

Video 1.4

Tools and Technologies



In this Video, we are going to take a look at...

- Provide support for nonlinear development
- Build tools with the Apache 2.0 license

Code Repositories

- Subversion is a control system that is used to track all the changes made to files and folders
- The code is sent to the repository to track the changes
- A comment can be made by the developer so that other developers can easily understand changes that were made

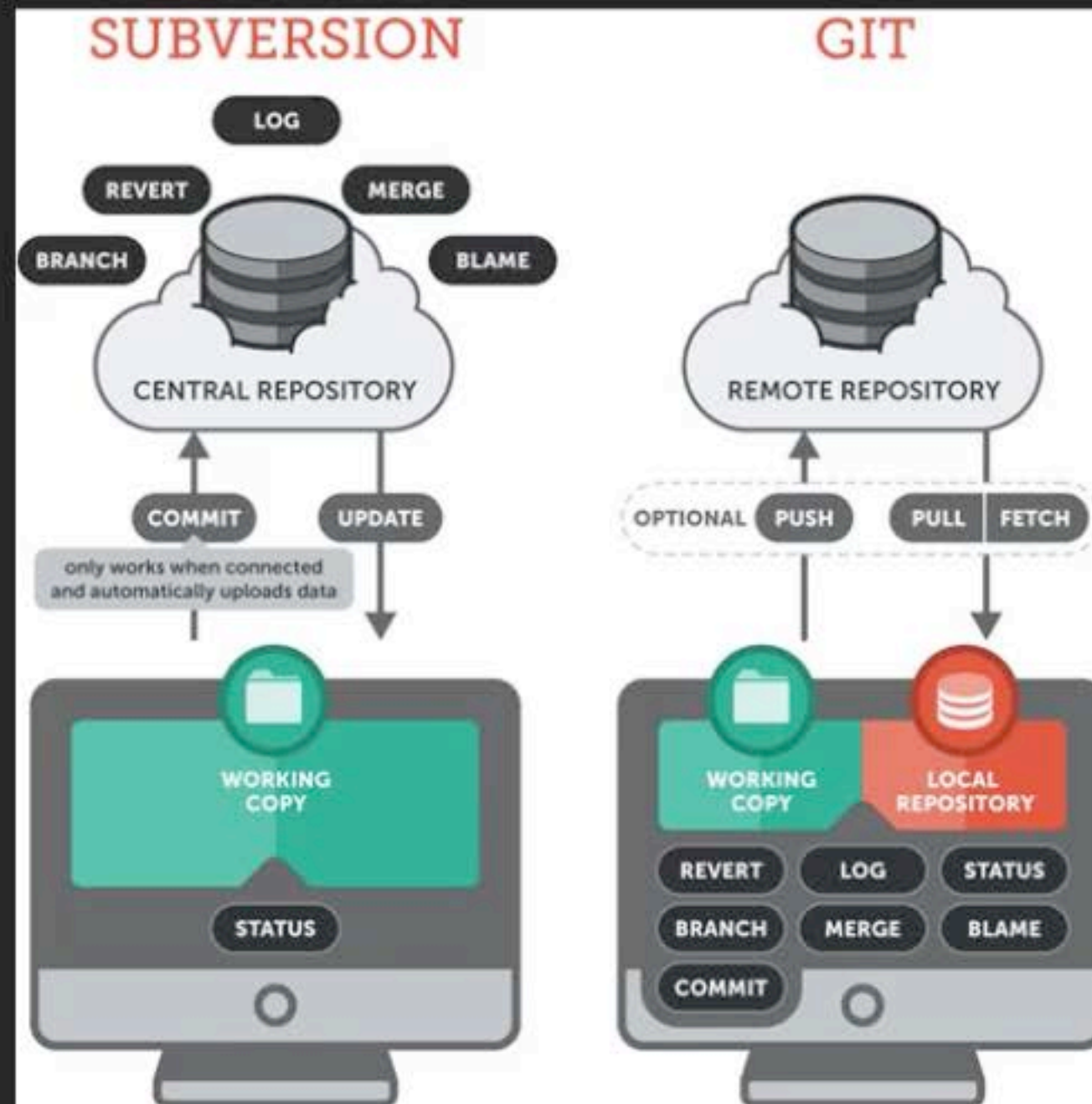
Advantages

- Many developers can work simultaneously on the same code
- If a computer crashes, the code can still be recovered as it had been committed in the server
- If a bug occurs, the new code can be easily reverted to the previous version

