Add instructor notes here.

Add instructor notes here.

Lesson Objectives

- Introduction to DevOps
  - What is DevOps
  - Evolution of DevOps
  - Agile Methodology
  - Why DevOps
  - Agile vs DevOps
  - DevOps Principles
  - DevOps Lifecycle
  - DevOps Tools
  - Benefits of DevOps
  - Continuous Integration and Delivery pipeline
  - Use-case walkthrough

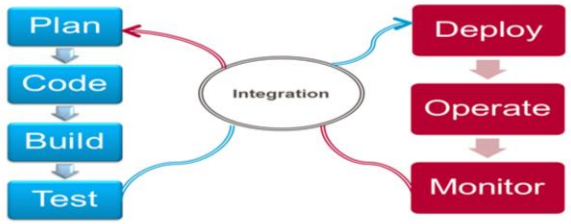
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

1.1: What is DevOps

What is DevOps

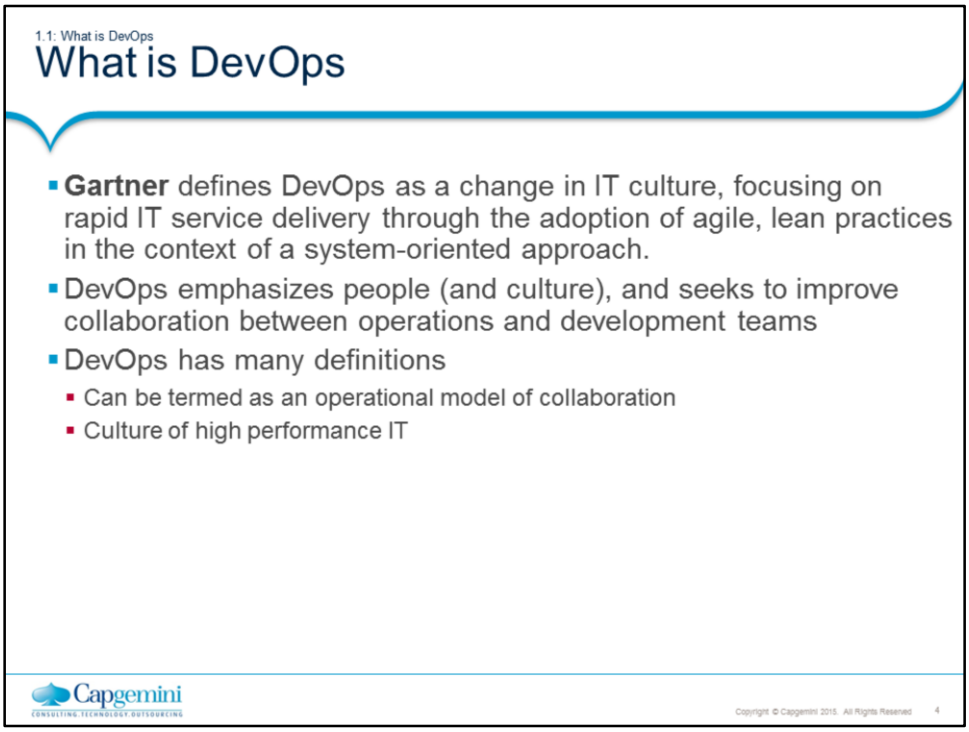
- DevOps is a term used to refer to a set of practices that emphasize the collaboration and communication of both software developers and information technology (IT) professionals while automating the process of software delivery and infrastructure changes.

It aims at establishing a culture and environment where building, testing, and releasing software can happen rapidly, frequently, and more reliably



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

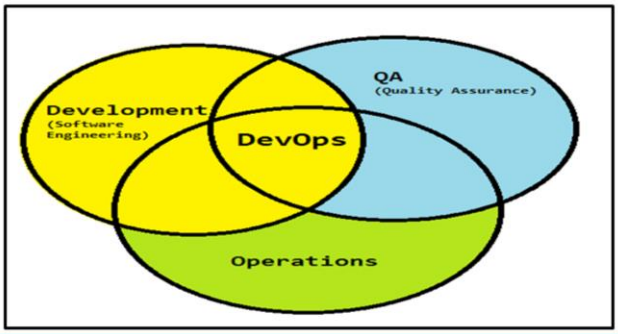
 3


1.1: What is DevOps
What is DevOps
Gartner defines DevOps as a change in IT culture, focusing on rapid IT service delivery through the adoption of agile, lean practices in the context of a system-oriented approach.
DevOps emphasizes people (and culture), and seeks to improve collaboration between operations and development teams
DevOps has many definitions
Can be termed as an operational model of collaboration
Culture of high performance IT
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
4

1.1: What is DevOps

What is DevOps

- DevOps integrates developers and operations team in order to improve collaboration and productivity by:
  - Automating infrastructure
  - Automating workflows
  - Continuous measuring application performance



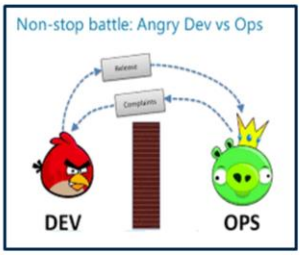
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

1.1: What is DevOps

Dev and Ops

- DevOps can be explained simply as operations working together with engineers to get things done faster in an automated and repeatable way.
- **Growing pains of operation**

  When we are responsible for large distributed applications the operations complexity grows quickly.
  - How do you provision virtual machines?
  - How do you configure network devices and servers?
  - How do you deploy applications?
  - How do you collect and aggregate logs?
  - How do you monitor services?
  - How do you monitor network performance?
  - How do you monitor application performance?
  - How do you alert and remediate when there are problems?



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 6

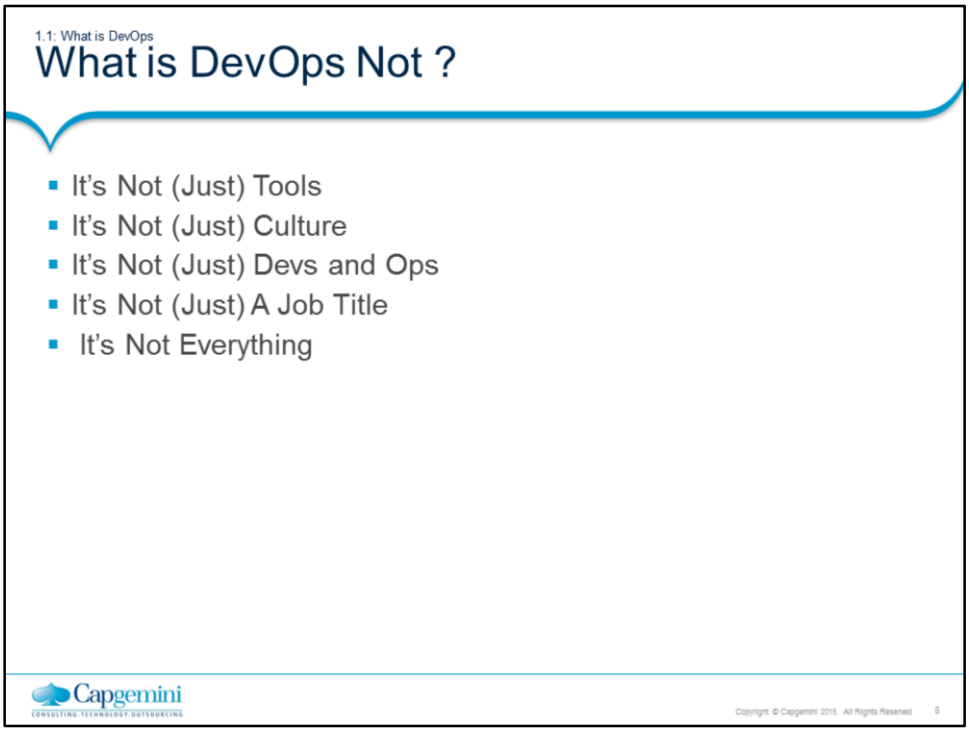
1.1: What is DevOps

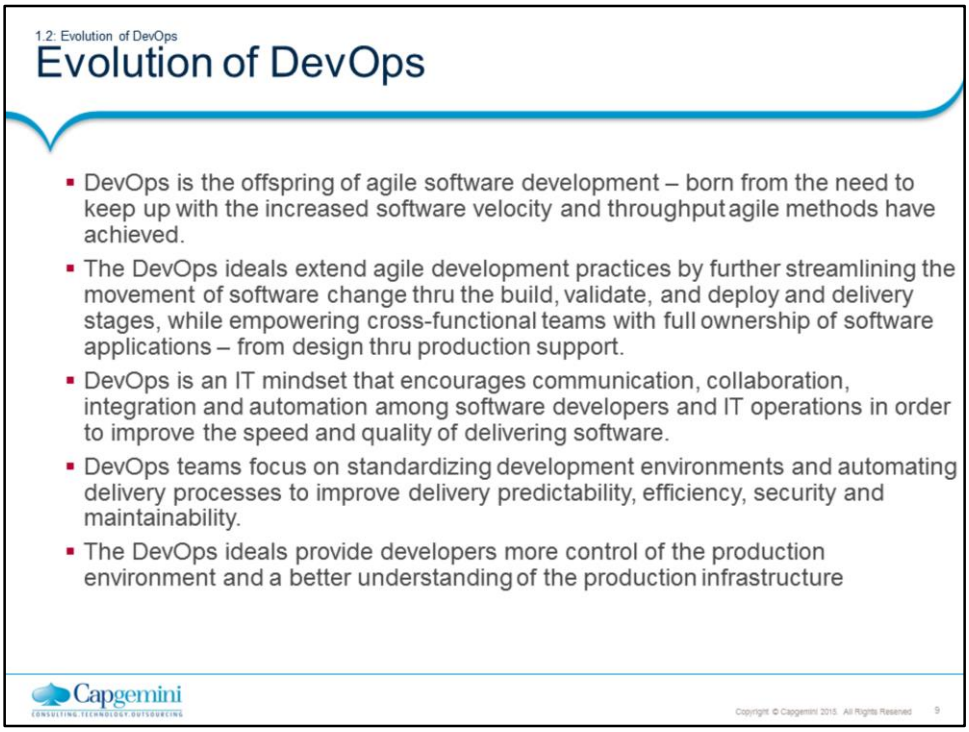
Dev and Ops

- **Growing pain of developer**
  - Dev is often unaware of Ops roadblocks that prevent the program from working as anticipated
- **Combining the power of developers and operations**

  - The focus on the developer/operations collaboration enables a new approach to managing the complexity of real world operations.
  - The operations complexity breaks down into a few main categories: infrastructure automation, configuration management, deployment automation, log management, performance management, and monitoring.

