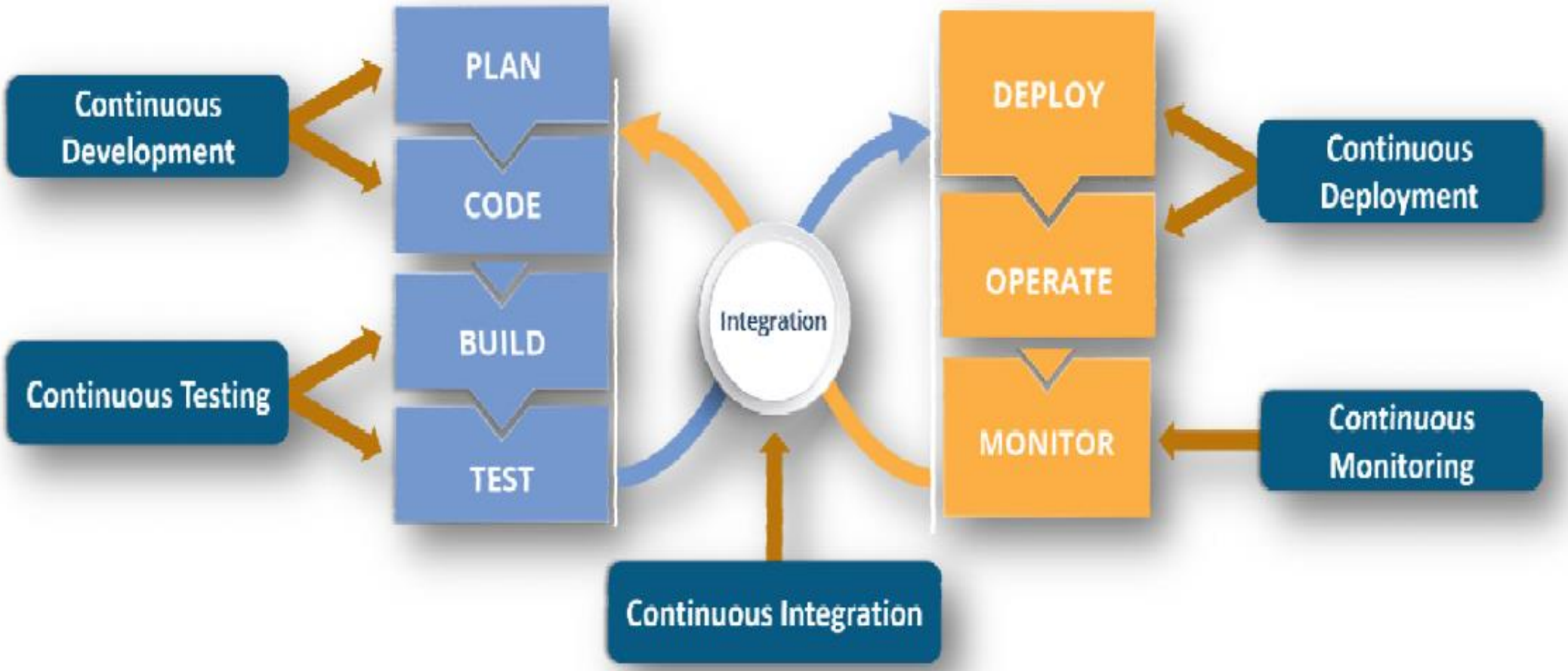


Waterfall

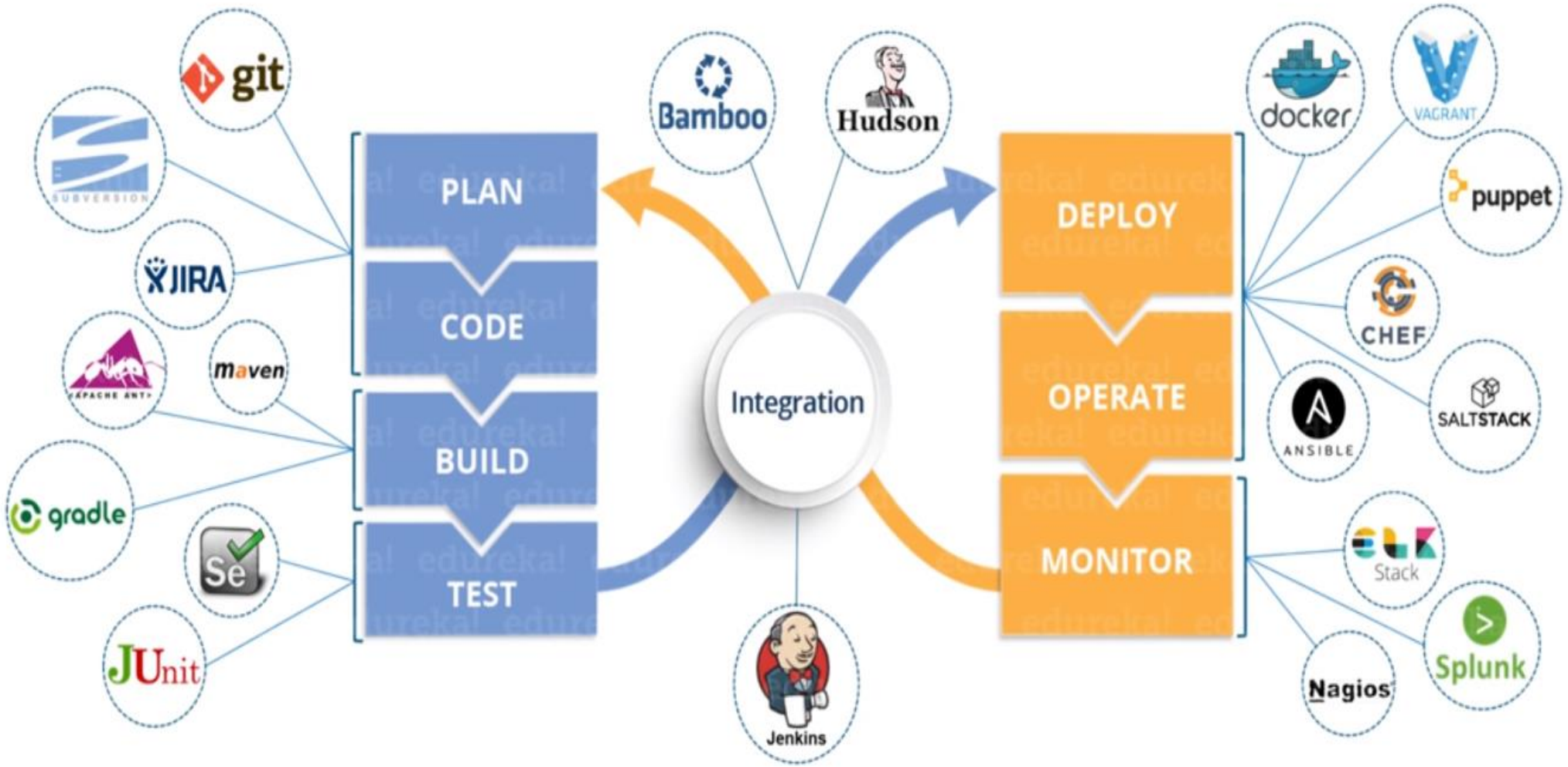
Agile

Agile with Continuous Deployment (Requires DevOps)

DevOps Practices



DevOps Phases and Tools



CONTINUOUS DEVELOPMENT

This is the phase that involves planning and coding , versioning and managing builds of the software application's functionality.