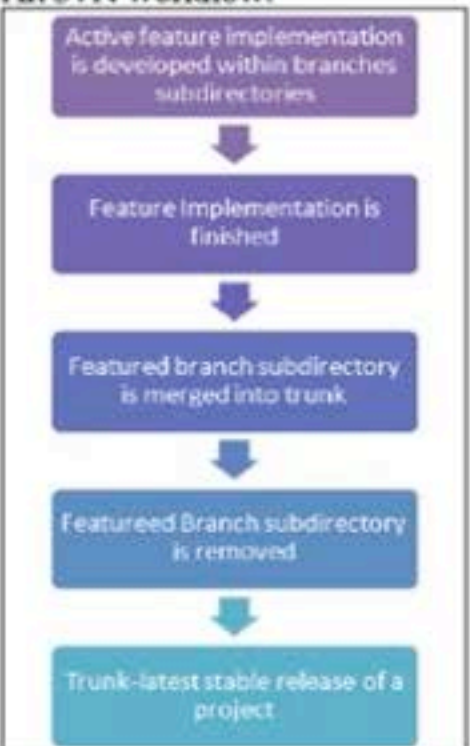

Characteristics

- It provides support for nonlinear development
- It is compatible with existing systems and protocols
- It ensures the cryptographic authentication of history
- It has well-designed pluggable merge strategies
- It consists of toolkit-based designs
- It supports various merging techniques, such as resolve, octopus, and recursive

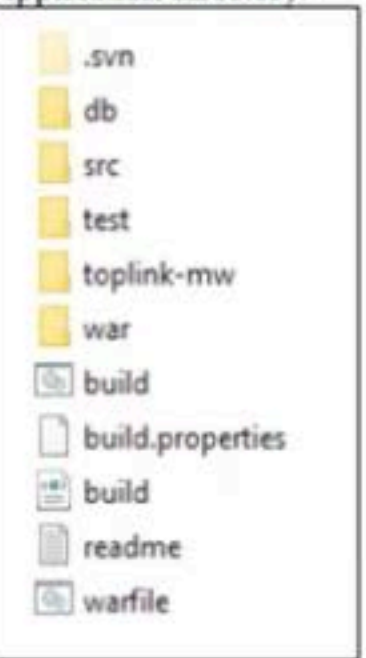
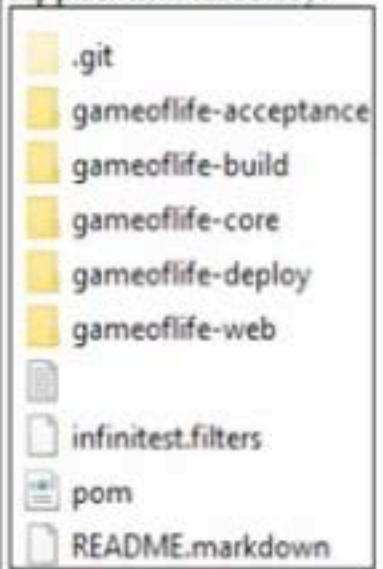
Difference Between SVN and Git



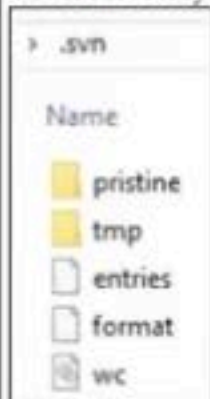
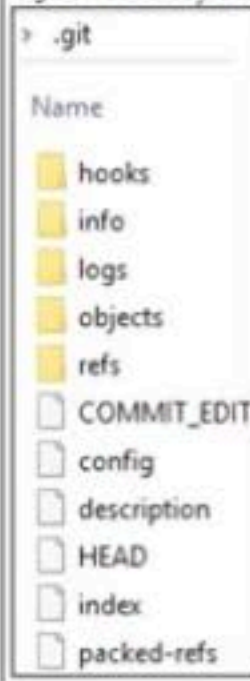
Description of SVN and Git

Subversion	Git
Centralized version control system	Distributed version control system
Snapshot of a specific version of the project is available on the developer's machine	Complete clone of a full-fledged repository is available on the developer's machine
Perform operations such as commit, merge, blame, and revert and verifies branch and log from a central repository	Perform operations such as commit, merge, and blame and verifies branch and log from a local repository, along with pull and push operation to a remote repository if the developer needs to share work with others
URLs are used for trunks, branches, or tags: https://<URL/IP Address>/svn/trunk/AntExample1/	.git is the root of projects, and commands are used to address branches and not URLs: git@github.com:mitesh51/game-of-life.git
<p>An SVN workflow:</p>  <pre>graph TD; A[Active feature implementation is developed within branches subdirectories] --> B[Feature implementation is finished]; B --> C[Featured branch subdirectory is merged into trunk]; C --> D[Featured Branch subdirectory is removed]; D --> E[Trunk-latest stable release of a project];</pre>	<p>A Git workflow:</p>  <pre>graph TD; A[History of all branches and tags within the .git directory] --> B[The latest stable release is available within the master branch]; B --> C[Active feature implementation is developed in separate branch]; C --> D[Featured branch subdirectory is merged into trunk]; D --> E[Featured branch subdirectory is removed];</pre>