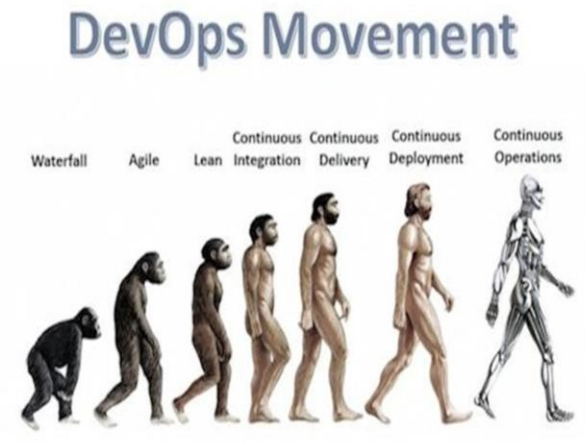
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING


1.1: What is DevOps
What is DevOps Not ?
It's Not (Just) Tools
It's Not (Just) Culture
It's Not (Just) Devs and Ops
It's Not (Just) A Job Title
It's Not Everything
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
8


1.2: Evolution of DevOps
Evolution of DevOps
DevOps is the offspring of agile software development – born from the need to keep up with the increased software velocity and throughput agile methods have achieved.
The DevOps ideals extend agile development practices by further streamlining the movement of software change thru the build, validate, and deploy and delivery stages, while empowering cross-functional teams with full ownership of software applications – from design thru production support.
DevOps is an IT mindset that encourages communication, collaboration, integration and automation among software developers and IT operations in order to improve the speed and quality of delivering software.
DevOps teams focus on standardizing development environments and automating delivery processes to improve delivery predictability, efficiency, security and maintainability.
The DevOps ideals provide developers more control of the production environment and a better understanding of the production infrastructure
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
9

1.2: Evolution of DevOps

DevOps Movement



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

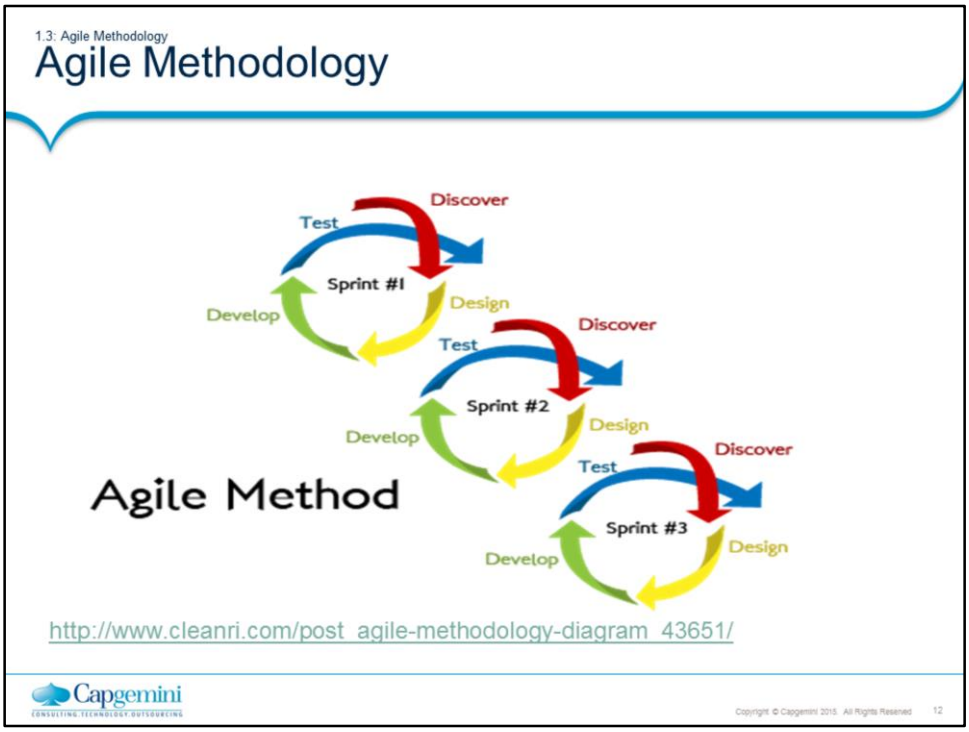
 10

1.3: Agile Methodology

Agile Methodology

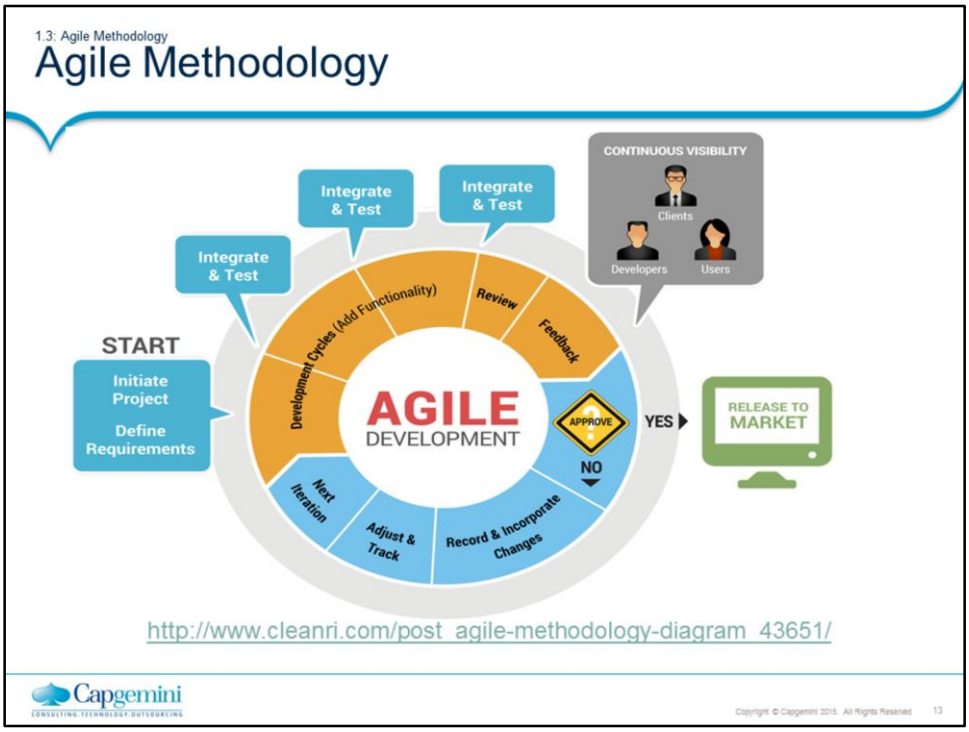
- **Agile Methodology**: In the Agile approach, software is developed and released incrementally in the iterations
- This results in small incremental releases with each release building on previous functionality
- Each release is thoroughly **tested** to ensure **software quality** is maintained
- It is used for time critical applications

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING



In the **Agile methodology**, each project is broken up into several 'Iterations'.

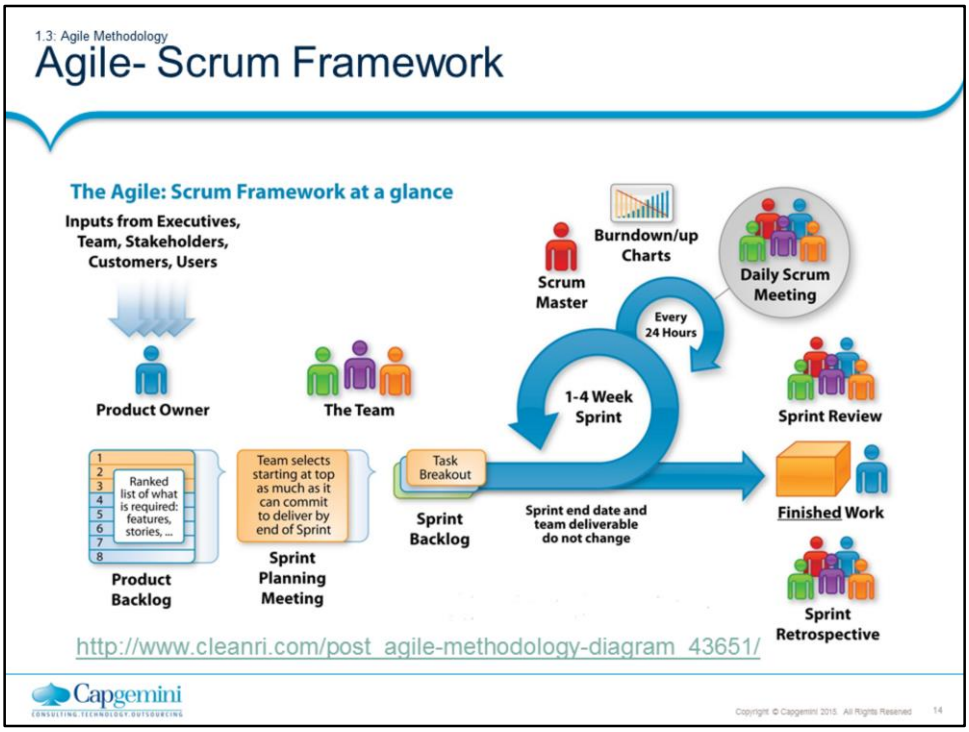
- All Iterations should be of the same time duration (between 2 to 8 weeks).
- At the end of each iteration, a working product should be delivered.
- **Sprint**- the agile(scrum) term for iteration
- Within each sprint, the development team builds and tests a functional part of the product until the product owner accepts it and the functionality becomes a potentially shippable product. When one sprint finishes, another sprint starts.
- Agile teams deliver product features in increments at the end of each sprint.
- A product release occurs at the end of a sprint or after several sprints.
- A core principle of the sprint is its cyclical nature: **The sprint, as well as the processes within it, repeats over and over, as shown:**



- This approach allows the customer to interact and work with functioning software at the end of each iteration and provide feedback on it.
- This approach allows teams to take up changes more easily and make course corrections if needed.
- In the Agile approach, software is developed and released incrementally in the iterations.
- Agile methodology gives more importance to collaboration within the team, collaboration with the customer, responding to change and delivering working software.

