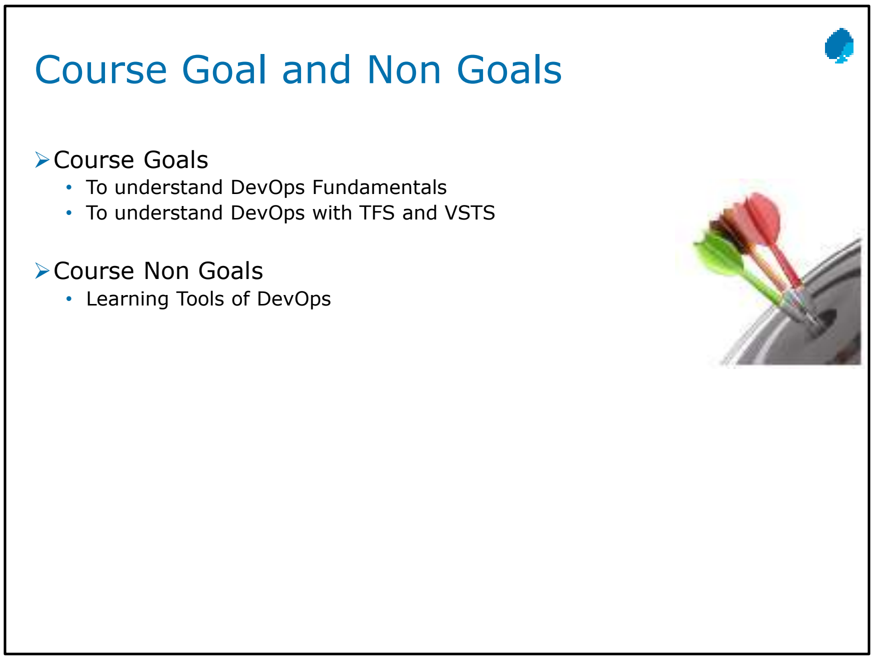
DevOps with TFS

Lesson 01 : Introducing DevOps

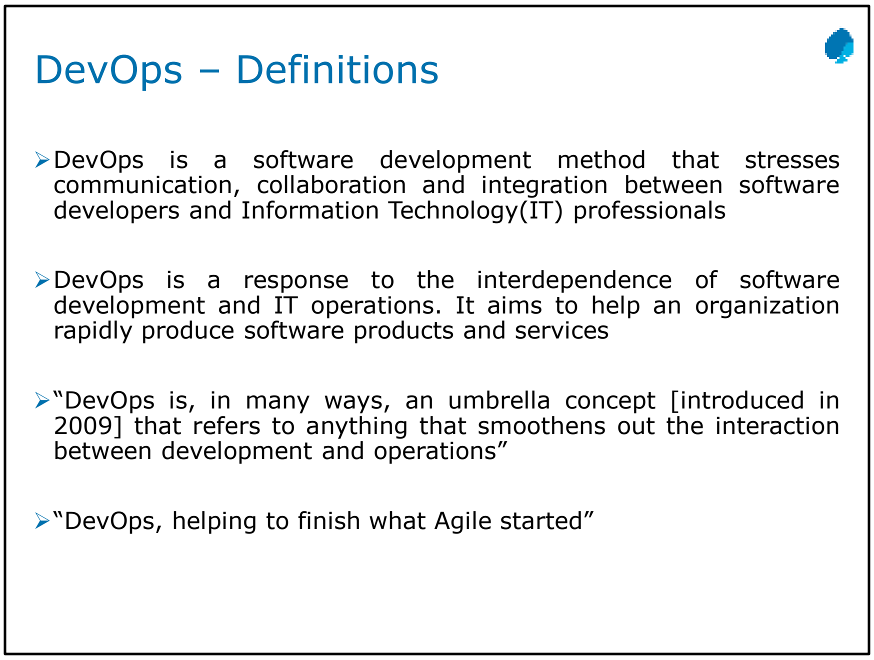
Capgemini



Course Goal and Non Goals

- Course Goals
  - To understand DevOps Fundamentals
  - To understand DevOps with TFS and VSTS
- Course Non Goals
  - Learning Tools of DevOps

Table of Contents

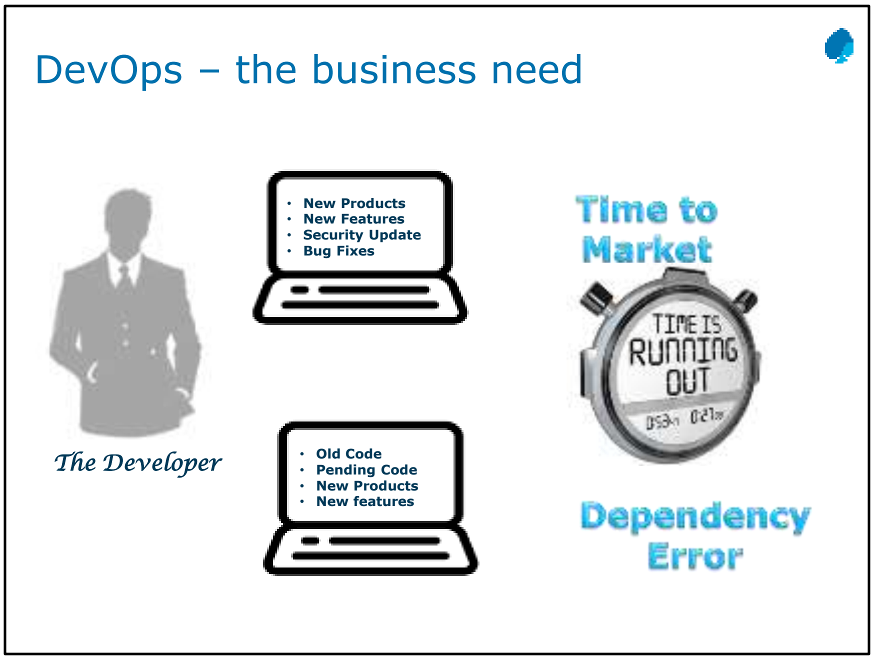


DevOps – Definitions

- DevOps is a software development method that stresses communication, collaboration and integration between software developers and Information Technology(IT) professionals
- DevOps is a response to the interdependence of software development and IT operations. It aims to help an organization rapidly produce software products and services
- "DevOps is, in many ways, an umbrella concept [introduced in 2009] that refers to anything that smoothens out the interaction between development and operations"
- "DevOps, helping to finish what Agile started"

Disadvantages of traditional project management

Tightly controlled projects
User Acceptance Testing is done and customer feedbacks are available only towards the end of the project
Changes or Fixes are expensive
Delayed delivery



DevOps – the business need

- New Products
- New Features
- Security Update
- Bug Fixes

- Old Code
- Pending Code
- New Products
- New features

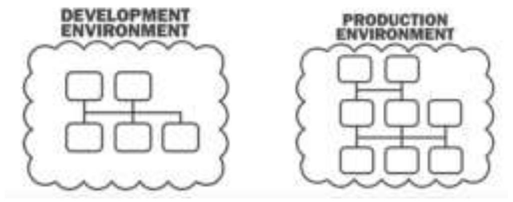
The Developer
Time to Market
TIME IS RUNNING OUT
Dependency Error