CONTINUOUS TESTING

Continuous testing is, executing automated tests, continuously and repeatedly against the code base and the various deployment environments. It is a software testing methodology which focuses on achieving continuous quality & improvement.

CONTINUOUS INTEGRATION

Continuous integration refers to the build and unit testing stages of the software release process. Every revision that is committed triggers an automated build and test.

Improve Developer Productivity

Find and Address Bugs Quicker

Deliver Updates Faster

Jenkins

Travis CI

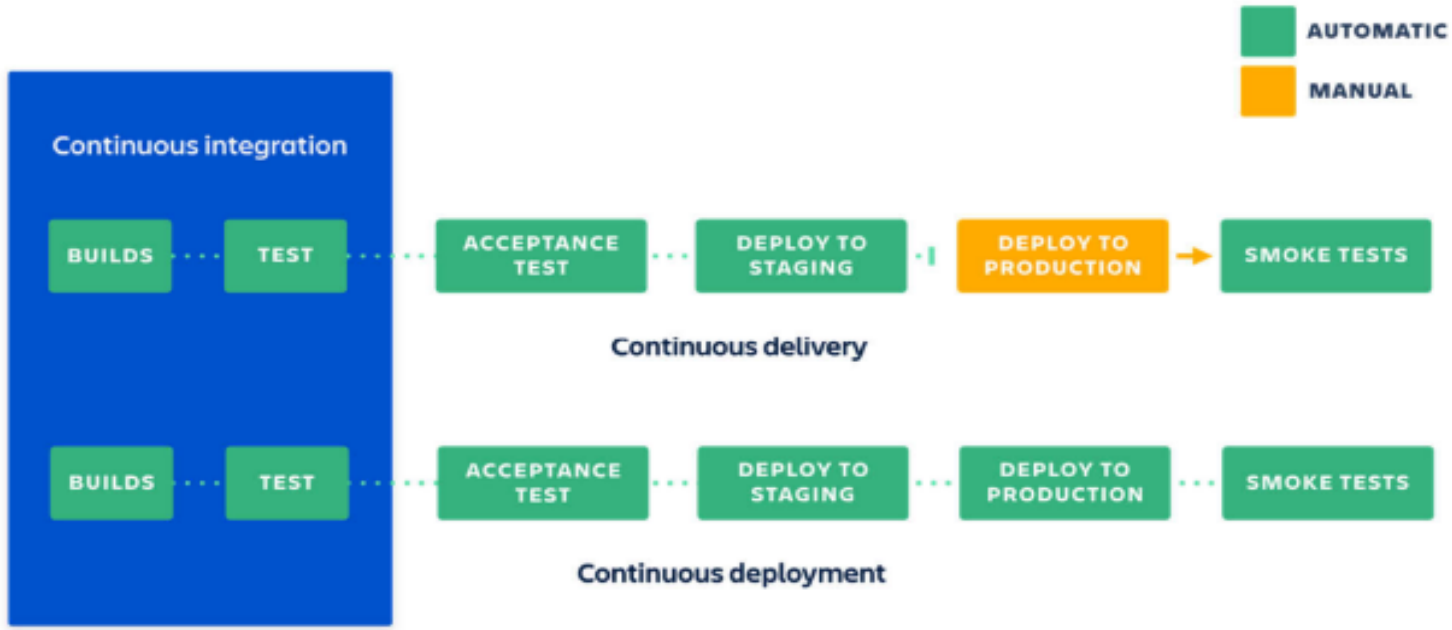
circleci

CONTINUOUS DELIVERY

Vs.

CONTINUOUS DEPLOYMENT

Continuous delivery and deployment originate from continuous integration, a method to develop, build and test new code rapidly with automation so that only code that is known to be good becomes part of a software product.



CONFIGURATION MANAGEMENT

Infrastructure as Code

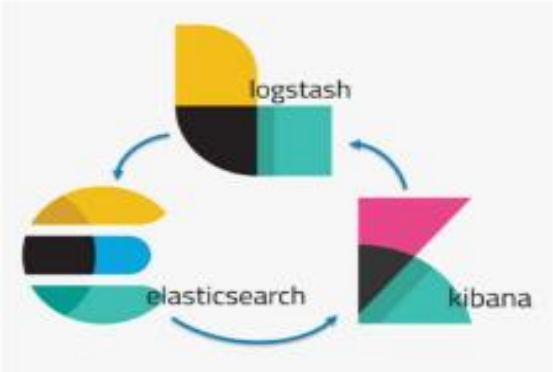
Infrastructure as Code is the practice of describing all software runtime environment and networking settings and parameters in simple textual format, that can be stored in your Version Control System (VCS) and versioned on request. These text files are called manifests and are used by DevOps tools to automatically provision and configure build servers, testing, staging and production environments.

CONTINUOUS MONITORING

Continuous monitoring refers to the process and technology required to incorporate monitoring across each phase of your DevOps and IT operations lifecycles. It helps to continuously ensure the health, performance, and reliability of your application and infrastructure as it moves from development to production.