Advantages of Agile model:

- Customer satisfaction by rapid, continuous delivery of useful software.
- People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
- Working software is delivered frequently (weeks rather than months).
- Face-to-face conversation is the best form of communication.
- Close, daily cooperation between business people and developers.
- Continuous attention to technical excellence and good design.
- Regular adaptation to changing circumstances.
- Even late changes in requirements are welcomed



Scrum is an efficient framework within which you can develop software with teamwork. It is based on agile principles.

Scrum Team : is a team comprising of 7 with + or - two members. These members are a mixture of competencies and comprise of developers, testers, data base people, support people etc. along with the product owner and a scrum master

Sprint: the scrum term for iteration.

Product Owner: is the key stakeholder or the lead user of the application to be developed.

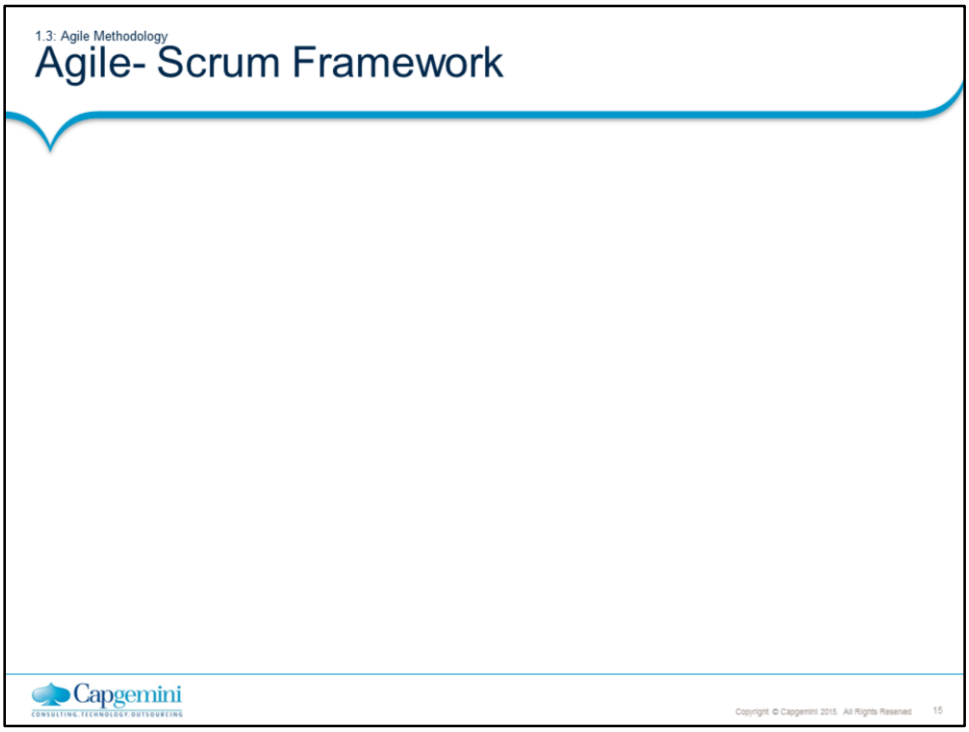
Scrum Master: is the facilitator of the scrum team. He / she make sure that the scrum team is productive and progressive.

User Story: User stories are nothing but the requirements or feature which has to be implemented. In scrum, we don't have those huge requirements documents, rather the requirements are defined in a single paragraph

Product Backlog: Product backlog is a kind of bucket: or source where all the user stories are kept. This is maintained by Product owner. Product backlog can be imagined as a wish list of the product owner who prioritizes it as per business needs.

Sprint Backlog: Based on the priority, user stories are taken from the Product Backlog one at a time. The Scrum team brainstorms on it, determines the feasibility and decides on the stories to work on a particular sprint. The collective list of all the user stories which the scrum team works on a particular sprint is called s Sprint backlog.

Story Points: Story points are quantitative indication of the complexity of a user story. Based on the story point, estimation and efforts for a story is determined. Story point is relative and is not absolute.



THE FUNCTION OF THE SCRUM AND SPRINT WITHIN AN AGILE PROJECT

- *Scrum*, the most popular agile framework in software development, is an iterative approach that has at its core the *sprint* — the scrum term for iteration.
- Scrum teams use inspection throughout an agile project to ensure that the team meets the goals of each part of the process.
- The scrum approach includes assembling the project's requirements and using them to define the project.
- You then plan the necessary sprints, and divide each sprint into its own list of requirements.
- Daily scrum meetings help keep the project on target as do regular inspections and reviews.
- At the end of every sprint, you hold a sprint retrospective to look for ways to improve the next sprint.
- The process looks as shown in the diagram:
- Within each sprint, the development team builds and tests a functional part of the product until the product owner accepts it and the functionality becomes a potentially shippable product.
- When one sprint finishes, another sprint starts.
- Scrum teams deliver product features in increments at the end of each sprint.
- A product release occurs at the end of a sprint or after several sprints.

1.3: Agile Methodology

Agile Practices

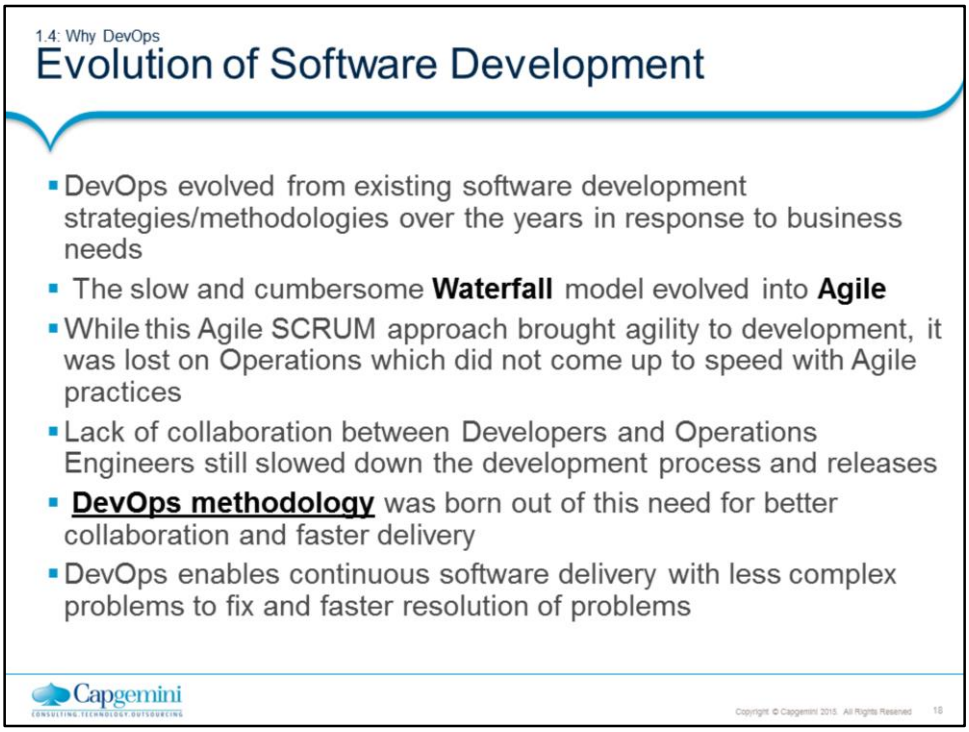
- **Development** is Agile, what about Agility in **Operations**?

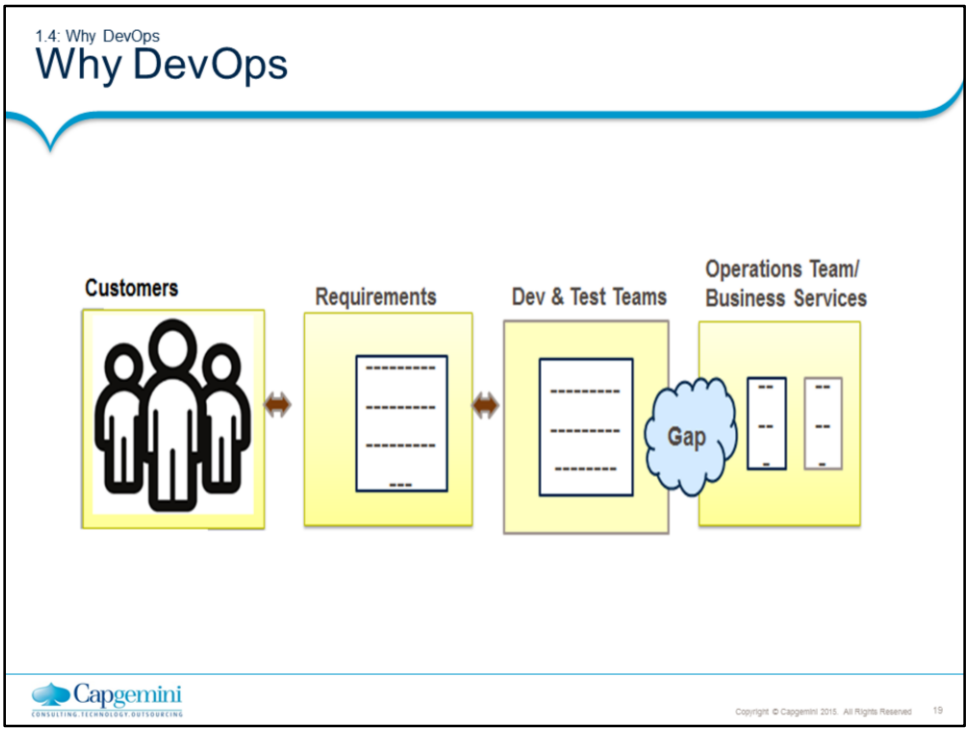
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

1.3: Agile Methodology

Agile Practices

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING


1.4: Why DevOps
Evolution of Software Development
DevOps evolved from existing software development strategies/methodologies over the years in response to business needs
The slow and cumbersome Waterfall model evolved into Agile
While this Agile SCRUM approach brought agility to development, it was lost on Operations which did not come up to speed with Agile practices
Lack of collaboration between Developers and Operations Engineers still slowed down the development process and releases
DevOps methodology was born out of this need for better collaboration and faster delivery
DevOps enables continuous software delivery with less complex problems to fix and faster resolution of problems
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
18