DevOps is a culture, movement or practice that emphasizes the collaboration and communication of both software developers and other information-technology (IT) professionals while automating the process of software delivery and infrastructure changes.[1][2] It aims at establishing a culture and environment where building, testing, and releasing software can happen rapidly, frequently, and more reliably

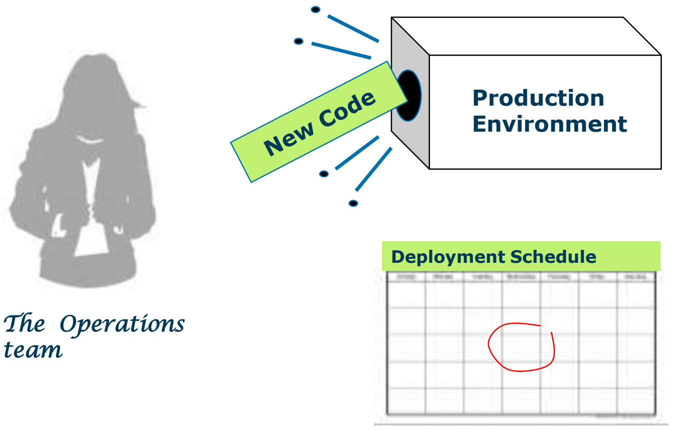
DevOps – the business need

As a developer I have always dabbled lightly in operations. I always wanted to focus on making my code great and let an operations team worry about setting up the production infrastructure.

The Developer



DevOps – the business need

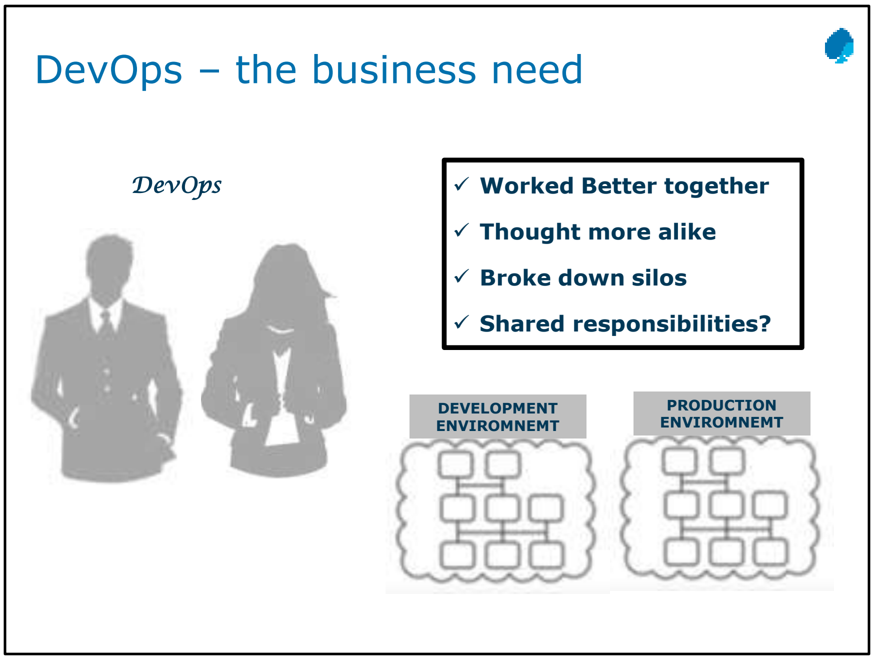


The Operations team

DevOps – the business need

I am responsible for maintaining 99% uptime. I think of servers and new code deployment mostly introduces bugs which I need to fix to ensure availability. These developers are pushing **their** work to me.

The Operations team



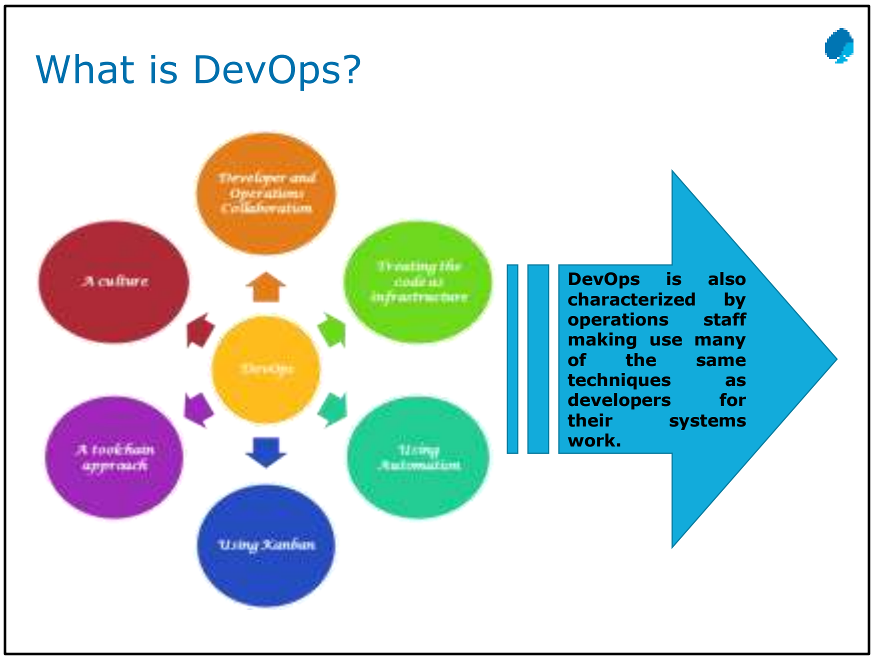
DevOps – the business need

DevOps

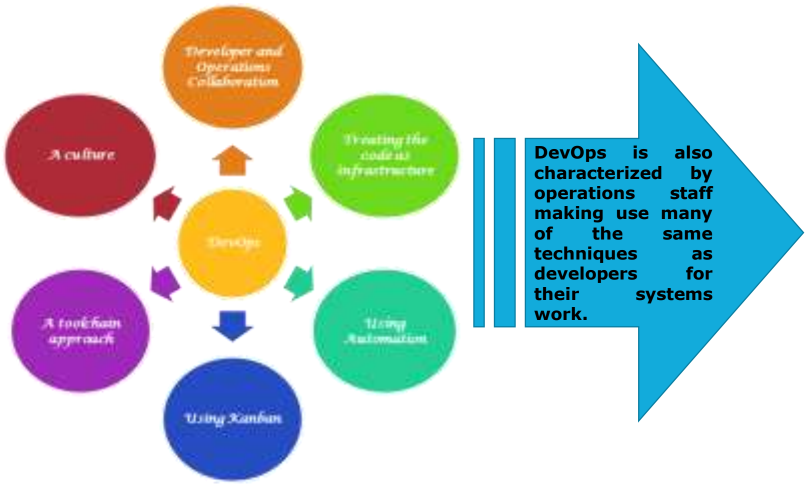
- ✓ Worked Better together
- ✓ Thought more alike
- ✓ Broke down silos
- ✓ Shared responsibilities?