Nagios®

splunk>

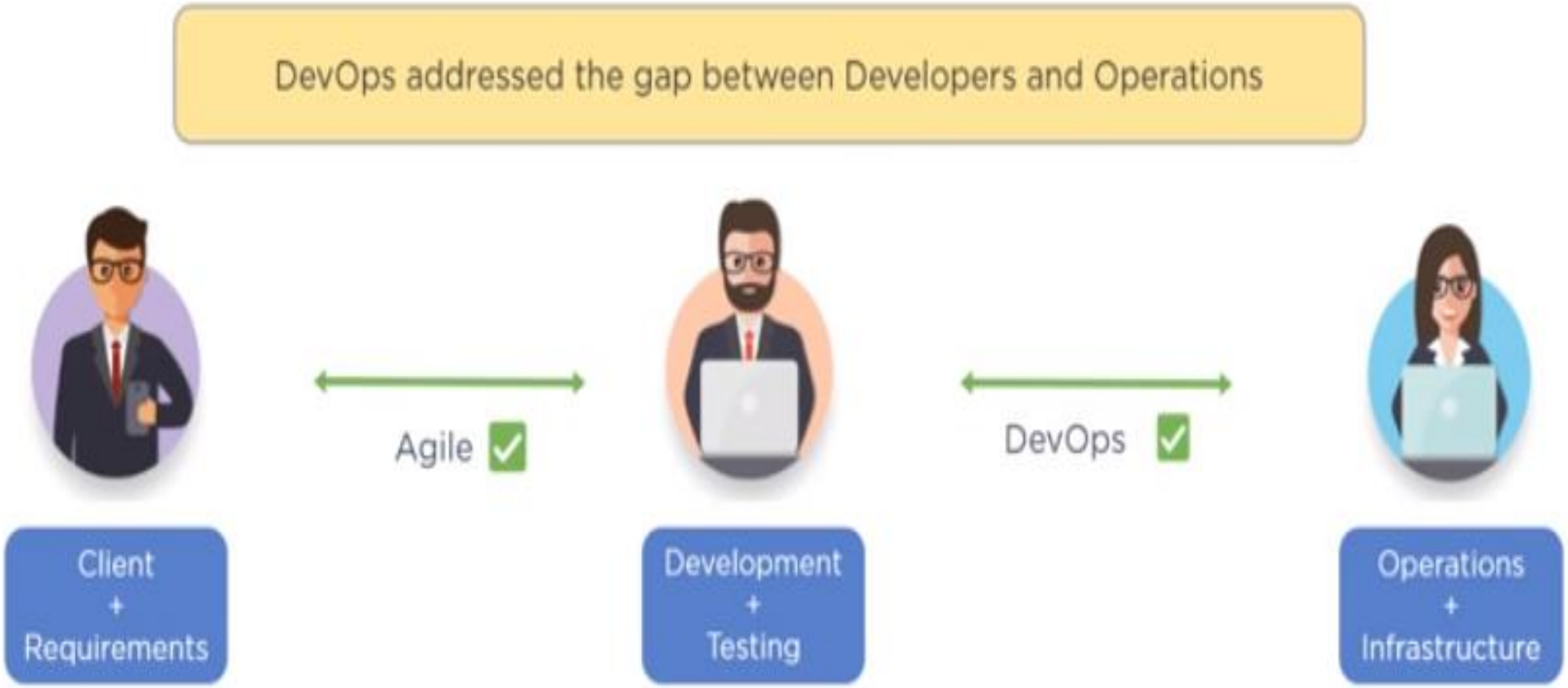


Conclusion DevOps

DevOps is an evolution from Agile model of software development

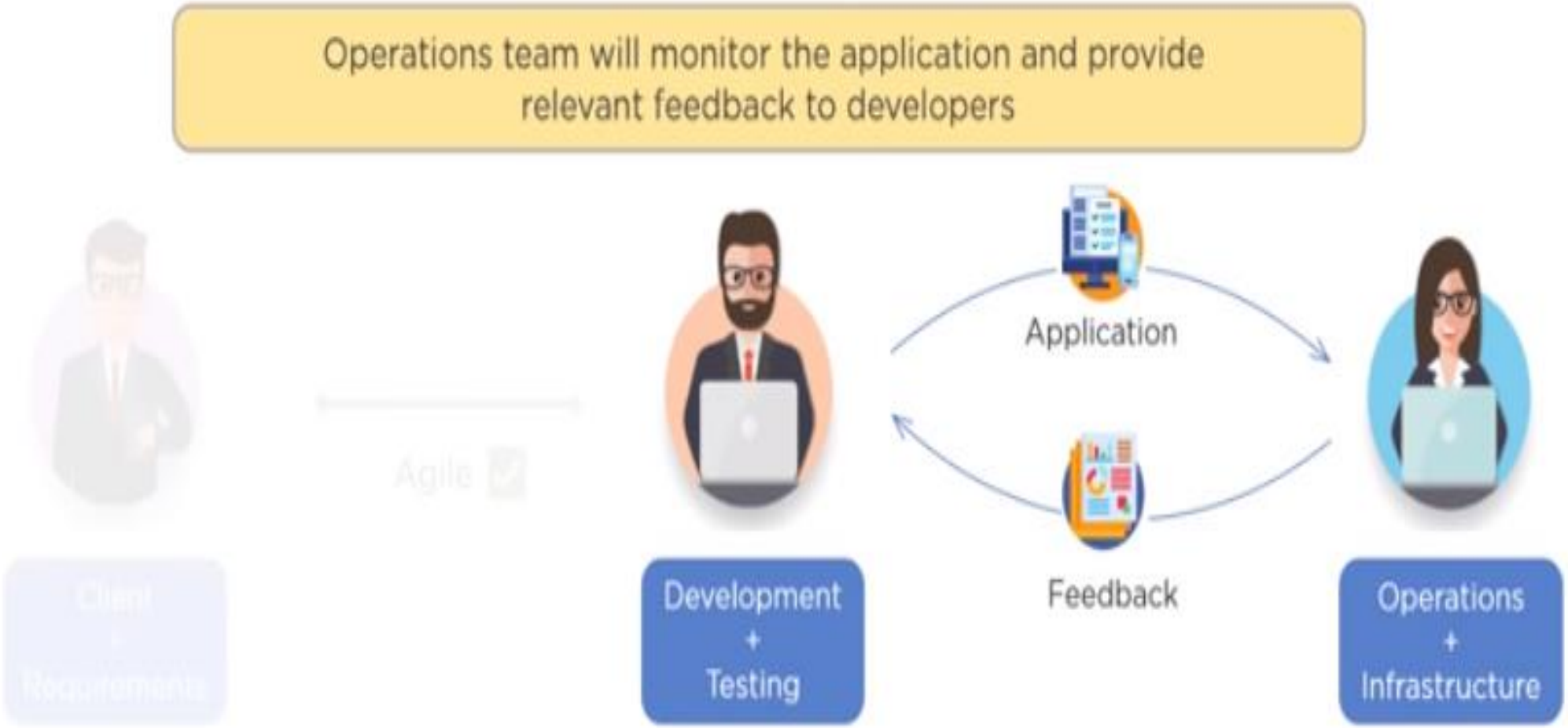
Agile addressed the gap between clients and developers