Description of SVN and Git

File changes are included in the next commit	File changes have to be marked explicitly and only then are they included in the next commit
Committed work is directly transferred to the central repository, and hence, direct connection to the repository must be available	Committed work is not directly transferred to the remote repository and committed to local repository, and to share it with other developers, we need to push it to the remote repository, in which case we need a connection to the remote repository
Each commit gets ascending revision numbers	Each commit gets commit hashes rather than ascending revision numbers
<p>Application directory:</p> 	<p>Application directory:</p> 

Description of SVN and Git

.svn directory structure:	.git directory structure:
 <p>File explorer view of the .svn directory structure. The root is .svn. It contains a 'Name' column with icons for each item: a yellow folder icon for 'pristine', a yellow folder icon for 'tmp', a document icon for 'entries', a document icon for 'format', and a document icon for 'wc'.</p>	 <p>File explorer view of the .git directory structure. The root is .git. It contains a 'Name' column with icons for each item: yellow folder icons for 'hooks', 'info', 'logs', 'objects', and 'refs'; and document icons for 'COMMIT_EDITMSG', 'config', 'description', 'HEAD', 'index', and 'packed-refs'.</p>
Short learning curve	Long learning curve

Build Tools Maven



Example pom.xml File

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="
http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.springframework.samples</groupId>
  <artifactId>spring-petclinic</artifactId>
  <version>1.0.0-SNAPSHOT</version>

  <name>petclinic</name>
  <packaging>war</packaging>

  <properties>

  <dependencies>

  <!-- Maven plugin versions are mentioned in order to guarantee the build reproducibility in the long term -->
  <build>
    <defaultGoal>install</defaultGoal>
    <testResources>
      <testResource>
        <!-- declared explicitly so Spring config files can be placed next to their corresponding JUnit test class -->
        <directory>${project.basedir}/src/test/java</directory>
      </testResource>
      <testResource>
        <directory>${project.basedir}/src/test/resources</directory>
      </testResource>
    </testResources>
    <plugins>
  </build>
  <reporting>