1.4: Why DevOps
Why DevOps
Customers
Requirements
Dev & Test Teams
Operations Team/
Business Services
Gap
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
19

1.4: Why DevOps

Why DevOps

- DevOps takes care of the challenges faced by Development and Operations
- Below table describes how DevOps addresses Dev Challenges

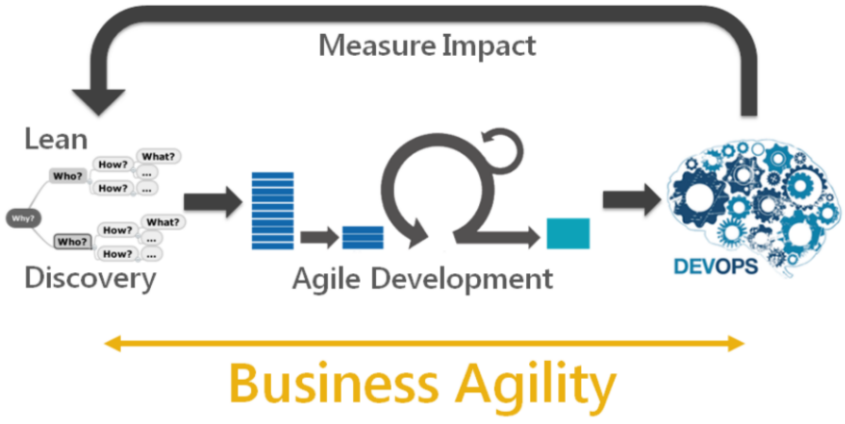
	Dev Challenges	DevOps Solution
	Waiting time for code deployment	• Continuous Integration: Ensures there is a quick deployment of code, faster testing and speedy feedback mechanism. • No waiting time to deploy the code. Hence developers focuses on building the current code.
	Pressure of work on old, pending and new code	
	Tools to automate infrastructure management are not effective	Configuration Management: Helps to organize and execute configuration plans and consistently provision the system
	No. of servers to be monitored increases	Continuous Monitoring: Effective monitoring and feedbacks system is established through Nagios. Thus effective administration is achieved
	Difficult to diagnose and provide feedback on the product	

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Accelerate software delivery Reduce time to customer feedback Balance speed, cost, quality and risk

1.4: Why DevOps

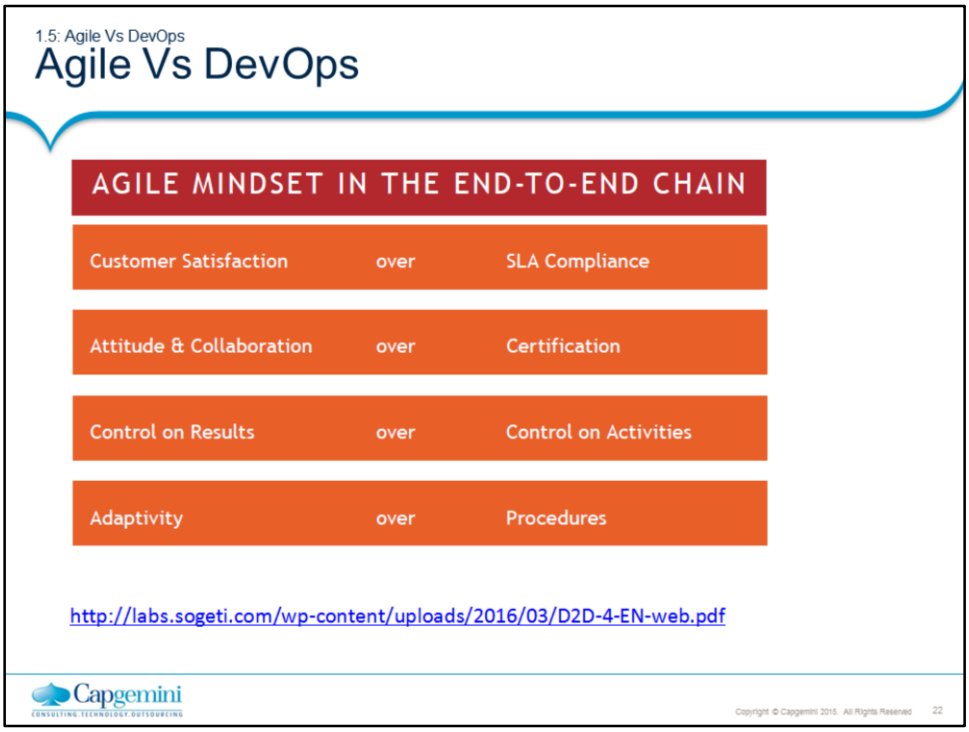
Emergence of DevOps: Influence on DevOps



https://theagileadmin.com/what-is-devops/

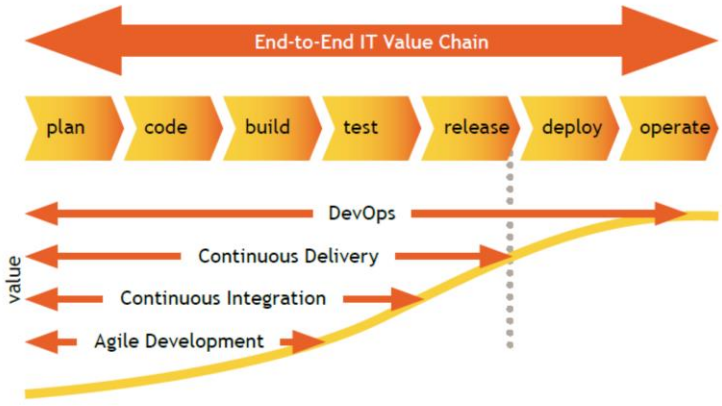
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 21


1.5: Agile Vs DevOps
Agile Vs DevOps
AGILE MINDSET IN THE END-TO-END CHAIN
Customer Satisfaction over SLA Compliance
Attitude & Collaboration over Certification
Control on Results over Control on Activities
Adaptivity over Procedures
http://labs.sogeti.com/wp-content/uploads/2016/03/D2D-4-EN-web.pdf
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
22

1.5: Agile Vs DevOps

Agile Vs DevOps

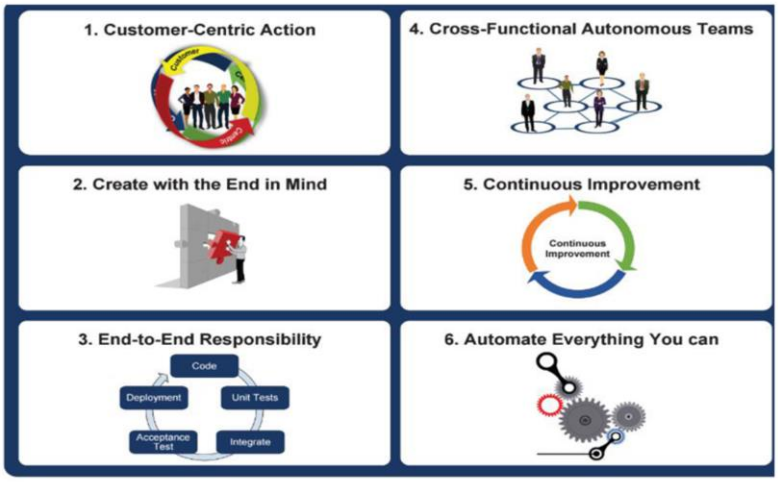


http://labs.sogeti.com/wp-content/uploads/2016/03/D2D-4-EN-web.pdf

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

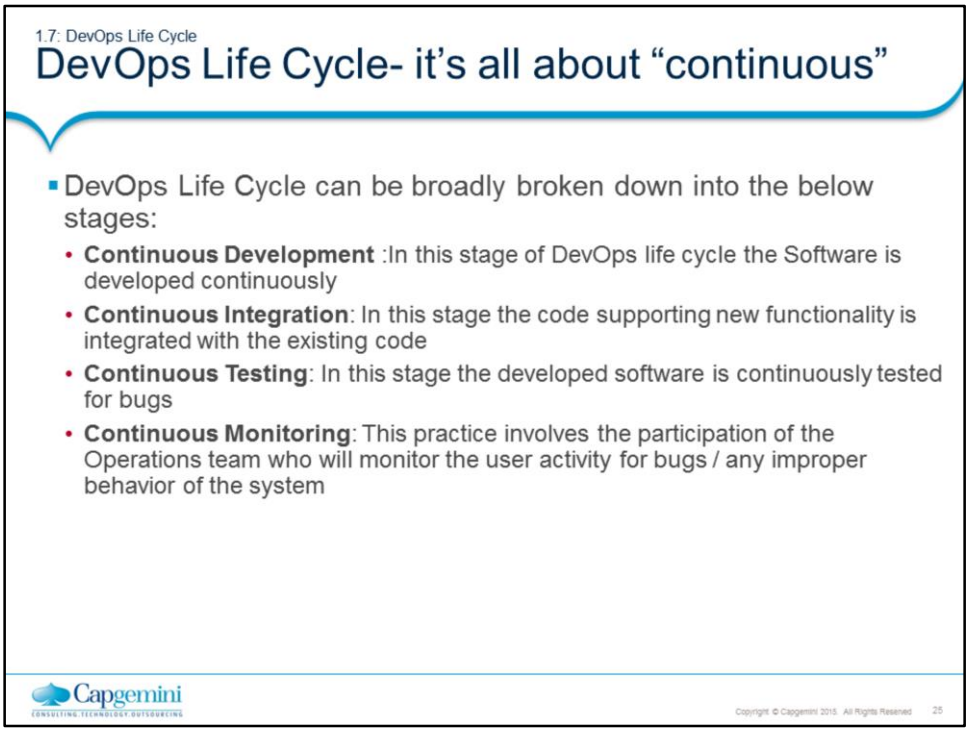
1.6:DevOps Principles

DevOps Principles



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

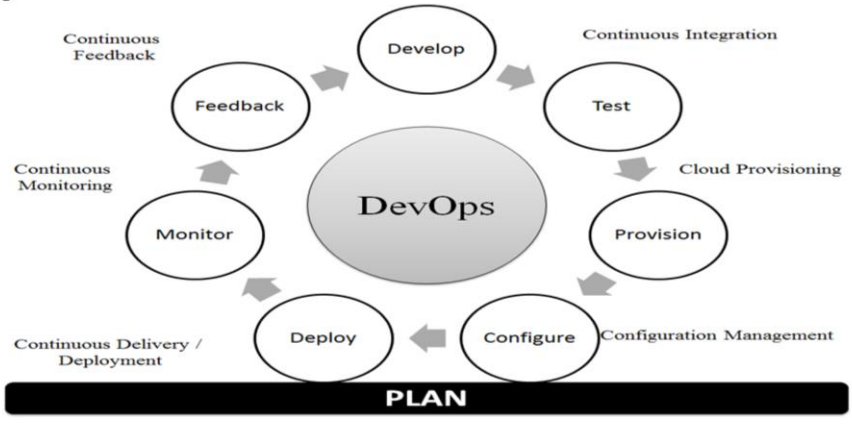
 24


1.7: DevOps Life Cycle
DevOps Life Cycle- it's all about "continuous"
DevOps Life Cycle can be broadly broken down into the below stages:
Continuous Development :In this stage of DevOps life cycle the Software is developed continuously
Continuous Integration: In this stage the code supporting new functionality is integrated with the existing code
Continuous Testing: In this stage the developed software is continuously tested for bugs
Continuous Monitoring: This practice involves the participation of the Operations team who will monitor the user activity for bugs / any improper behavior of the system
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
25