Conclusion DevOps



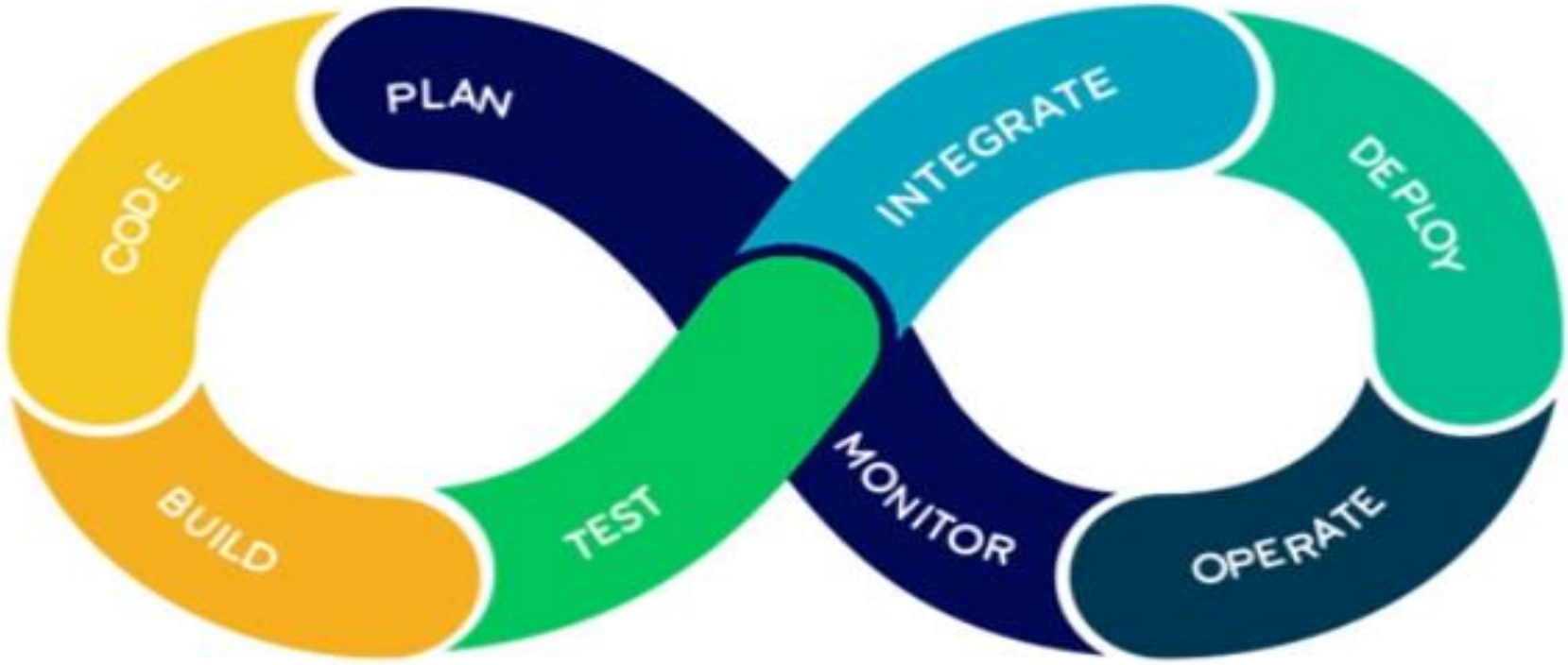
Conclusion DevOps

Conclusion DevOps



Conclusion DevOps Phases

According to DevOps practices, the workflow in software development and delivery is divided into 8 phases



Conclusion DevOps Phases

Plan

In plan stage, business owners and software development team discuss project goals and create a plan

Conclusion DevOps Phases

Programmers then design and code the application and use tools like Git to store application code