DEVELOPMENT ENVIROMNEMT
PRODUCTION ENVIROMNEMT

DevOps is a culture, movement or practice that emphasizes the collaboration and communication of both software developers and other information-technology (IT) professionals while automating the process of software delivery and infrastructure changes.[1][2] It aims at establishing a culture and environment where building, testing, and releasing software can happen rapidly, frequently, and more reliably



What is DevOps?


Developer and Operations Collaboration
Treating the code as infrastructure
Using Automation
Using Kanban
A toolchain approach
A culture
DevOps

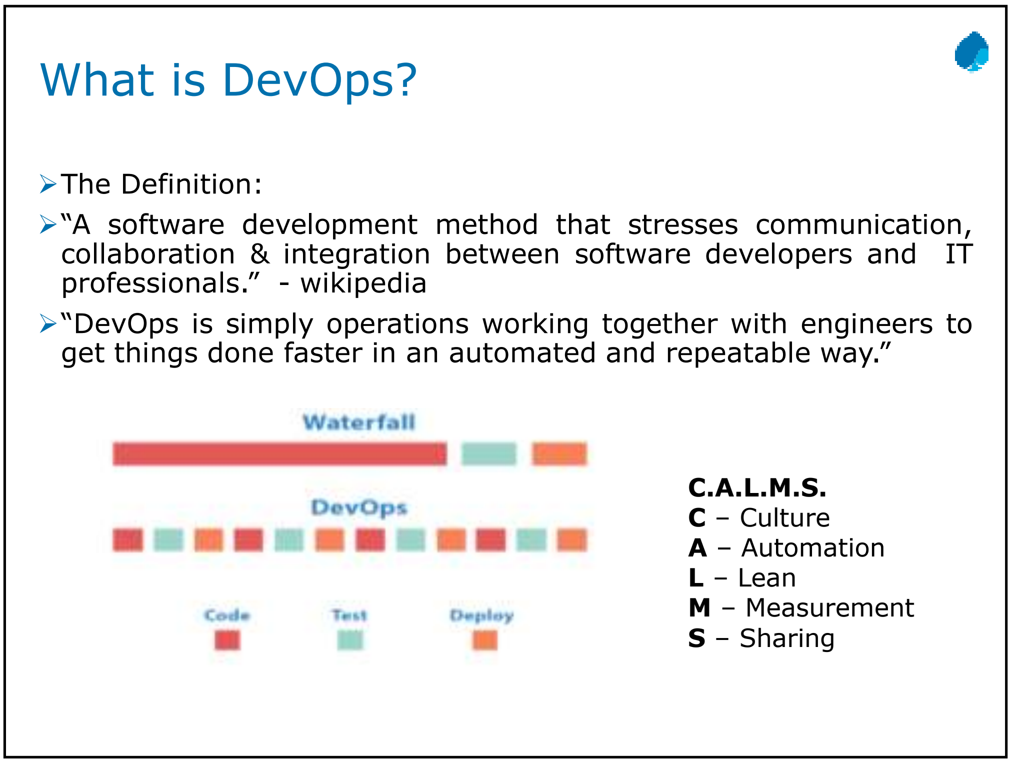
DevOps is also characterized by operations staff making use many of the same techniques as developers for their systems work.

DevOps is a term for a group of concepts that, while not all new, have catalyzed into a movement and are rapidly spreading throughout the technical community. Like any new and popular term, people have somewhat confused and sometimes contradictory impressions of what it is. Like “Quality” or “Agile,” DevOps is a large enough concept that it requires some nuance to fully understand.

DevOps has strong affinities with Agile and Lean approaches. The old view of operations tended towards the “Dev” side being the “makers” and the “Ops” side being the “people that deal with the creation after its birth” – the realization of the harm that has been done in the industry of those two being treated as siloed concerns is the core driver behind DevOps. In this way, DevOps can be interpreted as an outgrowth of Agile – agile software development prescribes close collaboration of customers, product management, developers, and (sometimes) QA to fill in the gaps and rapidly iterate towards a better product – DevOps says “yes, but service delivery and how the app and systems interact are a fundamental part of the value proposition to the client as well, and so the product team needs to include those concerns as a top level item.” From this perspective, DevOps is simply extending Agile principles beyond the boundaries of “the code” to the entire delivered service.

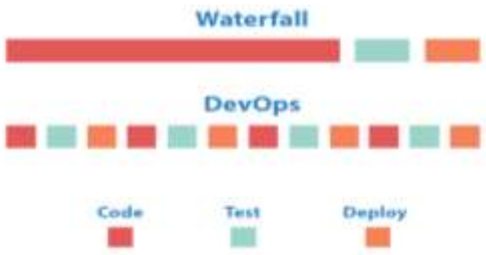
Definition In Depth

DevOps means a lot of different things to different people because the discussion around it covers a lot of ground. People talk about DevOps being “developer and operations collaboration,” or it’s “treating your code as infrastructure,” or it’s “using automation,” or “using kanban,” or “a toolchain approach,” or “culture,” or a variety of seemingly loosely related items.



What is DevOps?

- The Definition:
- "A software development method that stresses communication, collaboration & integration between software developers and IT professionals." - wikipedia
- "DevOps is simply operations working together with engineers to get things done faster in an automated and repeatable way."

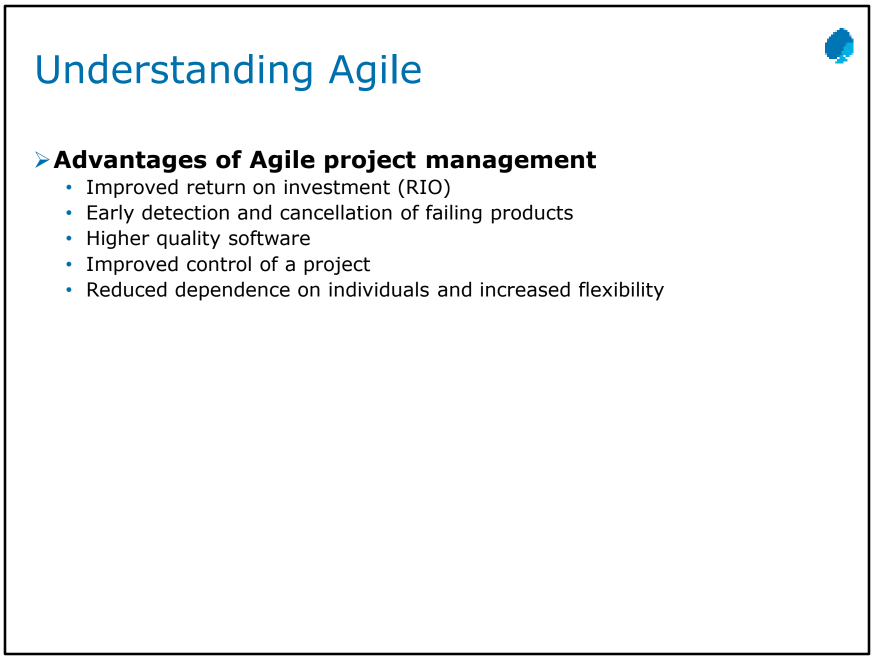


C.A.L.M.S.
C – Culture
A – Automation
L – Lean
M – Measurement
S – Sharing

You can ask 10 people for a definition of DevOps and likely get 10 different answers.
If you want to find a definition of your own, your research will probably begin by asking Google, "what is Dev
So let's delve into how DevOps works and can better adjoin two traditionally opposing departments.
I think DevOps can be explained simply as operations working together with engineers to get things done fas

DevOps promotes agility and continuous delivery in line with customer demands, while maintaining quality. W
there are many definitions of what DevOps is, essentially it's about ensuring that your business can release a
that meets your customer's' needs, and that each release delivers quality.