  <url>demopetclinic</url>
</project>
```

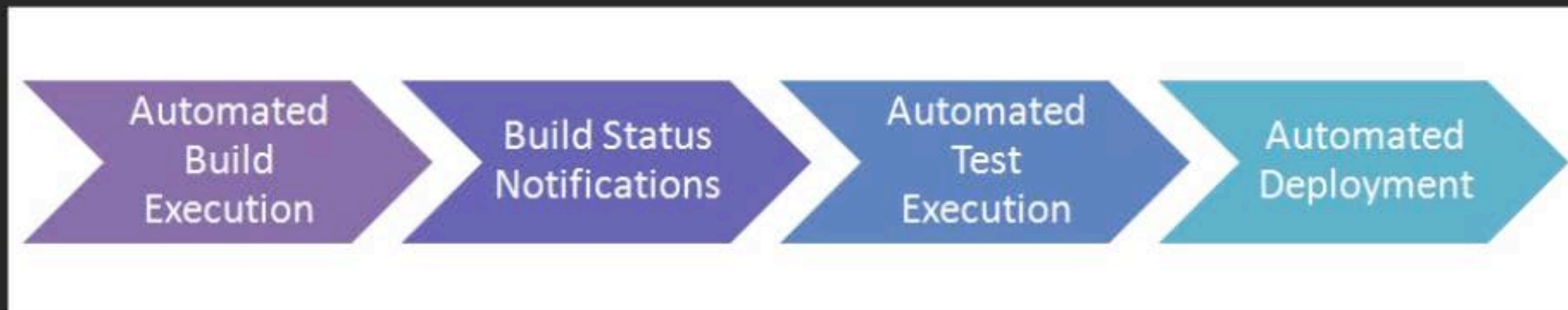

CI Tool - Jenkins

- Jenkins was originally open source continuous integration software written in Java under the MIT License
- Jenkins 2 an open source automation server that focuses on any automation, including continuous integration and continuous delivery
- Jenkins can be used across different platforms, such as Windows, Ubuntu/Debian, Red Hat/Fedora, Mac OS X, openSUSE, and FreeBSD
- It can be used to build freestyle software projects based on Apache Ant and Maven 2/Maven

CI Tool - Jenkins

- There are different kinds of plugins available for customizing Jenkins
- Categories of plugins include source code management, build triggers, build reports, authentication and user management, and cluster management and distributed build

Jenkins – Software Development Process



Key Features and Benefits

- Easy install, upgrade, and configuration
- Supported platforms: Windows, Ubuntu/Debian, Red Hat/Fedora/CentOS, Mac OS X, openSUSE, FreeBSD, OpenBSD, Solaris, and Gentoo
- Manages and controls development lifecycle processes
- Non-Java projects supported by Jenkins: Such as .NET, Ruby, PHP, Drupal, Perl, C++, Node.js, Python, Android, and Scala
- A development methodology of daily integrations verified by automated builds

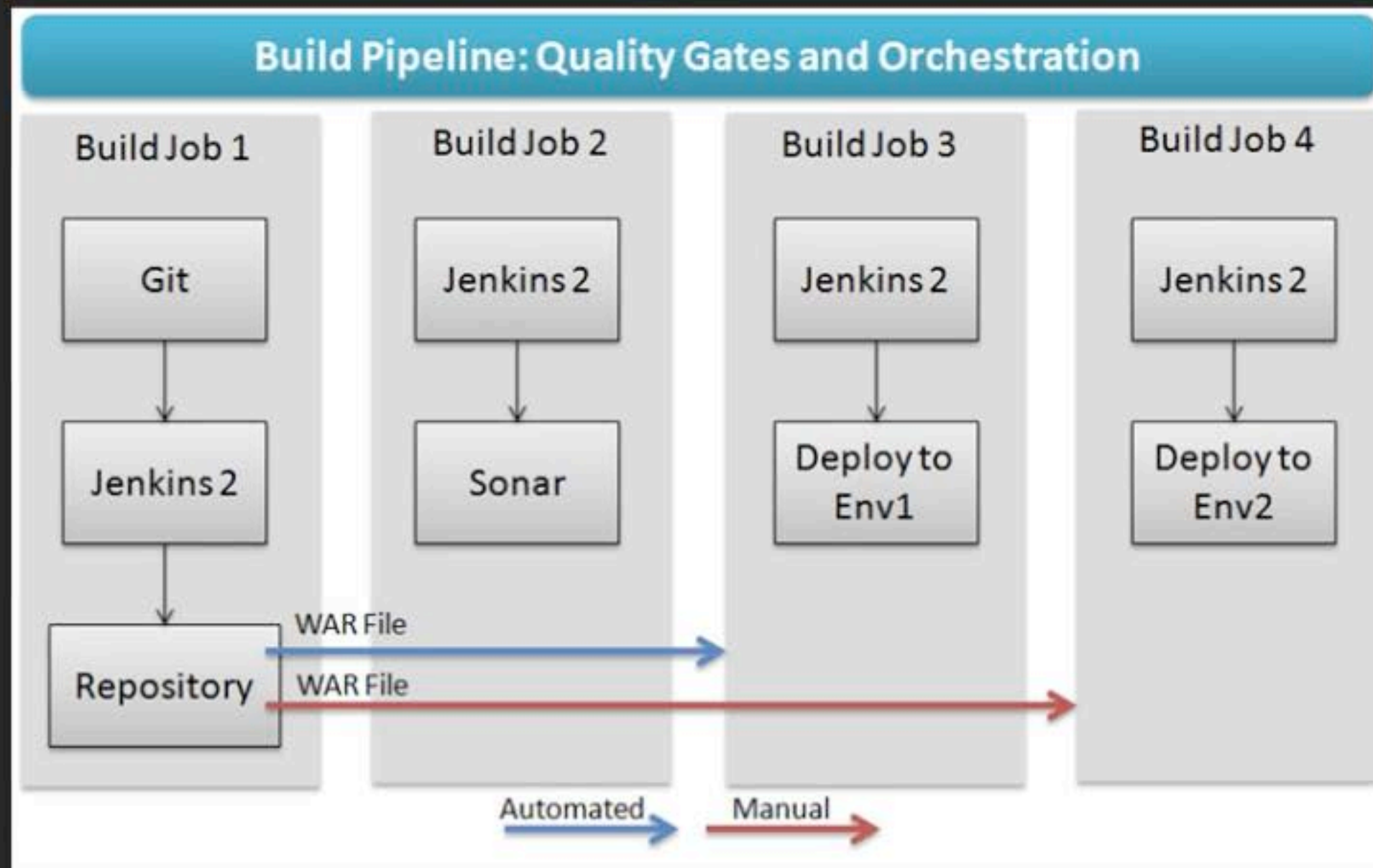
Key Features and Benefits

- Every commit can trigger a build
- Jenkins is a fully featured technology platform that enables users to implement CI and CD
- The use of Jenkins is not limited to CI and CD
- The pipeline as code provides a common language-DSL-to help the development