1.7: DevOps Life Cycle

DevOps SDLC

- DevOps life cycle and Software Development stages depicted in the diagram below.

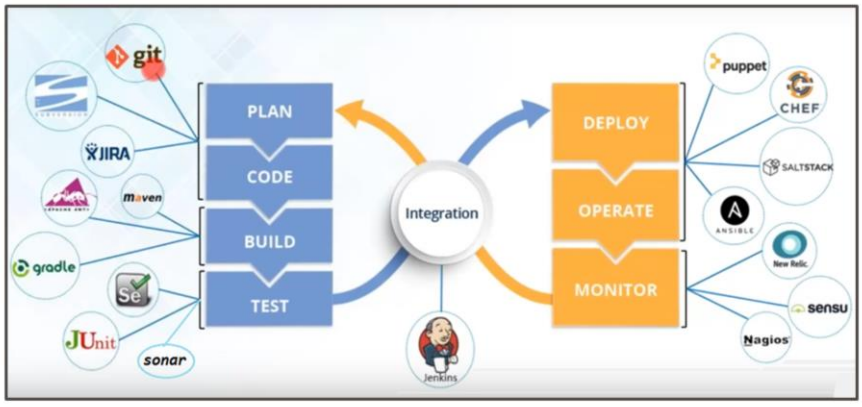


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

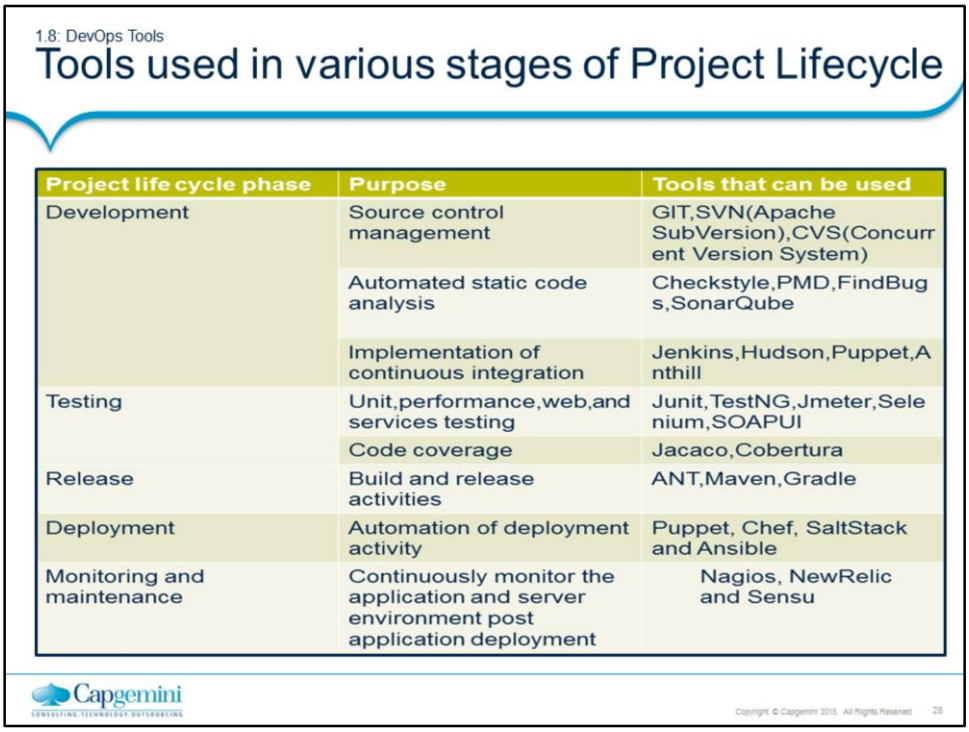
1.8: DevOps Tools

DevOps Tools

- The below diagram shows which tools can be used in which stage of the DevOps life cycle.



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING



Project life cycle phase	Purpose	Tools that can be used
Development	Source control management	GIT,SVN(Apache SubVersion),CVS(Concurrent Version System)
	Automated static code analysis	Checkstyle,PMD,FindBugs,SonarQube
	Implementation of continuous integration	Jenkins,Hudson,Puppet,Anthill
Testing	Unit,performance,web,and services testing	Junit,TestNG,Jmeter,Selenium,SOAPUI
	Code coverage	Jacaco,Cobertura
Release	Build and release activities	ANT,Maven,Gradle
Deployment	Automation of deployment activity	Puppet, Chef, SaltStack and Ansible
Monitoring and maintenance	Continuously monitor the application and server environment post application deployment	Nagios, NewRelic and Sensu

DevOps life cycle stages are carried out on loop continuously until the desired product quality is achieved.

Continuous Development:

This is the stage in the DevOps life cycle where the Software is developed continuously.

This stage involves the Coding and Building phases

Tools Used

Git and SVN are used for maintaining the different versions of the code

Ant, Maven, Gradle for building / packaging the code into an executable file that can be forwarded to the QAs for testing.

Continuous Testing-Test Automation

It is the stage where the developed software is continuously tested for bugs

Once the code is tested, it is continuously integrated with the existing code

Tools Used

For Continuous testing ,testing automation tools like **Selenium**, **JUnit** are used

These tools enables the QA's for testing multiple code-bases thoroughly in parallel to ensure that there are no flaws in the functionality

Continuous Integration:

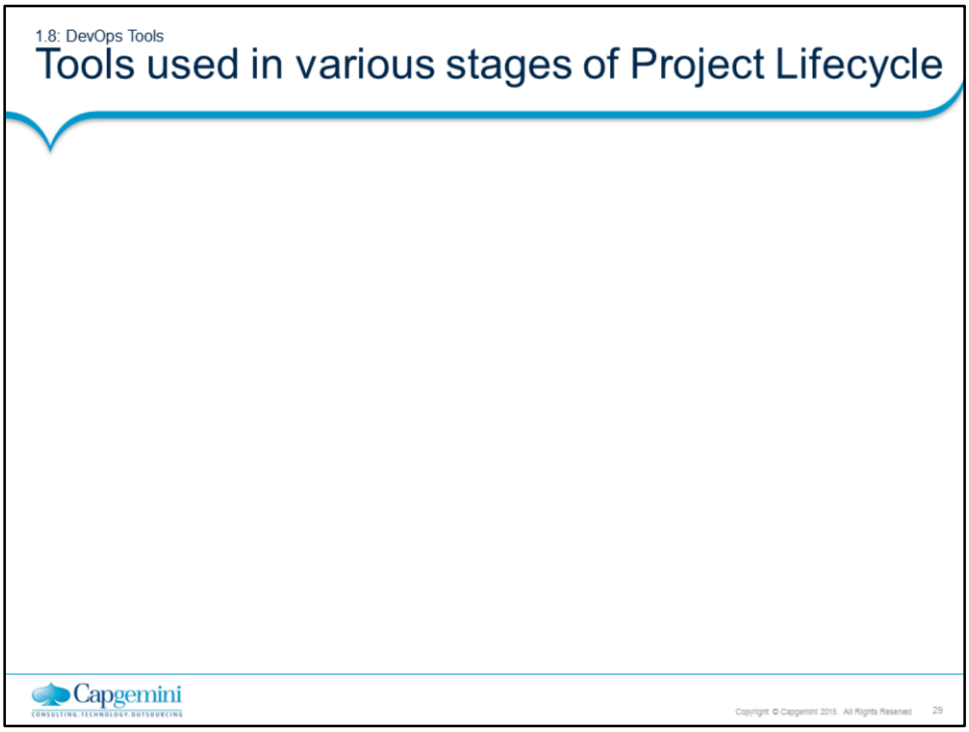
This is the stage where the code supporting new functionality is integrated with the existing code

Since there is continuous development of software, the updated code

needs to be integrated continuously as well as smoothly with the systems to reflect changes to the end users

Tools Used

Jenkins, SonarQube is a very popular tool used for Continuous Integration
Using Jenkins one can pull the latest code revision from GIT repository and produce a build which can finally be deployed to test or production server
It can be set to trigger a new build automatically as soon as there is change in the GIT repository or can be triggered manually on click of a button.