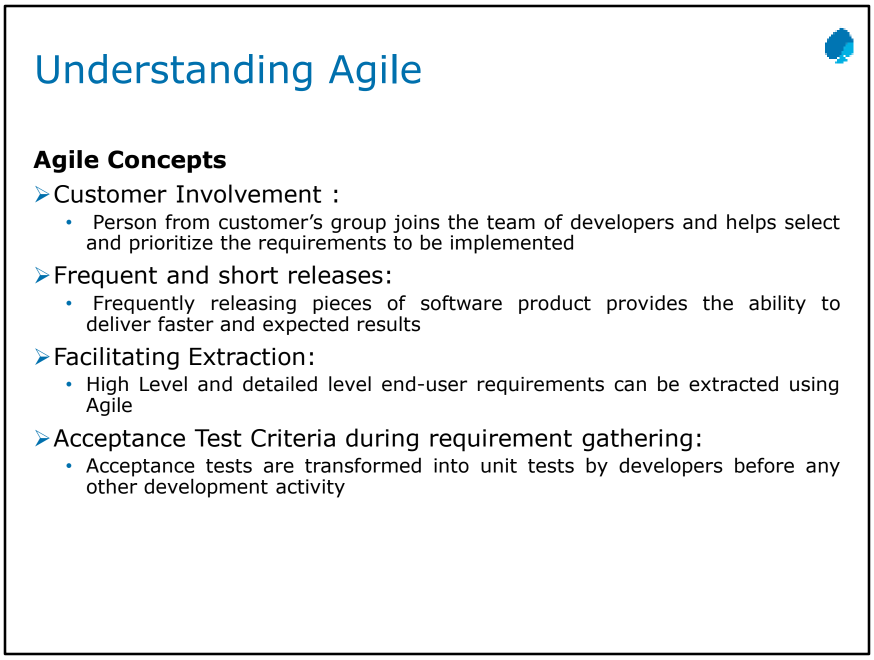
"But," you might ask, "how is the DevOps Lifecycle different?" It can be summed up with the acron



Understanding Agile

➢ **Advantages of Agile project management**

- Improved return on investment (RIO)
- Early detection and cancellation of failing products
- Higher quality software
- Improved control of a project
- Reduced dependence on individuals and increased flexibility



Understanding Agile

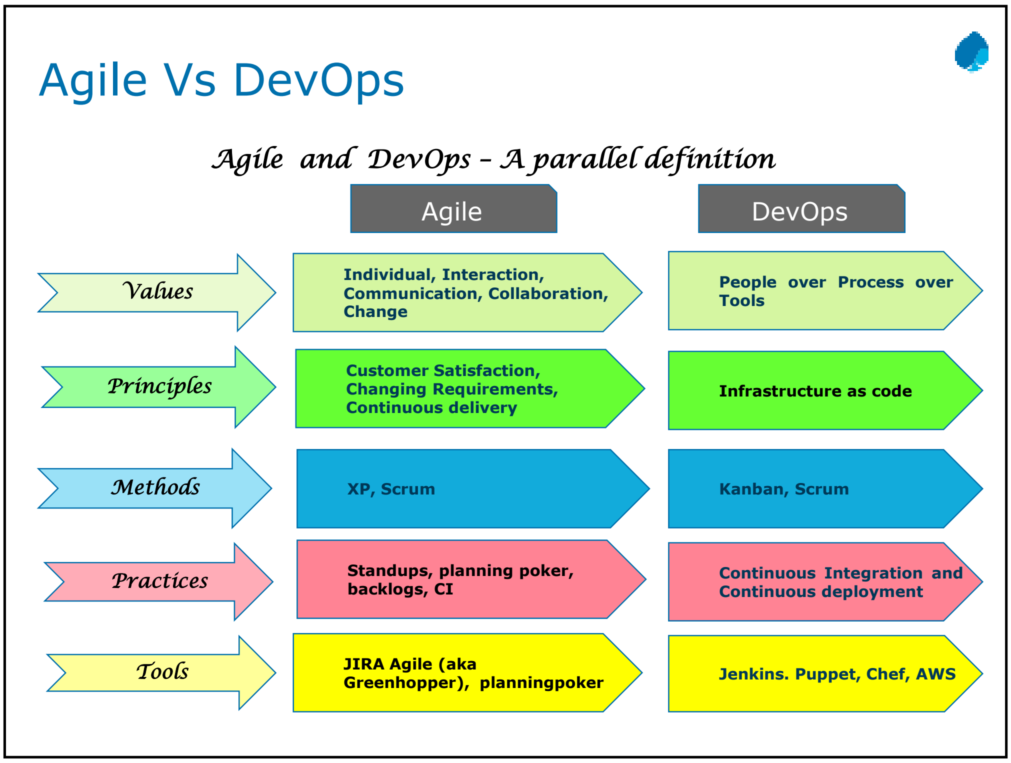
Agile Concepts

- Customer Involvement :
  - Person from customer’s group joins the team of developers and helps select and prioritize the requirements to be implemented
- Frequent and short releases:
  - Frequently releasing pieces of software product provides the ability to deliver faster and expected results
- Facilitating Extraction:
  - High Level and detailed level end-user requirements can be extracted using Agile
- Acceptance Test Criteria during requirement gathering:
  - Acceptance tests are transformed into unit tests by developers before any other development activity

Understanding Agile

Agile Features

- Iterative and evolutionary
- Time boxing
  - Set amount of time for iteration
  - Adapt future iteration based on the realities
- Adaptive planning
- Incremental delivery
- Focused towards success than sticking with a plan
- Working software is valued and considered as a measure of progress