- New GUI with stage view to observe the progress across the delivery pipeline

Key Features and Benefits

- Supports common test frameworks
- Jenkins supports static code analysis tools
- Highly configurable

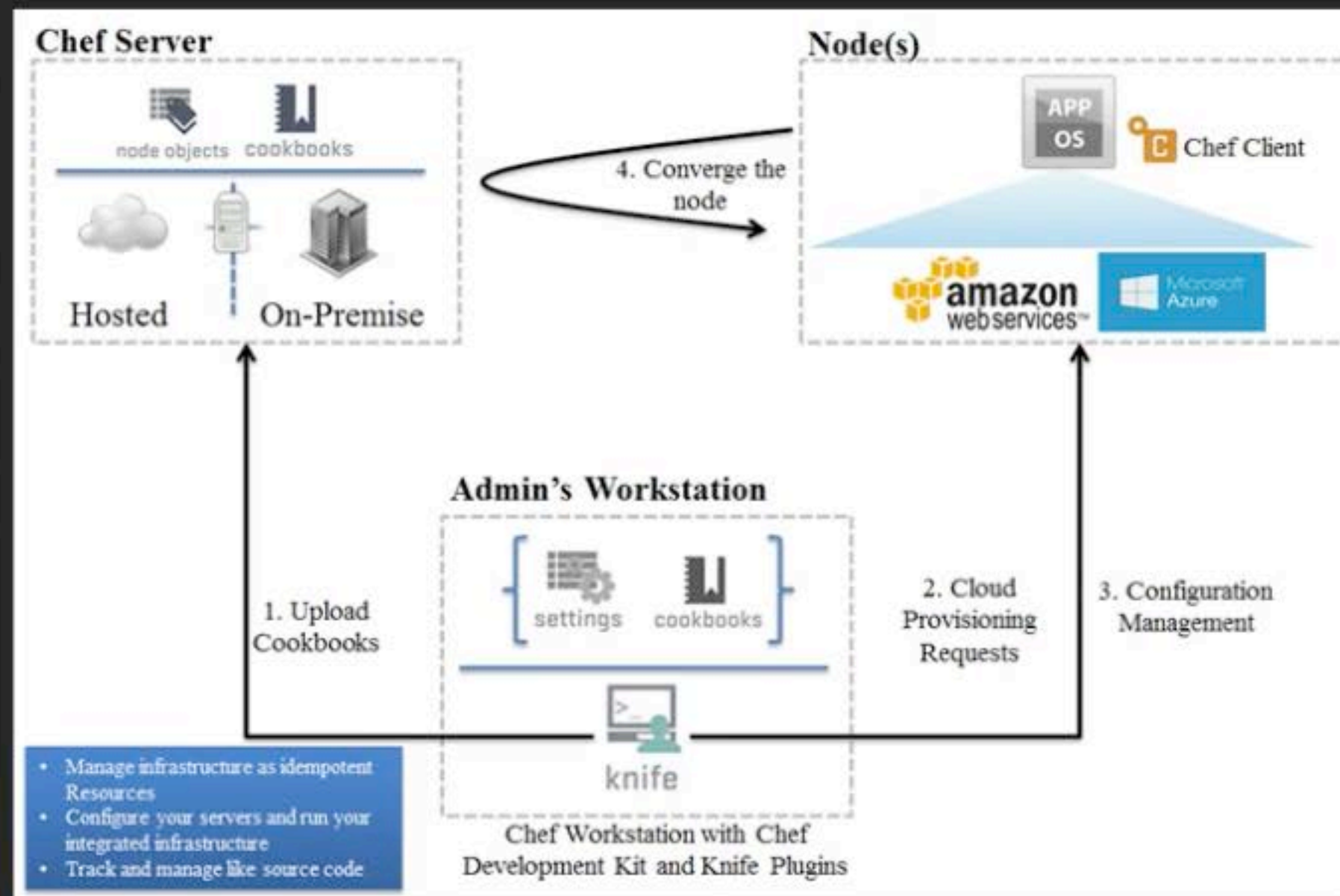
Build Pipeline - Jenkins



Different Categories Tools

Language	Java	.Net
Code repositories	Subversion, Git, CVS, StarTeam	Subversion, Git, CVS, StarTeam
Build tools	Ant, Maven	NAnt, MS Build
Code analysis tools	Sonar, CheckStyle, FindBugs, NCover, Visual Studio Code Metrics, PowerTool	Sonar, CheckStyle, FindBugs, NCover, Visual Studio Code Metrics, PowerTool
Continuous integration	Jenkins	Jenkins
Continuous testing	Jenkins plugins (HP Quality Center 10.00 with the QuickTest Professional add-in, HP Unified Functional Testing 11.5x and 12.0x, HP Service Test 11.20 and 11.50, HP LoadRunner 11.52 and 12.0x, HP Performance Center 12.xx, HP QuickTest Professional 11.00, HP Application Lifecycle Management 11.00, 11.52, and 12.xx, HP ALM Lab Management 11.50, 11.52, and 12.xx, JUnit, MSTest, and VsTest)	Jenkins plugins (HP Quality Center 10.00 with the QuickTest Professional add-in, HP Unified Functional Testing 11.5x and 12.0x, HP Service Test 11.20 and 11.50, HP LoadRunner 11.52 and 12.0x, HP Performance Center 12.xx, HP QuickTest Professional 11.00, HP Application Lifecycle Management 11.00, 11.52, and 12.xx, HP ALM Lab Management 11.50, 11.52, and 12.xx, JUnit, MSTest, and VsTest)
Infrastructure provisioning	Configuration management tool- Chef	Configuration management tool- Chef
Virtualization/cloud service provider	VMware, AWS, Microsoft Azure (IaaS), traditional environment	VMware, AWS, Microsoft Azure (IaaS), traditional environment