Conclusion DevOps Phases

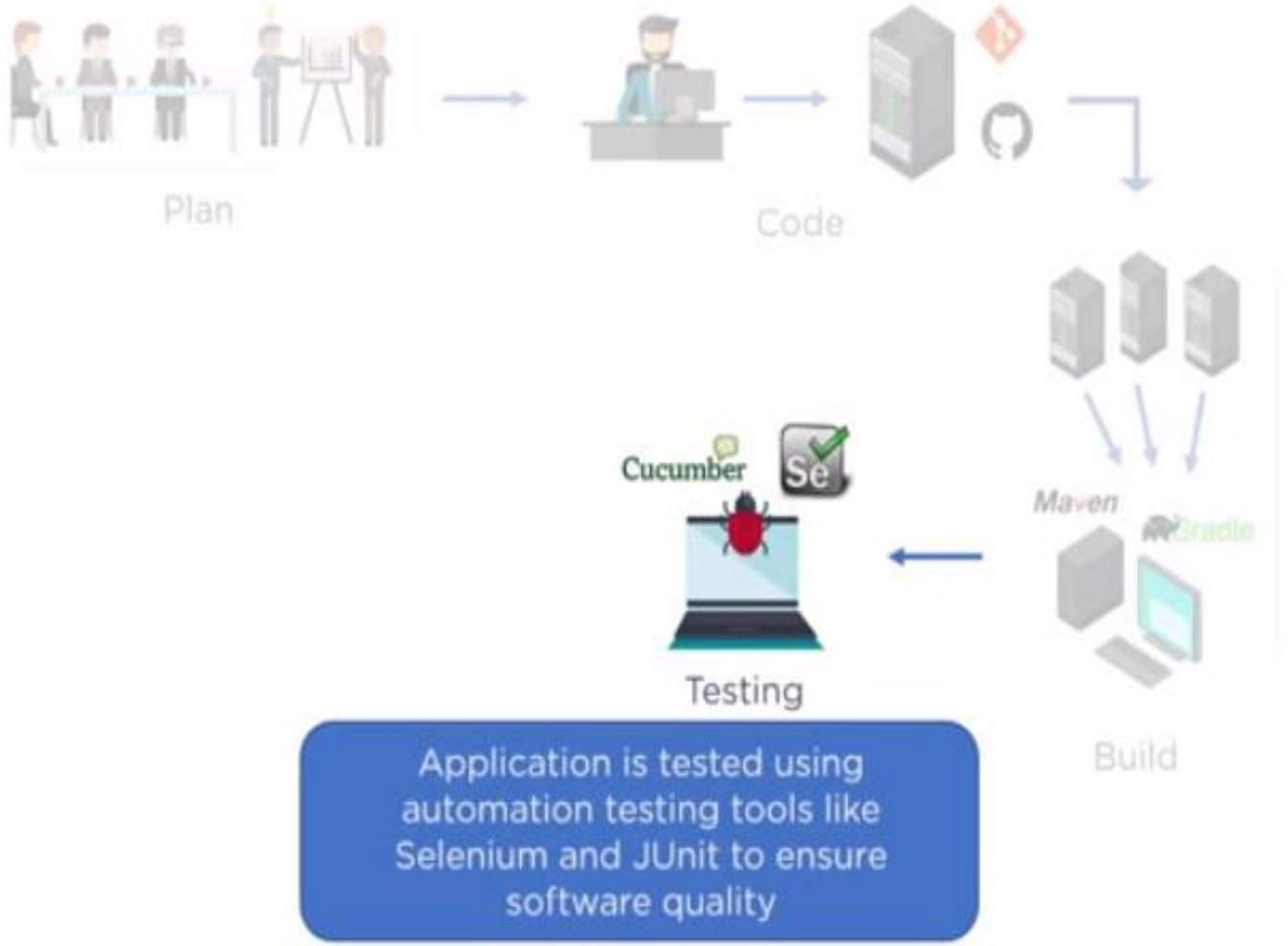
Plan

Code

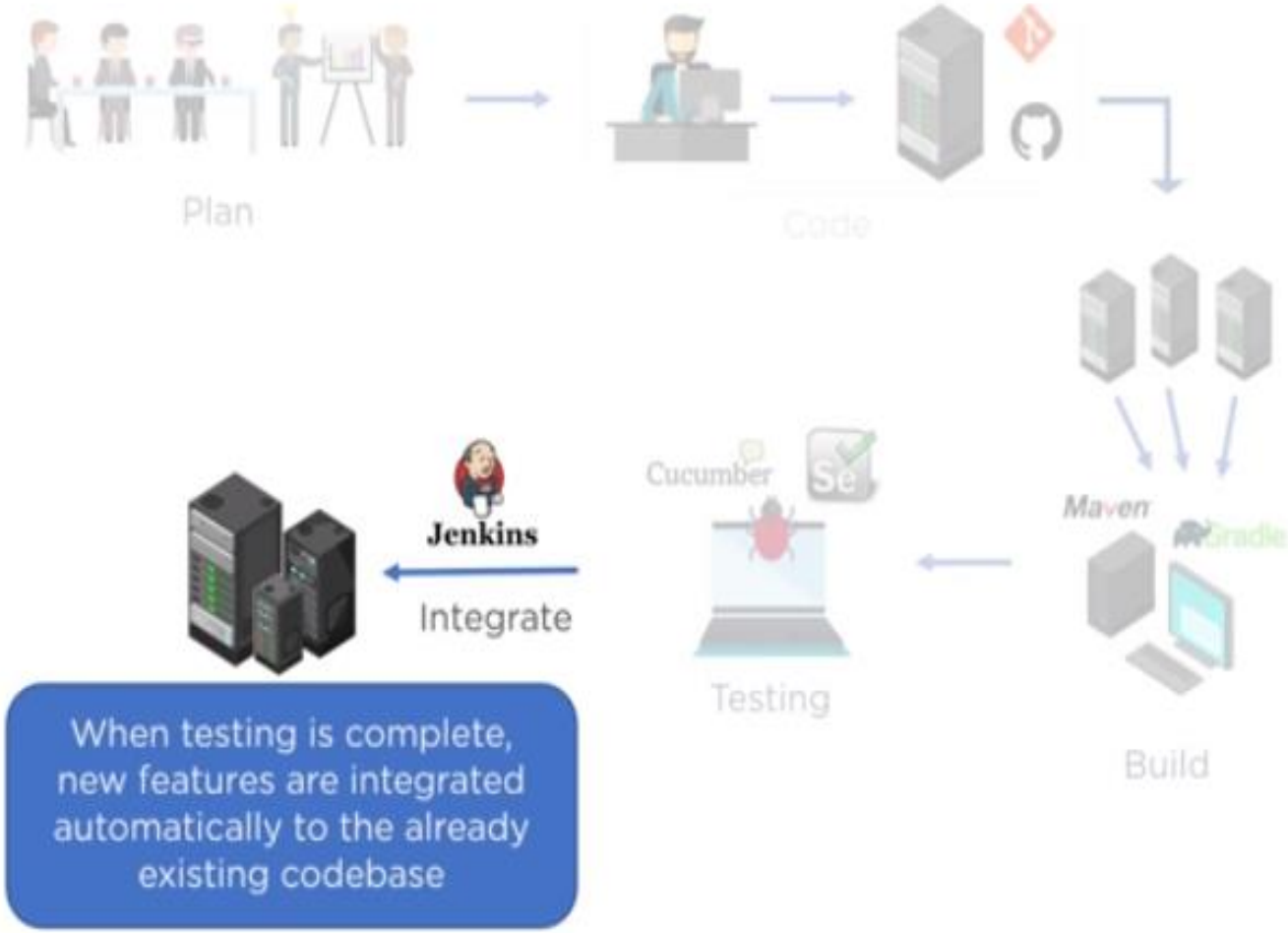
Build

Build tools like Maven and Gradle, take code from different repositories and combine them to build the complete application