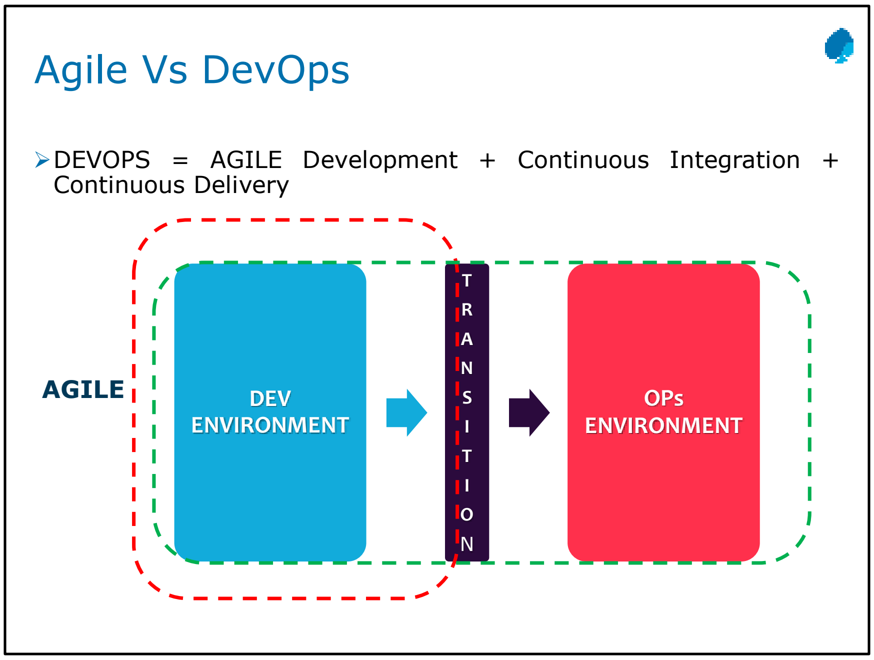


Agile Vs DevOps

Agile and DevOps – A parallel definition

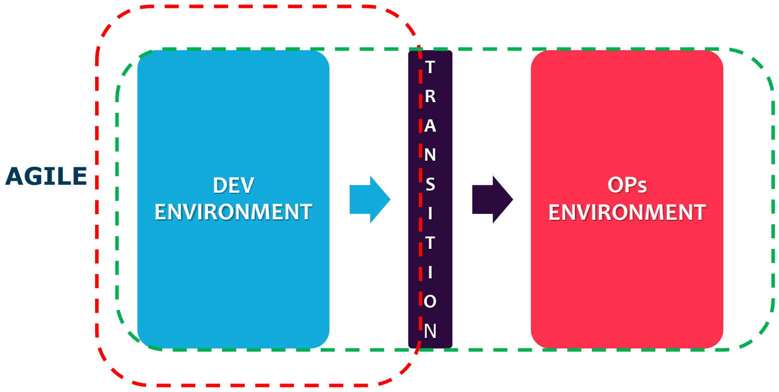
	Agile	DevOps
Values	**Individual, Interaction, Communication, Collaboration, Change**	**People over Process over Tools**
Principles	**Customer Satisfaction, Changing Requirements, Continuous delivery**	**Infrastructure as code**
Methods	**XP, Scrum**	**Kanban, Scrum**
Practices	**Standups, planning poker, backlogs, CI**	**Continuous Integration and Continuous deployment**
Tools	**JIRA Agile (aka Greenhopper), planningpoker**	**Jenkins. Puppet, Chef, AWS**

DevOps means a lot of different things to different people because the discussion around it covers a lot of ground. People talk about DevOps being "developer and operations coll
Agile Values – Top level philosophy, usually agreed to be embodied in the Agile Manifesto. These are the core values that inform agile.
Agile Principles – Generally agreed upon strategic approaches that support these values. The Agile Manifesto cites a dozen of these more specific principles. You don't have to b
Agile Methods – More specific process implementations of the principles. XP, Scrum, your own homebrew process – this is where the philosophy gives way to operational playbo
Agile Practices – highly specific tactical techniques that tend to be used in conjunction with agile implementations. None are required to be agile but many agile implementations
Agile Tools – Specific technical implementations of these practices used by teams to facilitate doing their work according to these methods. JIRA Agile (aka Greenhopper), planni
Ideally the higher levels inform the lower levels – people or organizations that pick up specific tools and practices without understanding the fundamentals may or may not see bene
DevOps Values – I believe the fundamental DevOps values are effectively captured in the Agile Manifesto – with perhaps one slight emendation to focus on the overall service or s
DevOps Principles – There is not a single agreed upon list, but there are several widely accepted attempts – here's John Willis coining "CAMS" andhere's James Turnbull giving h
DevOps Methods – Some of the methods here are the same; you can use Scrum with operations, Kanban with operations, etc. (although usually with more focus on integrating op
DevOps Practices –Specific techniques used as part of implementing the above concepts and processes. Continuous integration and continuous deployment, "Give your develop
DevOps Tools – Tools you'd use in the commission of these principles. In the DevOps world there's been an explosion of tools in release (jenkins, travis, teamcity), configuration r



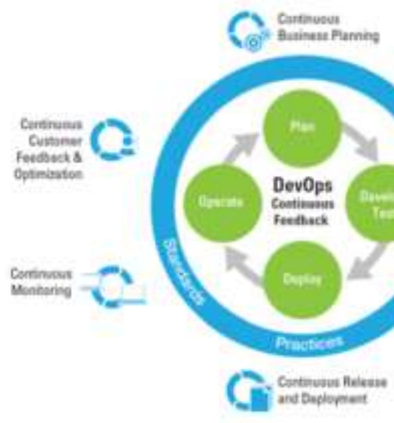
Agile Vs DevOps

- DEVOPS = AGILE Development + Continuous Integration + Continuous Delivery



DevOps is not a replacement of Agile.
Agile+ Continuous Integration + Continuous Deployment = DevOps

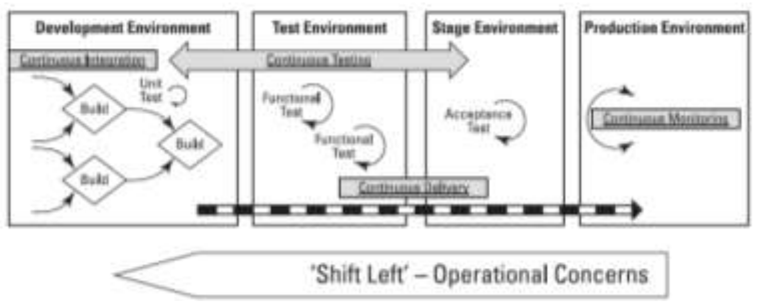
DevOps Life Cycle


Continuous Business Planning
Continuous Customer Feedback & Optimization
Collaborative Development
Continuous Monitoring
Continuous Testing
Continuous Release and Deployment
Plan
Operate
DevOps Continuous Feedback
Develop/ Test
Deploy
Standards
Practices
Ecosystem

A *reference architecture* provides a template of a proven solution by using a set of preferred methods and capabilities. The DevOps reference architectures discussed in this chapter help practitioners access and use the guidelines, directives, and other material that they need to architect or design a DevOps platform that accommodates people, processes, and technology (see Chapter 3). A reference architecture provides capabilities through its various components. These capabilities in turn may be provided
by a single component or a group of components working together. Therefore, you can view the DevOps reference architecture, shown in Figure 2-1 , from the perspective of the
core capabilities that it's intended to provide. As the abstract architecture evolves to concrete form, these capabilities are provided by a set of effectively enabled people, defined practices, and automation tools. undersized and basically wasn't going to work. The CIO of the school district scurried for other storage options that could quickly overcome the hardware failure. After looking at other solutions that could take up to 30 days to deliver, the district needed help sooner and decided on

How DevOps Works?