Continuous delivery/deployment	Chef/deployment plugin/shell scripting/Powershell scripts/Windows batch commands	Chef/deployment plugin/shell scripting/Powershell scripts/Windows batch commands

Configuration Management – Chef Server



Features – Chef Server

- It manages a huge number of nodes
- It maintains a blueprint of the infrastructure

Features – Chef Client

- It manages various operating systems, such as Linux, Windows, Mac OS, Solaris, and FreeBSD
- It provides integration with cloud providers
- It is easy to manage the containers in a version able, testable, and repeatable way
- Chef provides an automation platform to continuously define, build, and manage cloud infrastructure used for deployment

Features – Chef Client

- It enables resource provisioning and the configuration of resources programmatically
- It will help in the deployment pipeline in order to automate provisioning and configuration

Basic Concepts of Chef

- Achieving the desired state
- Centralized modeling of IT infrastructure
- Resource primitives that serve as building blocks

Cloud Service Providers

- AWS and Microsoft Azure are popular public cloud providers right now
- It provide cloud services in different areas, and both have their strong areas


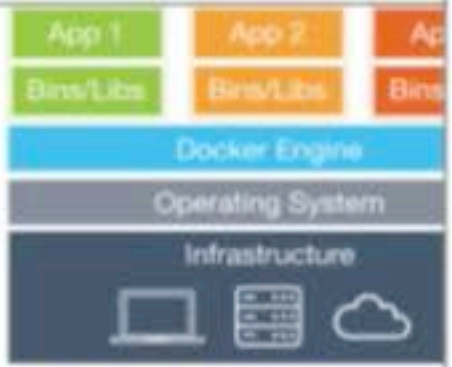
Side by Side Comparison

	AWS	Microsoft Azure
Virtual machines	Amazon EC2	Virtual machine
PaaS	Elastic Beanstalk	Azure Web Apps
Container services	Amazon EC2 Container Services	Azure Container Services
RDBMS	Amazon RDS	Azure SQL Database
NoSQL	DynamoDB	DocumentDB
BIG Data	Amazon EMR	HD Insight