Continuous Deployment:

It is the stage where the code is deployed to the production environment

Here it is ensured that the code is correctly deployed on all the servers

Since the new code is deployed on a continuous basis, automation tools play an important role for executing tasks quickly and frequently

Tools Used

Puppet, **Chef**, **SaltStack** and **Ansible** are some popular tools that are used in this stag

Continuous Monitoring:

This stage in the DevOps life cycle is aimed at improving the quality of the software by monitoring its performance

This practice involves the participation of the Operations team who will monitor the user activity for bugs / any improper behavior of the system

This can also be achieved by making use of dedicated monitoring tools which will continuously monitor the application performance and highlight issues

Tools Used

Some popular tools used are **Nagios**, **NewRelic** and **Sensu**

These tools help us to monitor the application and the servers closely to check the health of the system proactively

They can also improve productivity and increase the reliability of the systems, reducing IT support costs

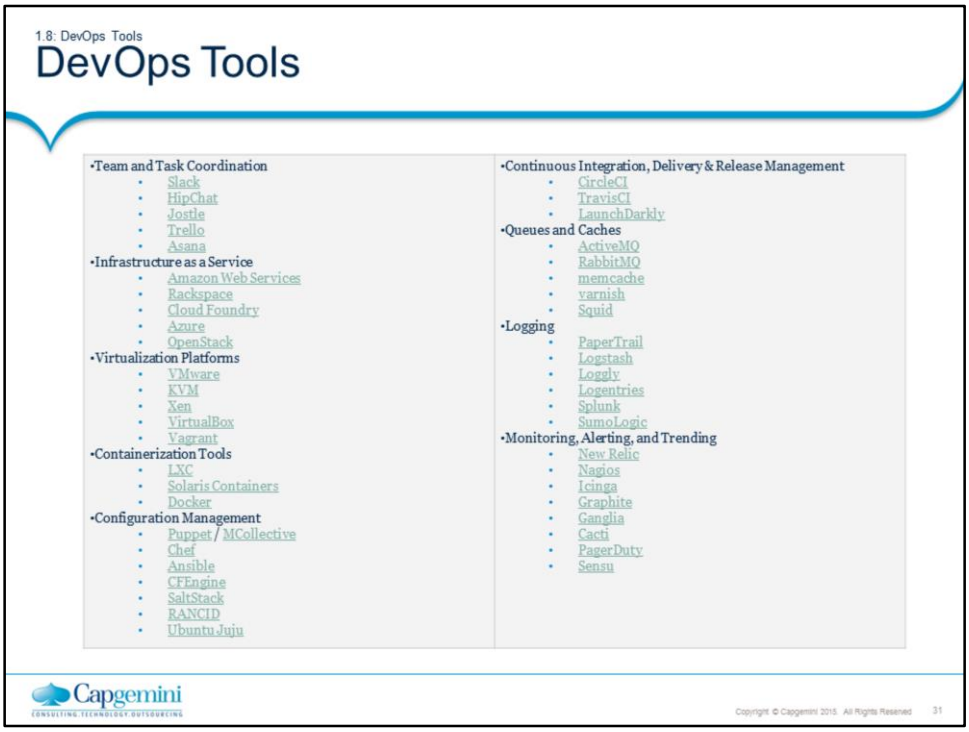
1.8: DevOps Tools

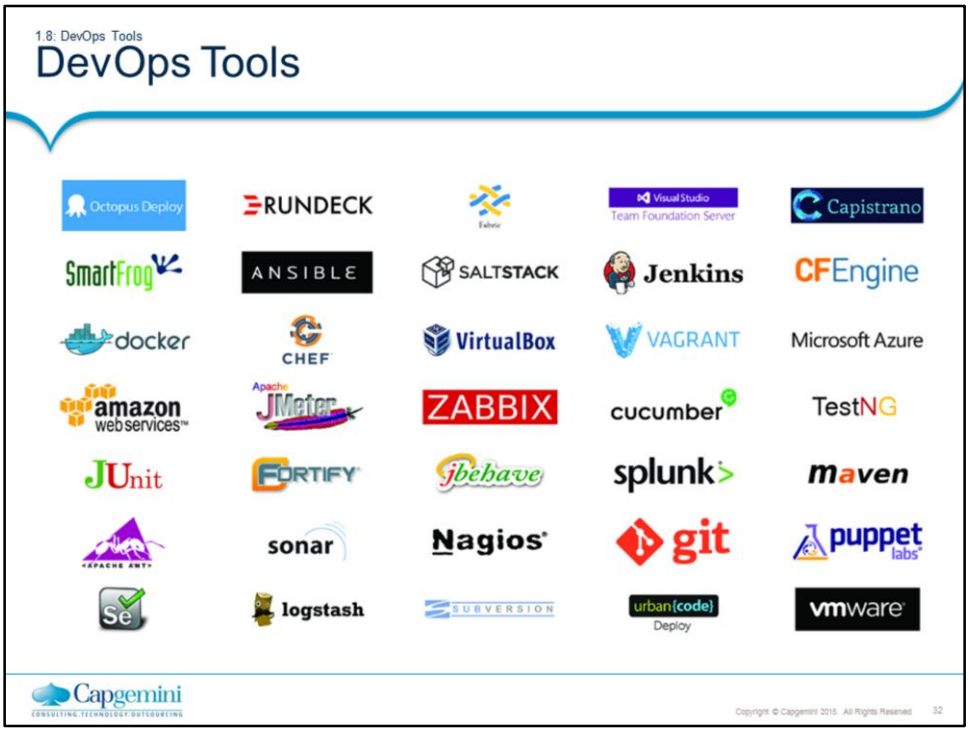
DevOps Tools

- Because DevOps is a cultural shift and collaboration (between development, operations and testing), there is no single "DevOps tool": it is rather a set (or "DevOps toolchain"),consisting of multiple tools.
- DevOps is not about using one particular tool, it is about using the right combination of tools to accelerate development and mitigate risk. This list captures some of emergent DevOps tools categorized by purpose:

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

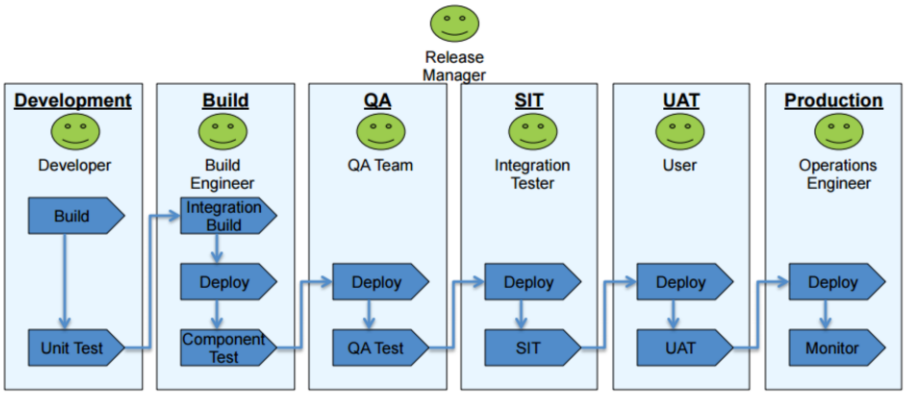
 30


1.8: DevOps Tools
DevOps Tools
•Team and Task Coordination
Slack
HipChat
Jostle
Trello
Asana
•Infrastructure as a Service
Amazon Web Services
Rackspace
Cloud Foundry
Azure
OpenStack
•Virtualization Platforms
VMware
KVM
Xen
VirtualBox
Vagrant
•Containerization Tools
LXC
Solaris Containers
Docker
•Configuration Management
Puppet / MCollective
Chef
Ansible
CFEngine
SaltStack
RANCID
Ubuntu Juju
•Continuous Integration, Delivery & Release Management
CircleCI
TravisCI
LaunchDarkly
•Queues and Caches
ActiveMQ
RabbitMQ
memcache
varnish
Squid
•Logging
PaperTrail
Logstash
Loggly
Logentries
Splunk
SumoLogic
•Monitoring, Alerting, and Trending
New Relic
Nagios
Icinga
Graphite
Ganglia
Cacti
PagerDuty
Sensu
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
31


1.8: DevOps Tools
DevOps Tools
Octopus Deploy
RUNDECK
Fabric
Visual Studio
Team Foundation Server
Capistrano
SmartFrog
ANSIBLE
SALTSTACK
Jenkins
CFEngine
docker
CHEF
VirtualBox
VAGRANT
Microsoft Azure
amazon web services
Apache JMeter
ZABBIX
cucumber
TestNG
JUnit
FORTIFY
jbehave
splunk>
maven
sonar
Nagios
git
puppet labs
logstash
SUBVERSION
urban{code} Deploy
vmware
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
32

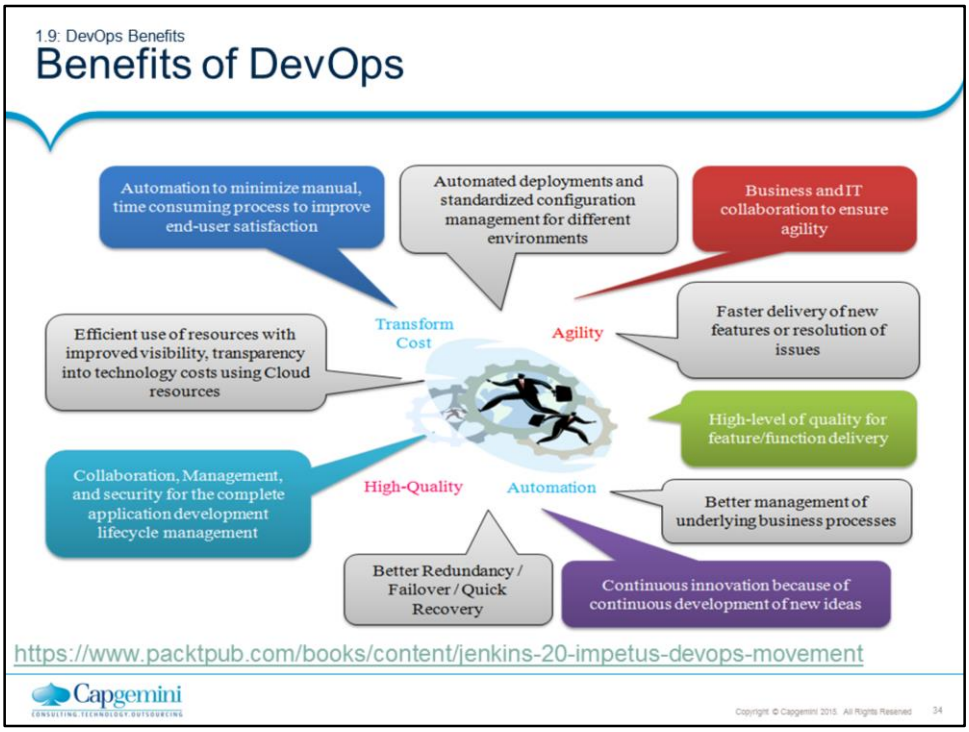
1.8: DevOps Tools

A Typical Deployment Landscape



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 33


1.9: DevOps Benefits
Benefits of DevOps
Automation to minimize manual, time consuming process to improve end-user satisfaction
Automated deployments and standardized configuration management for different environments
Business and IT collaboration to ensure agility
Efficient use of resources with improved visibility, transparency into technology costs using Cloud resources
Transform Cost
Agility
Faster delivery of new features or resolution of issues
High-level of quality for feature/function delivery
Collaboration, Management, and security for the complete application development lifecycle management
High-Quality
Automation
Better management of underlying business processes
Better Redundancy / Failover / Quick Recovery
Continuous innovation because of continuous development of new ideas
https://www.packtpub.com/books/content/jenkins-20-impetus-devops-movement
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
34

1.10: CI&CD

Continuous Integration and Delivery Pipeline

- **Continuous Integration** is a software engineering practice where each check-in made by a developer is verified by either of the following:
  - **Pull mechanism**: Executing an automated build at a scheduled time
  - **Push mechanism**: Executing an automated build when changes are saved in the repository
  - This step is followed by executing a unit test against the latest changes available in the source code repository.