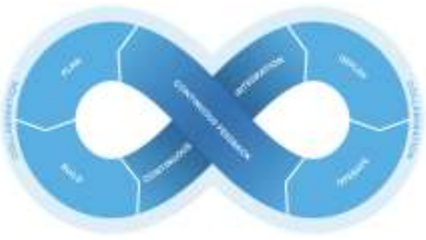
- Develop and test against production-like systems
- Deploy with repeatable, reliable processes
- Monitor and validate operational quality
- Amplify feedback loops

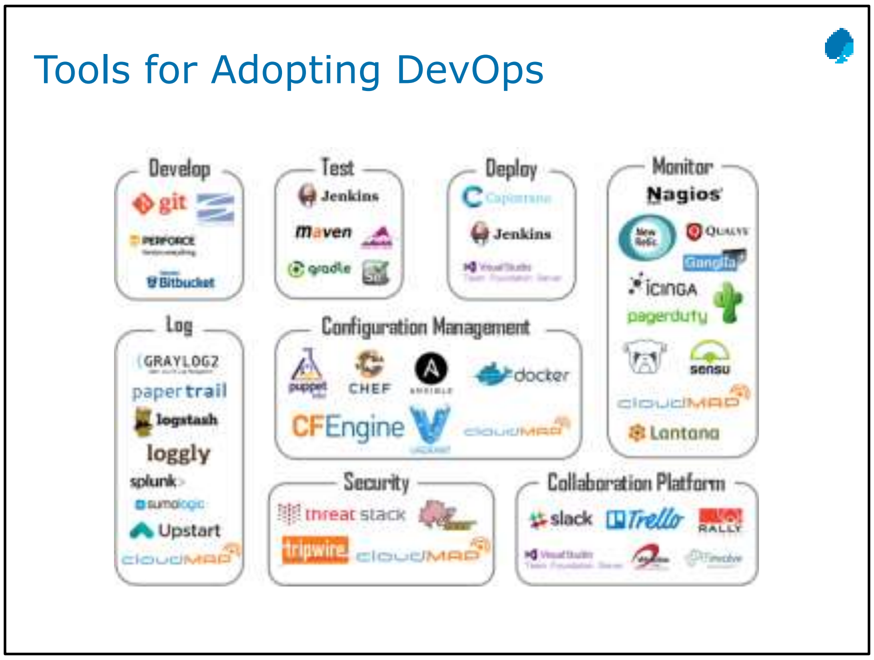


Shift Left – Operational Concerns:
This principle stems from the DevOps concept *shift left,* in which operations concerns move earlier in the software delivery life cycle, toward development . The goal is to allow development and quality assurance (QA) teams to develop and test against systems that behave like the production system, so that they can see how the
application behaves and performs well before it's ready for deployment. The first exposure of the application to a production-like system should be as early in the life cycle as possible to address two major potential challenges. First, it allows the application to be tested in an environment that's close to the actual production environment the application will be delivered to; and second, it allows for the application delivery processes themselves to be tested and validated upfront. From an operations perspective, too, this principle has tremendous value. It enables the operations team to see early in the cycle how their environment will behave when it supports the application, thereby allowing them to create a fine-tuned, application-aware environment.

How DevOps Works?

- Plan:
  - Focuses on establishing business goals and adjusting them based on customer feedback: continuous business planning.
- Develop/Test:
  - Forms the core of development and quality assurance (QA) capabilities. It involves two practices - collaborative development and continuous testing.
- Deploy
  - Continuous release and deployment take the concept of continuous integration to the next step
- Operate
  - It involves two practices - continuous monitoring and continuous customer feedback.

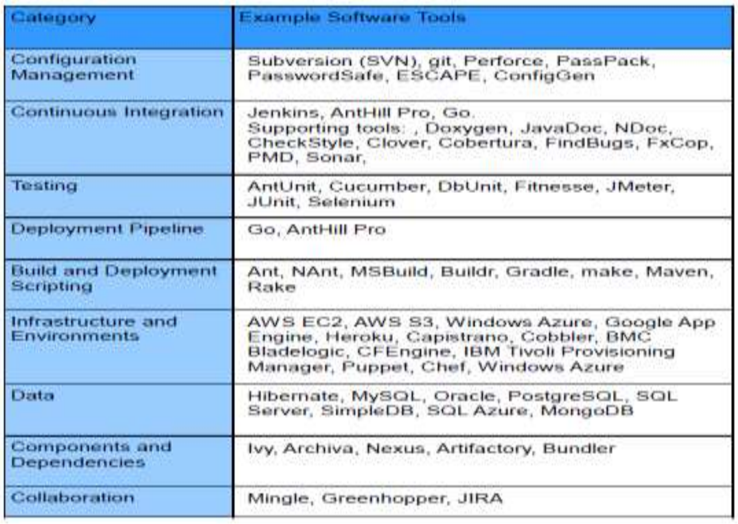




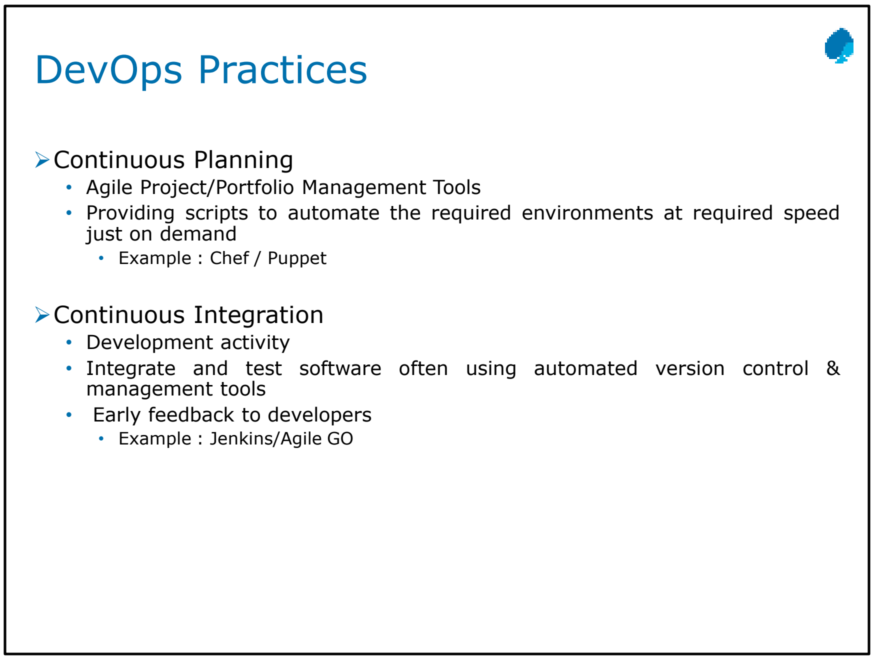
Tools for Adopting DevOps

- *Version control*
  - *GitHub, Mercurial, Perforce, Subversion, Team Foundation Server*
- *Configuration management*
  - *Docker, Puppet*
- *Continuous integration*
  - *Jenkins, TeamCity, Travis CI, Atlassian Bamboo, Go*
- *Deployment*
  - *Capistrano, MCollective [part of Puppet Enterprise]*
- *Monitoring*
  - *New Relic, Nagios, Splunk, AppDynamics, Loggly, Elastic*