Networking	Amazon VPC	Virtual network
Cache	Amazon ElastiCache	Azure RedisCache
Import/export	Amazon import/export	Azure import/export
Search	Amazon CloudSearch	Azure Search
CDN	CloudFront	Azure CDN
Identity and access management	AWS IAM and Directory Services	Azure Active Directory
Automation	AWS OpsWorks	Azure Automation

Container Technology

- Containers use OS-level virtualization
- Docker and OpenVZ are popular open source example of OS—level virtualization technologies

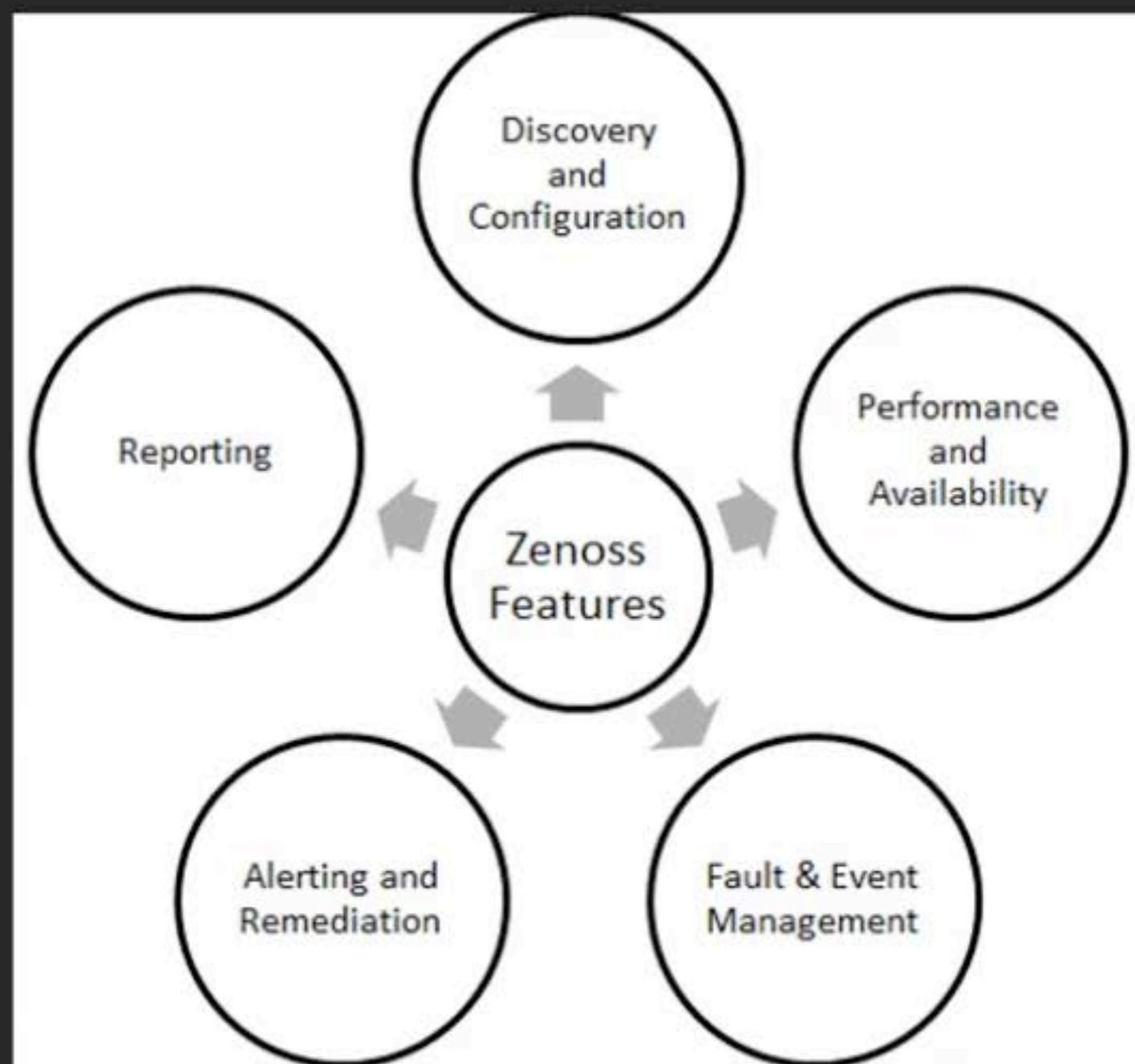
Virtual machine Versus Containers

Virtual machines	Docker containers
 <p>The diagram shows three vertical stacks representing different applications (App 1, App 2, App 3). Each stack contains 'App', 'Bins/Libs', and 'Guest OS' layers. These stacks sit on a 'Hypervisor' layer, which is on top of the 'Host Operating System', which is on top of the 'Infrastructure' layer (represented by icons of a laptop, server, and cloud). A URL http://www.docker.com/ is at the bottom.</p>	 <p>The diagram shows three vertical stacks representing different applications (App 1, App 2, App 3). Each stack contains 'App' and 'Bins/Libs' layers. These stacks sit on a 'Docker Engine' layer, which is on top of the 'Operating System', which is on top of the 'Infrastructure' layer (represented by icons of a laptop, server, and cloud). A URL http://www.docker.com/ is at the bottom.</p>
Virtual machines depend on traditional virtualization. They can be considered hardware-level virtualization.	Containers depends on the containerization technique at the kernel level. They can be considered OS-level virtualization.