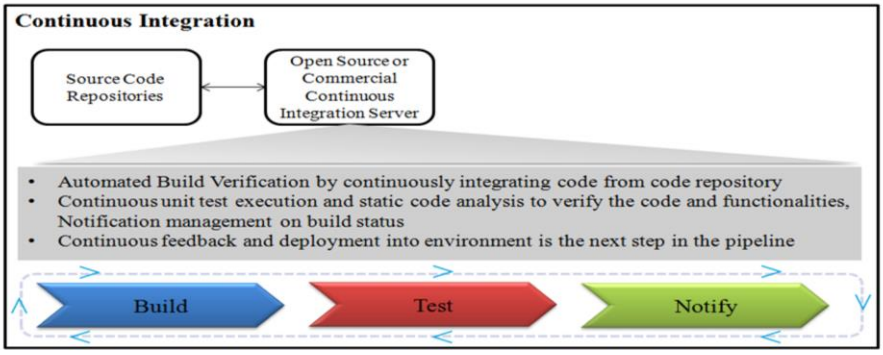
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 35

1.10: CI&CD

Continuous Integration and Delivery Pipeline

- The main benefit of continuous integration is quick feedback based on the result of build execution. If it is successful, all is well; else, assign responsibility to the developer whose commit has broken the build, notify all stakeholders, and fix the issue.



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 36

1.10: CI&CD

Continuous Integration and Delivery Pipeline

- **Continuous Delivery** is a DevOps software development practice where code changes are automatically built, tested, and prepared for a release to production
  - When continuous delivery is implemented properly, developers will always have a deployment-ready build artifact that has passed through a standardized test process
  - With continuous delivery, every code change is built, tested, and then pushed to a non-production testing or staging environment.
  - There can be multiple, parallel test stages before a production deployment
  - In the last step, the developer approves the update to production when they are ready
  - This is different from continuous deployment, where the push to production happens automatically without explicit approval

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 37

1.10: CI&CD

Continuous Integration and Delivery Pipeline

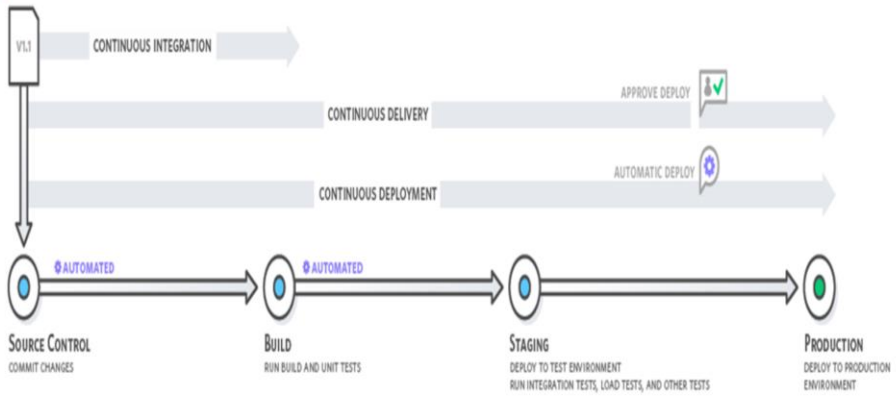
- Continuous delivery lets developers automate testing beyond just unit tests so they can verify application updates across multiple dimensions before deploying to customers.
- These tests may include UI testing, load testing, integration testing, API reliability testing, etc.
- This helps developers more thoroughly validate updates and pre-emptively discover issues.
- With the cloud, it is easy and cost-effective to automate the creation and replication of multiple environments for testing, which was previously difficult to do on-premises.

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 38

1.10: CI&CD

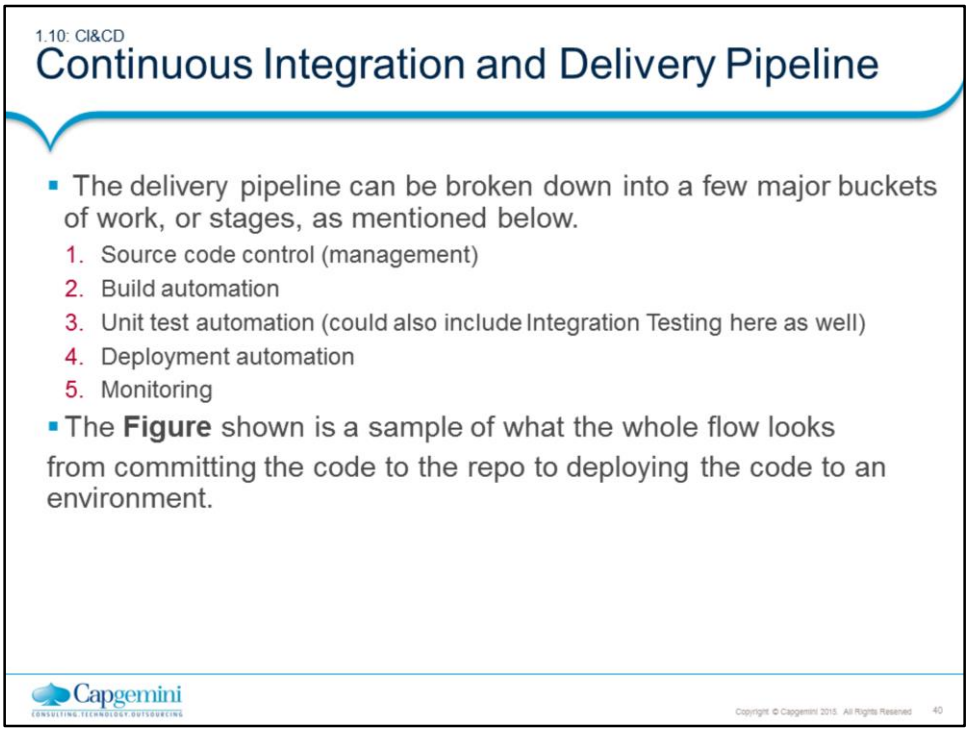
Continuous Integration and Delivery Pipeline



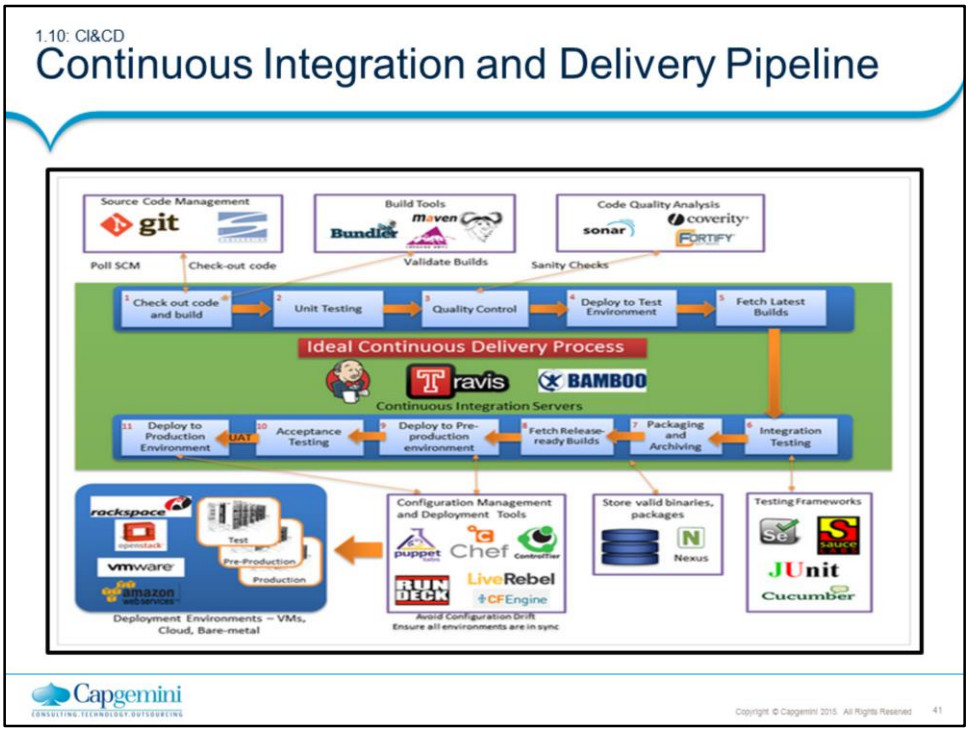
Continuous delivery automates the entire software release process. Every revision that is committed triggers an automated flow that builds, tests, and then stages the update. The final decision to deploy to a live production environment is triggered by the developer

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 39



1. Source Code Control (Management)
The basics here are that the organization stores its code in a source code control system or repository so that it can be tracked, maintained, versioned, and audited. You do not want the developers storing the code on their laptops or virtual machines and trust that will suffice for managing the code.
Possible Tools
Git is probably the most widely used SCM system out there. It is an open source system.
2. Build Automation
Once a source code management system is in place and actively being used by your development team, the team will need to be able to compile and build their code. This is probably the first step in the whole chain of Continuous Integration events. This is what gets the ball rolling. The code needs to build cleanly before you can even think about deploying out to your environments for testing and production.
Possible Tools
Gradle is an open source build automation system. Pretty widely used by top companies like Netflix, Google, and LinkedIn.
Maven is another open source build automation system.