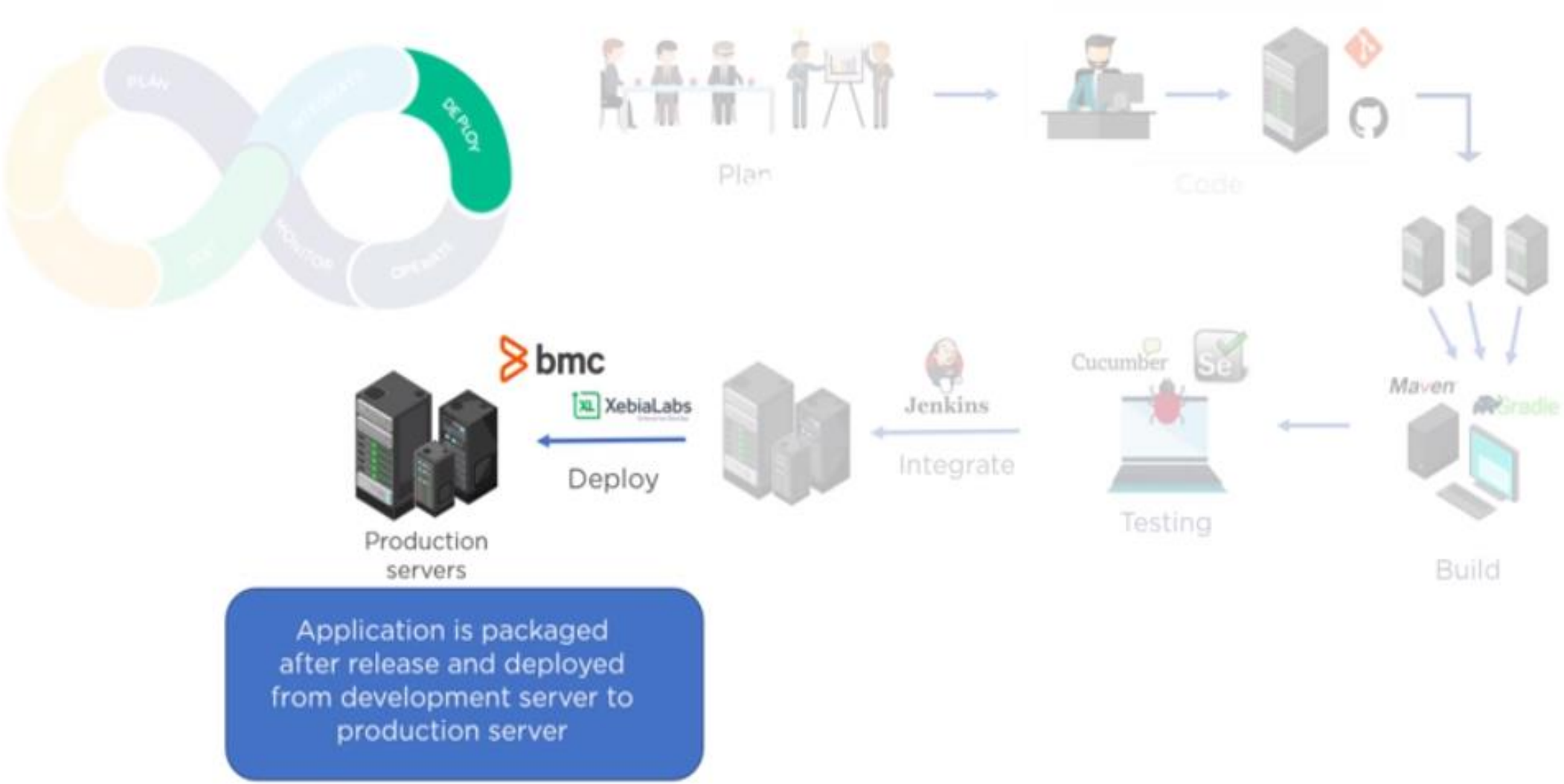
Conclusion DevOps Phases



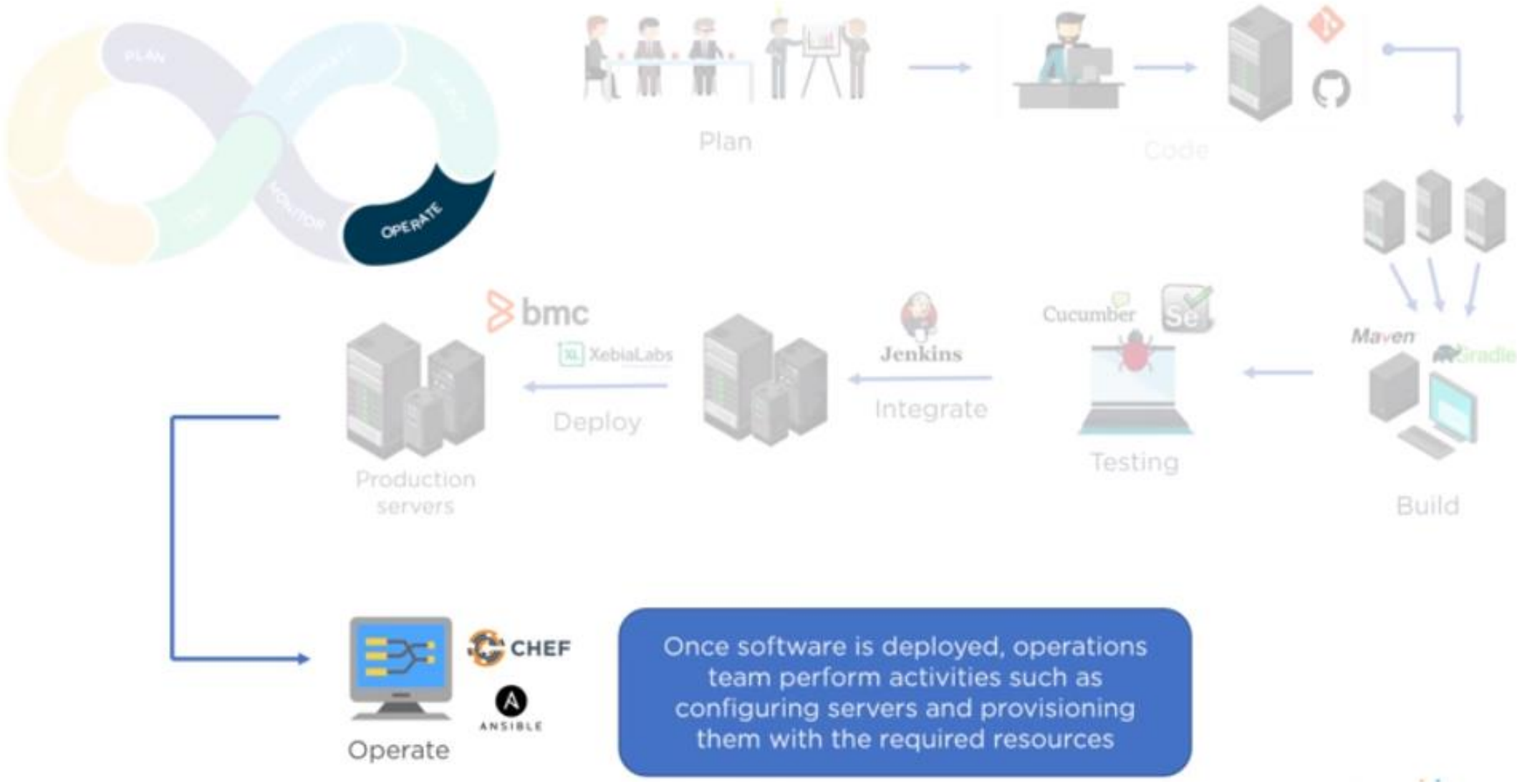
Conclusion DevOps Phases



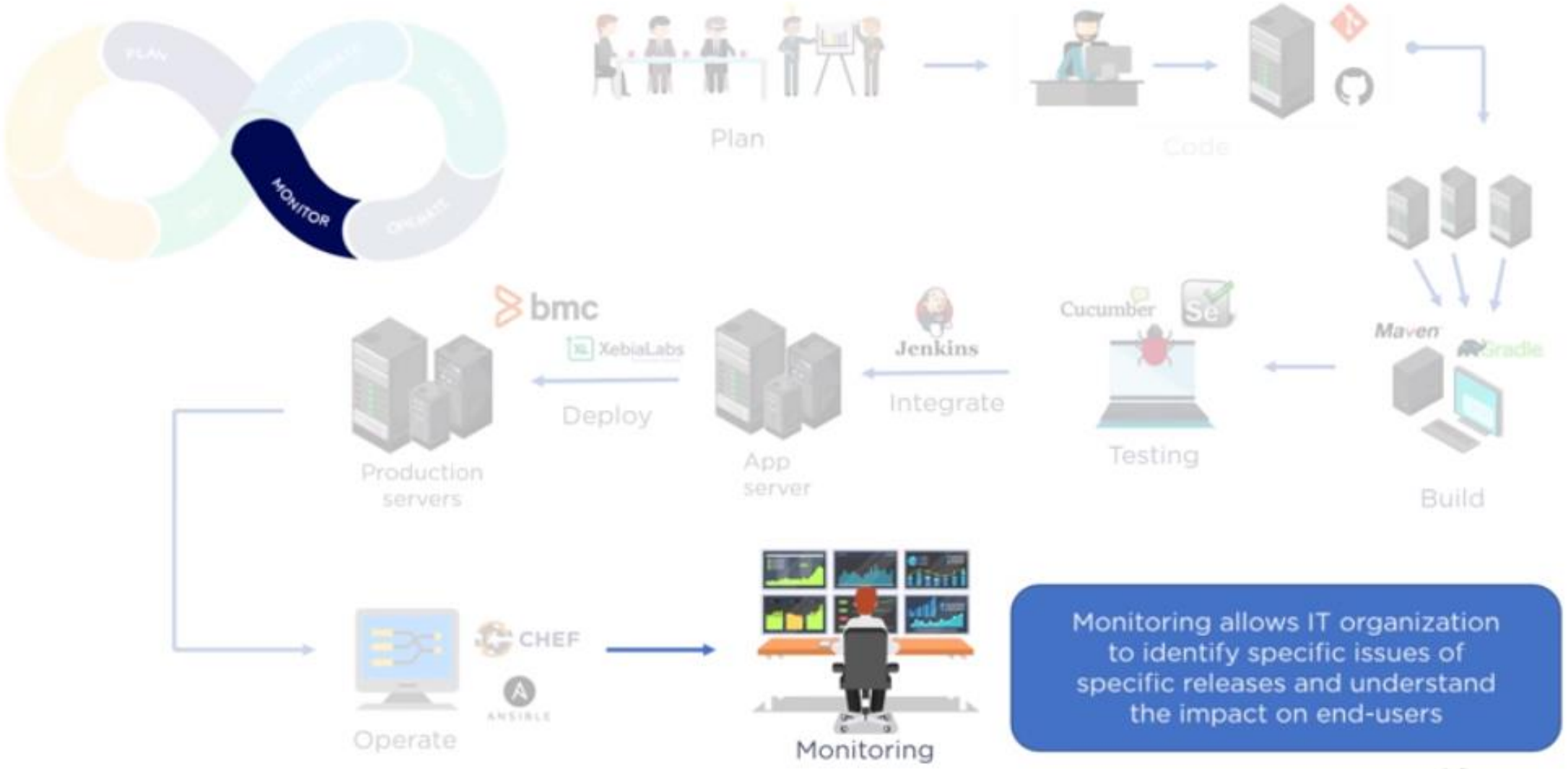
Conclusion DevOps Phases



Conclusion DevOps Phases



Conclusion DevOps Phases



Conclusion DevOps Advantages

Companies which follow DevOps, release more products and features within a short amount of time

Conclusion DevOps Advantages

Time taken to create and deliver software is reduced

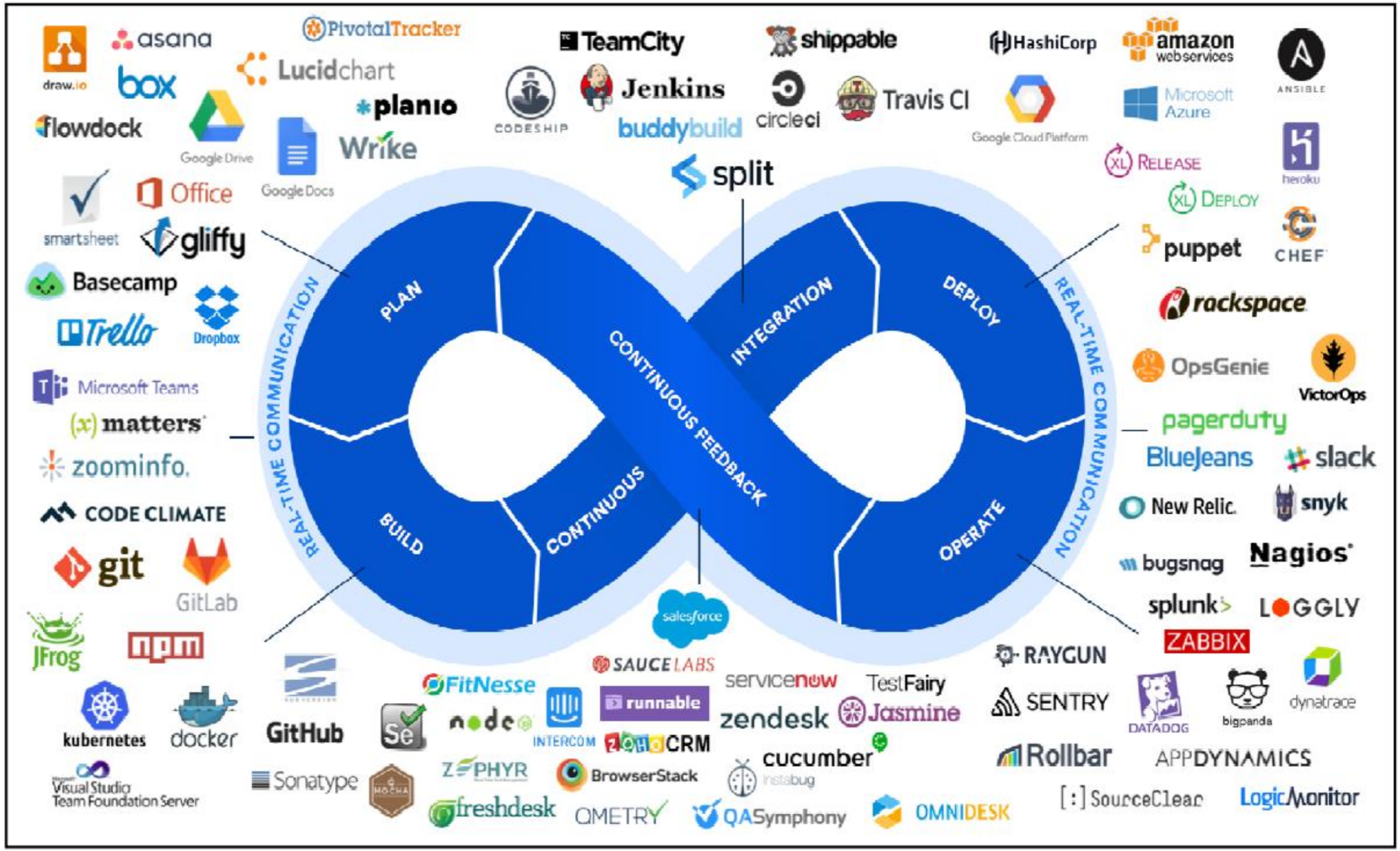
Complexity of maintaining an application is reduced

Improved collaboration between developers and operations team

Continuous integration and delivery ensure faster time to market


Jenkins
git
Se
ANSIBLE
splunk>
Nagios
docker
CHEF
elastic
puppet
DEVOPS TOOLS

Configuration Management	Continuous integration	Microservices	Collaboration	Monitoring	Development
CHEF	Jenkins	docker	JIRA Software	MONIT	Visual Studio
SALTSTACK	TeamCity	MESOS	slack	Ganglia	APACHE ACTIVEMQ
puppet	CodeShip	TRITON Compute	HipChat	SNORT	HashiCorp Vagrant
ANSIBLE	circleci	ElasticBox	Trello Organize anything, together.	cacti	Microsoft Azure