Tools for Adopting DevOps



Category	Example Software Tools
Configuration Management	Subversion (SVN), git, Perforce, PassPack, PasswordSafe, ESCAPE, ConfigGen
Continuous Integration	Jenkins, AntHill Pro, Go. Supporting tools: , Doxygen, JavaDoc, NDoc, CheckStyle, Clover, Cobertura, FindBugs, FxCop, PMD, Sonar,
Testing	AntUnit, Cucumber, DbUnit, Fitnesse, JMeter, JUnit, Selenium
Deployment Pipeline	Go, AntHill Pro
Build and Deployment Scripting	Ant, NAnt, MSBuild, Buildr, Gradle, make, Maven, Rake
Infrastructure and Environments	AWS EC2, AWS S3, Windows Azure, Google App Engine, Heroku, Capistrano, Cobbler, BMC Bladelogic, CFEngine, IBM Tivoli Provisioning Manager, Puppet, Chef, Windows Azure
Data	Hibernate, MySQL, Oracle, PostgreSQL, SQL Server, SimpleDB, SQL Azure, MongoDB
Components and Dependencies	Ivy, Archiva, Nexus, Artifactory, Bundler
Collaboration	Mingle, Greenhopper, JIRA



DevOps Practices

- ➢ Continuous Planning
  - Agile Project/Portfolio Management Tools
  - Providing scripts to automate the required environments at required speed just on demand
    - Example : Chef / Puppet
- ➢ Continuous Integration
  - Development activity
  - Integrate and test software often using automated version control & management tools
  - Early feedback to developers
    - Example : Jenkins/Agile GO

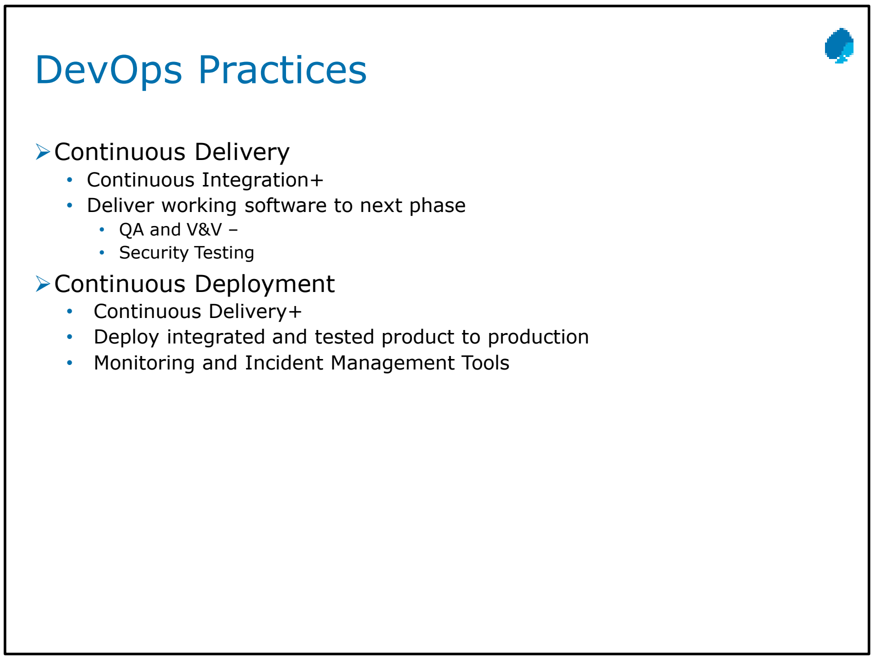
Infrastructure as Code

To enable continuous delivery.
Automated frequent builds on various configuration environments and instances on automated CI environments demands continued automation
Providing scripts to automate the create the required environments at required speed just on demand

Example : Chef / Puppet

Continuous Integration
Automated Testing

Includes testing for each environment in the pipeline
Staging Environment
Performance Testing
Stress Testing
Load Testing
End-To-End Testing
System Testing

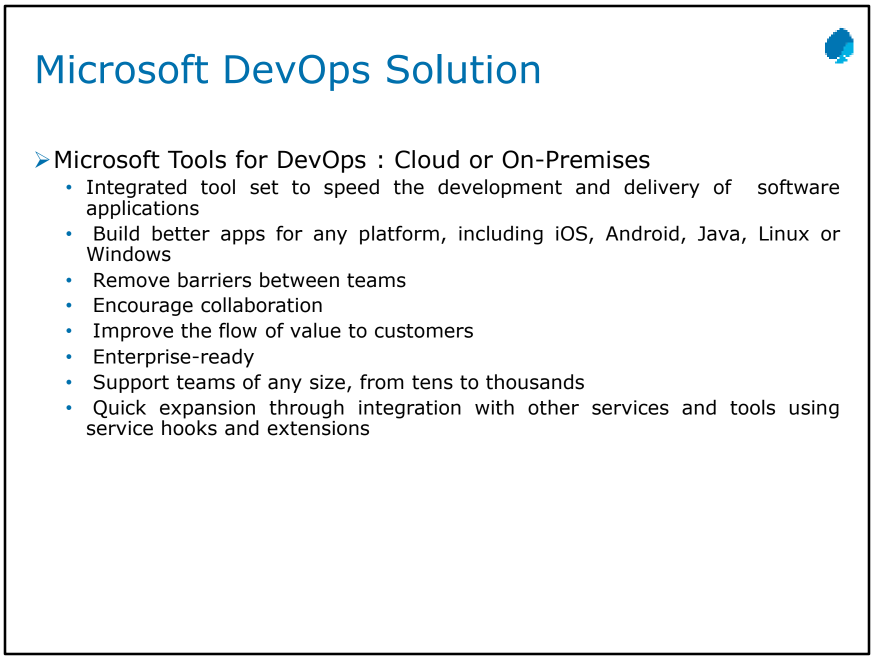


DevOps Practices

- Continuous Delivery
  - Continuous Integration+
  - Deliver working software to next phase
    - QA and V&V –
    - Security Testing
- Continuous Deployment
  - Continuous Delivery+
  - Deploy integrated and tested product to production
  - Monitoring and Incident Management Tools