Each virtual machine contains a guest OS, binaries, library files, and the application itself.	Each container include application, binaries, and library files, but the major difference compared to virtual machine is the shared kernel. each container runs as an isolated process in the user-space on the host OS.
The size of a virtual machine is in the order of GBs, as each one runs on its own.	As each container runs as an isolated process in the user-space on the host OS and a separate OS is not required for each container, the size of each container is much smaller.
Each virtual machine has its own set of resources, which results in better isolation and less sharing of resources.	Each container shares the kernel, and hence, there's more scope of sharing resources.
A virtual machine cannot run on a container	A container can run on a virtual machine.

Monitoring Tools

- There are many open source tools available for monitoring resources
- Zenoss and Nagios are two of the most popular open source tools

Zenoss



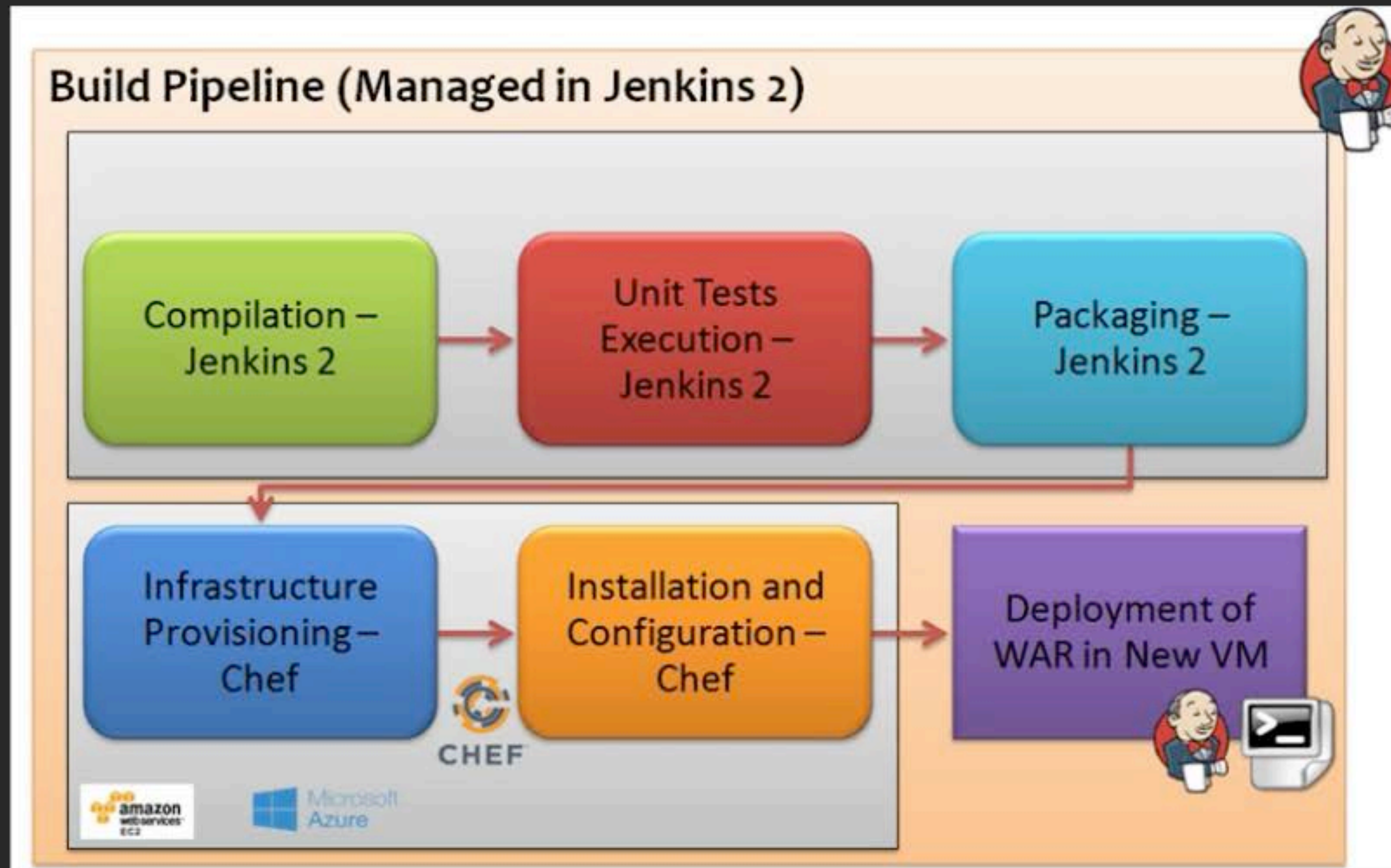
Nagios

- It is a cross-platform and open source monitoring tool for infrastructure and networks
- It monitors network services such as FTP, HTTP, SSH, and SMTP
- It monitors resources, detects problems, and alerts stakeholders
- Nagios can empower organizations and service providers to identify and resolve issues

Continuous Delivery - Jenkins

- Continuous integration: Jenkins
- Configuration management: Chef
- Cloud service providers: AWS, Microsoft Azure
- Container technology: Docker
- Continuous delivery/deployment: SSH

Jenkins Plugins



The DevOps Dashboard



Next Video

An Overview of a Sample Java EE Application