3. Unit Test Automation

Developers unit test their code to ensure that the functionality they are building works as expected. In an ideal world, the development team should be saving these unit tests, so that they can be reused and also put into a regression test bed.

Possible Tools

There are multiple tools out there for helping developers unit test their code. Many of these tools are open source and can be used freely.

JUnit is an open source unit test framework. This is pretty widely used in the industry.

CA DevTest allows for the automation of unit testing, as well as a few other bells and whistles, like service virtualization.

4. Deployment Automation

For the last stage in the process, delivery teams need to deploy their code and applications out to various test environments and, of course, production. To reduce errors and overhead in the deployment process, while increasing speed to market, this step can be automated through a variety of tools and methods.

Possible Tools

IBM Urbancode uDeploy allows you to model a process and orchestrate your deployment. This process can then be repeated across all your environments, and of course tweaked for each environment as needed.

Ansible is an open source IT automation tool. It can be used for everything from configuration management to product installation to application deployments.This tool is rapidly gaining acceptance and momentum in the DevOps community.

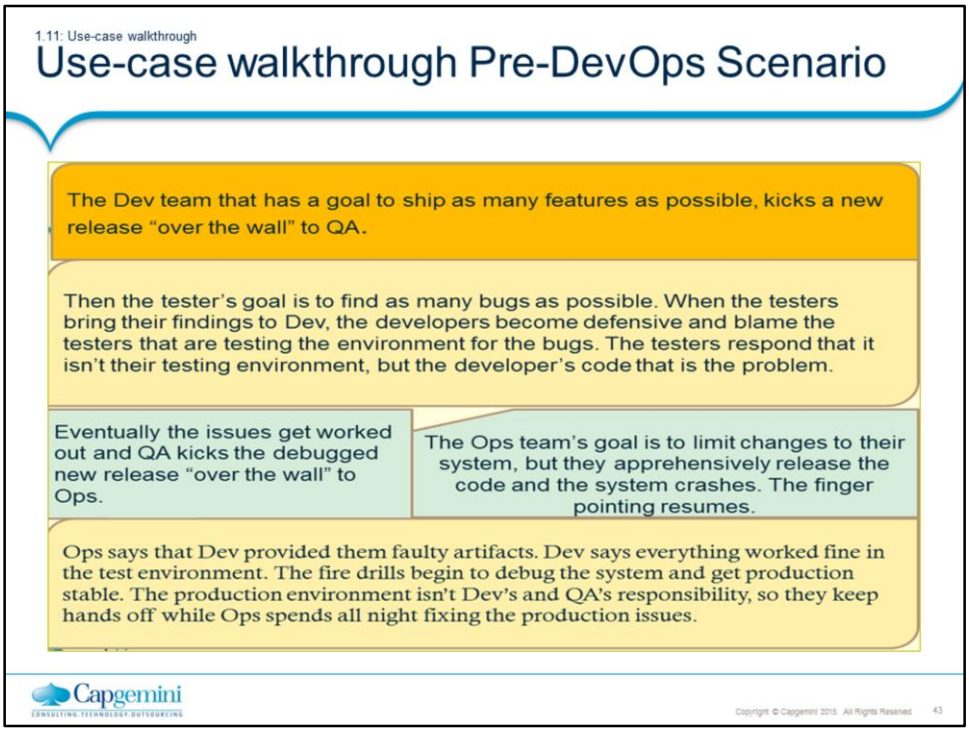
1.10: CI&CD

Continuous Integration and Delivery Pipeline

- Following are immediate benefits of **Continuous Integration**:
  - Automated integration with pull or push mechanism
  - Repeatable process without any manual intervention
  - Automated test case execution
  - Coding standard verification
  - Execution of scripts based on requirement
  - Quick feedback: build status notification to stakeholders via e-mail
  - Teams focused on their work and not in the managing processes
- Benefits **Continuous Delivery**
  - Automate the software release process
  - Improve developer productivity
  - Find and address bugs quickly
  - Deliver updates faster

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

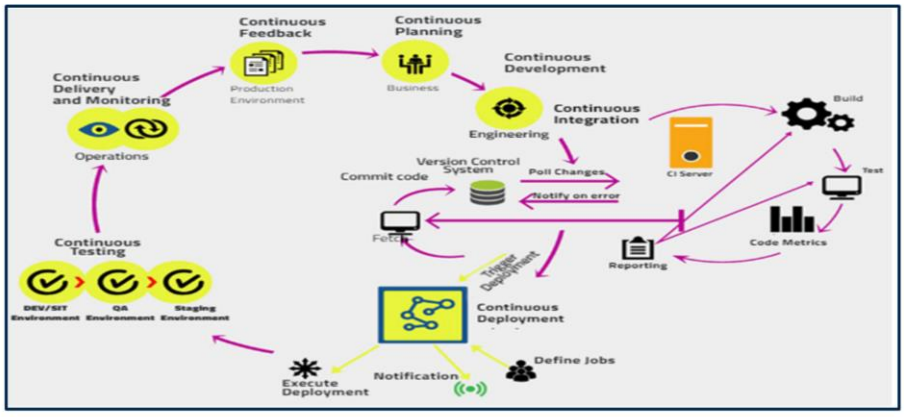
 42


1.11: Use-case walkthrough
Use-case walkthrough Pre-DevOps Scenario
The Dev team that has a goal to ship as many features as possible, kicks a new release "over the wall" to QA.
Then the tester's goal is to find as many bugs as possible. When the testers bring their findings to Dev, the developers become defensive and blame the testers that are testing the environment for the bugs. The testers respond that it isn't their testing environment, but the developer's code that is the problem.
Eventually the issues get worked out and QA kicks the debugged new release "over the wall" to Ops.
The Ops team's goal is to limit changes to their system, but they apprehensively release the code and the system crashes. The finger pointing resumes.
Ops says that Dev provided them faulty artifacts. Dev says everything worked fine in the test environment. The fire drills begin to debug the system and get production stable. The production environment isn't Dev's and QA's responsibility, so they keep hands off while Ops spends all night fixing the production issues.
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
43

1.11: Use-case walkthrough

DevOps End-to-End Implementation

- Analyze, design, construct, automate and implement according to the needs identified for each project



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

1.11: Use-case walkthrough

Use-case walkthrough Post-DevOps Scenario

- The table below lists typical transformations that occur post successful DevOps implementation:

Category	Before Implementation	After Implementation
Teams	Development and operations teams with different goals and processes	Development and operations working as a single global team with common set of goals and processes
	Each team spends considerable, manual effort to execute,develop,test and release management activities	Increased collaboration across all teams using automated processes and consistent goals. This leads to increased team productivity and lowered operations cost
Infrastructure	Nonstandard and fragmented	Organized and standard technology stack
	Manual infrastructure setup and provisioning	Automated, infrastructure provisioning with metrics-based monitoring tools
Delivery model	Waterfall model	Agile and iterative delivery
	Longer release cycle	Incremental releases

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 45

1.11: Use-case walkthrough

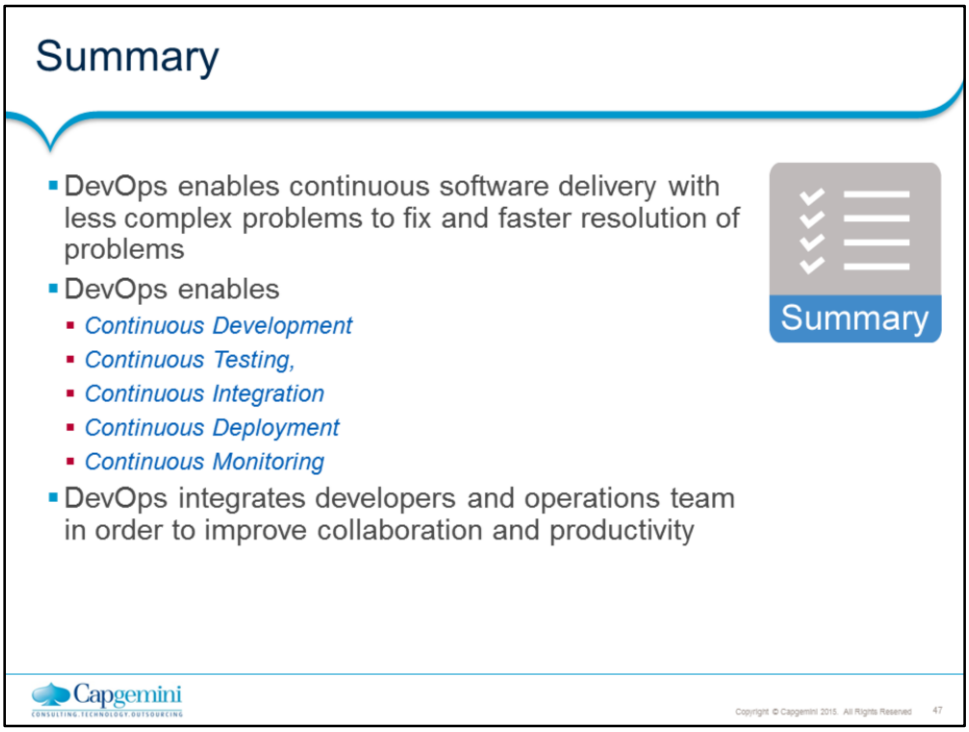
Use-case walkthrough

Category	Before Implementation	After Implementation
Development and testing processes	Traditional	Iterative/agile development and testing with incremental releases
	Manual	Automated testing and continuous validation
	Costly and error prone	Reduced cost and risk due to continuous integration and testing
Effectiveness of end user feedback and change requests	Inability to overcome challenges in handling change requests	Higher effectiveness of change requests,feedback,and enhancement due to agile delivery

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 46

Add instructor notes here.



Add the notes here.

Instructor Notes:

Question 1: True

Question 2: All the above

Question 3:Jenkins

Review Question

- DevOps integrates Developers and Operations team for better collaboration and productivity
  - True
  - False
- Select the tools used for maintaining the different versions of the code
  - GIT
  - SVN
  - Maven
  - All the above
- _____ tool is used for Continuous Integration
  - GIT
  - Puppet
  - Jenkins

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