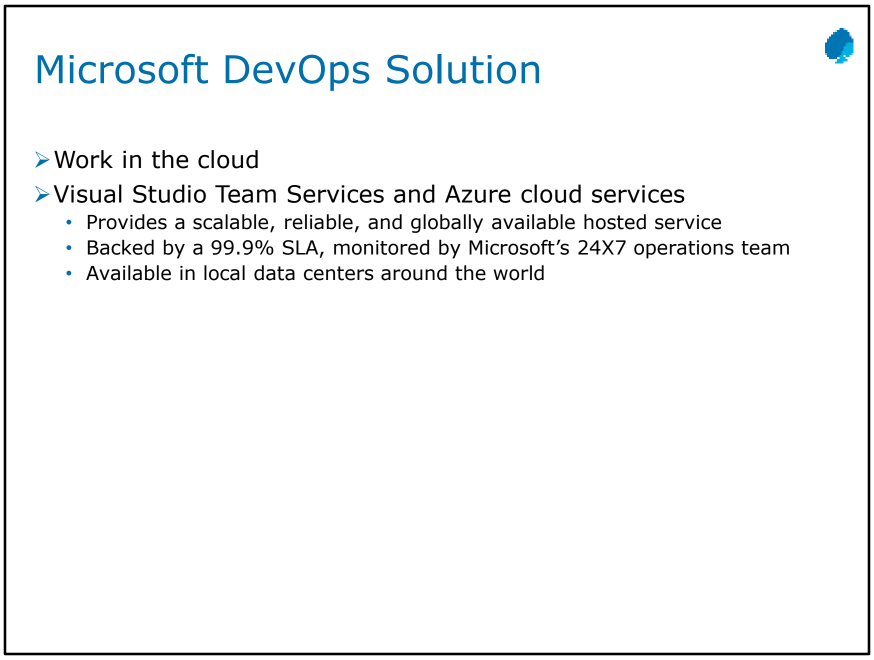
Continuous Deployment & Release Management

Continuous release and deployment makes it possible to release new features to customers and users at the earliest possible..
Correct selection of tooling and processes make up the core of DevOps to facilitate continuous integration, continuous release, and continuous deployment.



Microsoft DevOps Solution

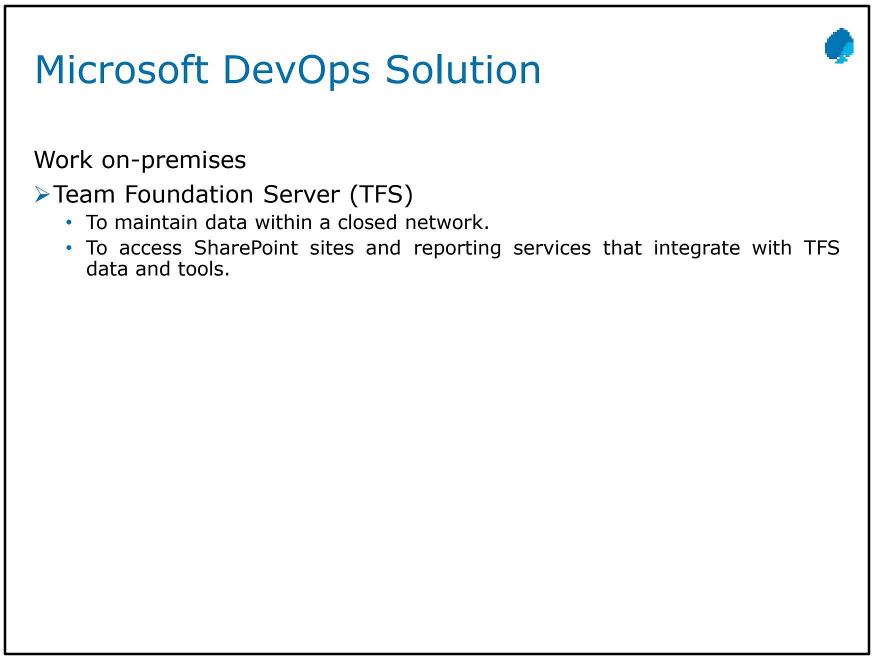
- Microsoft Tools for DevOps : Cloud or On-Premises
  - Integrated tool set to speed the development and delivery of software applications
  - Build better apps for any platform, including iOS, Android, Java, Linux or Windows
  - Remove barriers between teams
  - Encourage collaboration
  - Improve the flow of value to customers
  - Enterprise-ready
  - Support teams of any size, from tens to thousands
  - Quick expansion through integration with other services and tools using service hooks and extensions



Microsoft DevOps Solution

- Work in the cloud
- Visual Studio Team Services and Azure cloud services
  - Provides a scalable, reliable, and globally available hosted service
  - Backed by a 99.9% SLA, monitored by Microsoft’s 24X7 operations team
  - Available in local data centers around the world

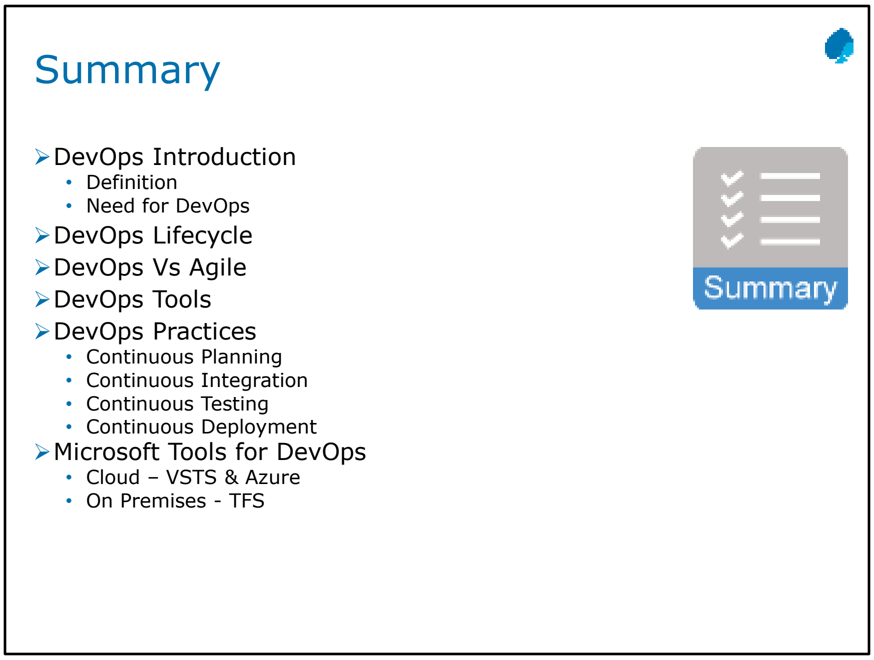
Choose Team Services for quick setup, maintenance-free operations, easy collaboration across domains, elastic scale, and rock solid security.



Microsoft DevOps Solution

Work on-premises

- Team Foundation Server (TFS)
  - To maintain data within a closed network.
  - To access SharePoint sites and reporting services that integrate with TFS data and tools.



Summary

- ➢ DevOps Introduction
  - Definition
  - Need for DevOps
- ➢ DevOps Lifecycle
- ➢ DevOps Vs Agile
- ➢ DevOps Tools
- ➢ DevOps Practices
  - Continuous Planning
  - Continuous Integration
  - Continuous Testing
  - Continuous Deployment
- ➢ Microsoft Tools for DevOps
  - Cloud – VSTS & Azure
  - On Premises - TFS

Summary