PLAN
CONTINUOUS
BUILD
CONTINUOUS FEEDBACK
INTEGRATION
DEPLOY
OPERATE
REAL-TIME COMMUNICATION
REAL-TIME COMMUNICATION
draw.io
asana
PivotalTracker
Lucidchart
box
planio
flowdock
Google Drive
Google Docs
Wrike
smartsheet
Office
gliffy
Basecamp
Trello
Dropbox
Microsoft Teams
(x) matters
zoominfo.
CODE CLIMATE
git
GitLab
JFrog
npm
kubernetes
docker
Visual Studio Team Foundation Server
GitHub
Sonatype
CODESHIP
TeamCity
Jenkins
buddybuild
shippable
circleci
Travis CI
split
FitNesse
node
runnable
INTERCOM
zoho CRM
ZEPHYR
BrowserStack
freshdesk
QMETRY
SAUCE LABS
servicenow
TestFairy
zendesk
Jasmine
cucumber
Instabug
QASymphony
OMNIDESK
salesforce
HashiCorp
amazon web services
Microsoft Azure
Google Cloud Platform
ANSIBLE
RELEASE
heroku
DEPLOY
puppet
CHEF
rackspace
OpsGenie
VictorOps
pagerduty
BlueJeans
slack
New Relic.
snyk
bugsnag
Nagios
splunk>
LOGGLY
ZABBIX
RAYGUN
SENTRY
DATADOG
bigpanda
dynatrace
Rollbar
APPDYNAMICS
[:] SourceClear
LogicMonitor

PERIODIC TABLE OF DEVOPS TOOLS (V3)

Os	Open Source
Fr	Free
Fm	Freemium
Pd	Paid
En	Enterprise

- Source Control Mgmt.
- Database Automation
- Continuous Integration
- Testing
- Configuration
- Deployment
- Containers
- Release Orchestration
- Cloud
- AIOps
- Analytics
- Monitoring
- Security
- Collaboration

1 Os Gl GitLab																		2 En Sp Splunk
3 Fm Gh GitHub	4 En Dt Datical											5 En XLr XebiaLabs XL Release	6 Fm Aws AWS	7 Pd Az Azure	8 En Gc Google Cloud	9 Fm Op OpenShift	10 Fm Sg Sumo Logic	
11 Os Sv Subversion	12 En Db DBMaestro											13 Os Dk Docker	14 En Ur UrbanCode Release	15 Pd Af Azure Functions	16 Pd Ld Lambda	17 Fm Ic IBM Cloud	18 Os Fd Fluentd	
19 En Cw ISPW	20 En Dp Delphix	21 Os Jn Jenkins	22 Fm Cs Codeship	23 Os Fn FitNesse	24 Fr Ju JUnit	25 Fr Ka Karma	26 Fm Su SoapUI	27 En Ch Chef	28 Fr Tf Terraform	29 En XLd XebiaLabs XL Deploy	30 En Ud UrbanCode Deploy	31 Os Ku Kubernetes	32 Fm Cc CA CD Director	33 En Pr Plutora Release	34 Pd Al Alibaba Cloud	35 Os Os OpenStack	36 Os Ps Prometheus	
37 Pd At Artifactory	38 Fm Rg Redgate	39 Pd Ba Bamboo	40 Fm Vs VSTS	41 Fr Se Selenium	42 Fr Jm JMeter	43 Os Ja Jasmine	44 Pd Sl Sauce Labs	45 En An Ansible	46 Os Ru Rudder	47 En Oc Octopus Deploy	48 Os Go GoCD	49 Os Ms Mesos	50 Pd Gke GKE	51 Fm Om OpenMake	52 Pd Cp AWS CodePipeline	53 Pd Cy Cloud Foundry	54 En It ITRS	
55 Pd Nx Nexus	56 Os Fw Flyway	57 Os Tr Travis CI	58 Fm Tc TeamCity	59 Os Ga Gatling	60 Fr Tn TestNG	61 Fm Tt Tricentis Tosca	62 Pd Pe Perfecto	63 En Pu Puppet	64 Os Pa Packer	65 Fm Cd AWS CodeDeploy	66 En Ec ElectricCloud	67 Os Ra Rancher	68 Pd Aks AKS	69 Os Rk Rkt	70 Os Sp Spinnaker	71 Pd Ir Iron.io	72 Pd Mg Moogsoft	
73 Fm Bb BitBucket	74 En Pf Perforce	75 Fm Cr Circle CI	76 Pd Cb AWS CodeBuild	77 Fr Cu Cucumber	78 Os Mc Mocha	79 Os Lo Locust.io	80 En Mf Micro Focus UFT	81 Os Sa Salt	82 Os Ce CFEngine	83 En Eb ElasticBox	84 En Ca CA Automic	85 En De Docker Enterprise	86 Pd Ae AWS ECS	87 Fm Cf Codefresh	88 Os Hm Helm	89 Os Aw Apache OpenWhisk	90 Os Ls Logstash	

XebiaLabs
Enterprise DevOps

Follow @xebialabs

91 En XLi XebiaLabs XL Impact	92 Os Ki Kibana	93 Fm Nr New Relic	94 En Dt Dynatrace	95 En Dd Datadog	96 Fm Ad AppDynamics	97 Os El ElasticSearch	98 Os Ni Nagios	99 Os Zb Zabbix	100 En Zn Zenoss	101 En Cx Checkmarx SAST	102 En Sg Signal Sciences	103 En Bd BlackDuck	104 Os Sr SonarQube	105 Os Hv HashiCorp Vault				
106 En Sw ServiceNow	107 Pd Jr Jira	108 Fm Tl Trello	109 Fm Sk Slack	110 Fm St Stride	111 En Cn CollabNet VersionOne	112 En Ry Remedy	113 En Ac Agile Central	114 Pd Og OpsGenie	115 Pd Pd Pagerduty	116 Os Sn Snort	117 Fm Tw Tripwire	118 En Ck CyberArk	119 En Vc Veracode	120 En Ff Fortify SCA				


STEPS TO BECOME A DEVOPS ENGINEER
CONGRATULATIONS
YOU ARE A
DEVOPS ENGINEER
Linux Fundamentals and Scripting Knowledge
Source Code Management Skills (Version Control System)
Learn How To Take Continuous Integration To The Next Step (Continuous Delivery) For Continuous Testing
Maven
Learn How To Integrate Build Tools And Source Code Management Tools For Continuous Build/ Integration (CI)
Learn How To Deploy And Configure Dev, Test And Prod Environment: Puppet, Ansible, Docker Kubernetes Logo
Learn How You Can Use Monitoring Tools To Collect Useful Feedback
Nagios
splunk>
Understand How Cloud Service Platforms Work


Thank you

End of Module 1

Any Query??