Ansible

Topics

- Understanding Configuration Management and its origins
- The aims of Configuration Management
- Basic principles of Configuration Management
- Configuration Management best practices
- How Ansible simplifies DevOps implementations
- Binary Artifact Management and Ansible

What is Configuration Management ?

Provision infrastructure, deploy code, and maintain the operational consistency of environments.

A way of development where automation can be used to provision and enforce the state, consistency, and accuracy of a set of systems.

What is Configuration Management ?

Hosts can now be automatically configured and provisioned to be in a specific state via code

Packages Installed, Users Created and so on…

The idea that infrastructure can be defined and codified is not only novel but is now the norm

What is Configuration Management ?

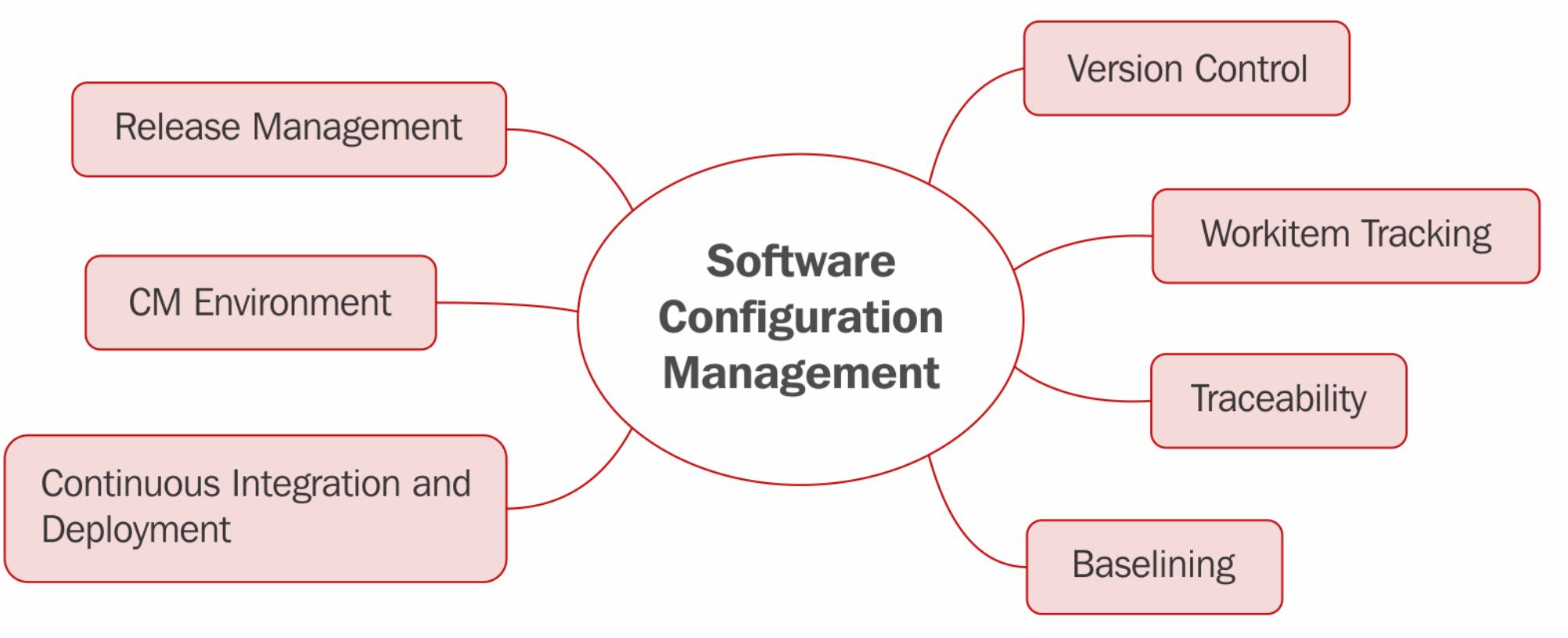
Configuration Management is the **detailed recording** and updating of information that describes an enterprise's **hardware and software**.

Such information typically includes the **versions** and **updates** that have been applied to installed software packages and the **locations** and **network addresses** of hardware devices.

Why Automation is must for companies ?

- Significant amount of pressure to maintaining clean and stateful systems in production
- With the rapid adoption of broadband, mobile devices, SaaS, and internet access, consumers now demand more frequent updates
- Automation is a must for any company wishing to manage distribution channels in order to remain competitive

Core concepts of CM



Core concepts of CM

Release Management

Reliably publish a piece of software for consumption by the target end user.

CM Environment

Represent a set of physical or virtual replications of a production-release environment

Continuous Integration and Deployment

Important set of practices that encourage collaboration and span the entire delivery pipeline.

Core concepts of CM

Version Control

where all source code and development efforts should be stored.

Management scripts, Ansible playbook's, development code, and automated QA tests.

Work item Tracking

Practice of dividing, tracking, developing, testing and deploying large implementation efforts for a project into smaller, more reasonable chunks that can be worked on individually.

Traceability

Traceability of each change to a given software system be available throughout the delivery pipeline.

Core concepts of CM

Baselining

- An infrastructure solution is important because it provides a solid starting point for all future infrastructure implementations.

- This means that at any given time, you can reimage a system with the known good baseline and build on it from there.

- In many ways, this also provides a level of consistency.

Origins of CM

1990s

IBM and the Department of Defense

Effort to track the steps necessary to recreate the system or deployment if the system were to fail or have some kind of fault

The Aims of Configuration Management

If I spend four hours configuring a given development system for a developer to use, and then the developer quits the next day, and a new developer gets hired, I have just wasted eight hours.

Whereas if I spend eight hours writing a single set of automated scripts to automatically provision a development system and the automation takes 20 minutes to run from start to finish, I can now recreate the developer system easily and with minimal fuss and ceremony.

The Aims of Configuration Management

Configuration Management is all about:

saving time,

saving money,

saving resources, and

minimizing waste

Aims of Configuration Management

- To track changes made to a given system or set of systems
- To provide traceability and auditability for defects that may arise as part of a set of changes made to a given system
- To help reduce the amount of manual effort made by developers, QA, and operations folks by maintaining a set of automated solutions that can aid in the provisioning and configuration of a given system
- To provide a level of repeatability to the organization by clearly defining (in automation form) the steps required to build out a given system

Examples of CM and how it could potentially benefit an organization

Scenario 1 :

Bob works in a large IT software organization as a developer. Bob is working on a bug that he needs some help with.

With a proper Configuration Management strategy in place, Bob could easily automate the recreation of his bug environment (the virtual machine used for his local testing and bug recreation efforts) so that both he and his coworker can work on the same bug simultaneously and with minimal fuss in recreating the environment.

Examples of CM and how it could potentially benefit an organization

Scenario 2 :

The development team has been working very hard at completing this month's release. The release includes a number of infrastructure changes and environment spin-up requirements.

As a result of a solid Configuration Management plan and successful automation, the implementation and rollout of this production release will take a matter of minutes instead of the previous month's hours-long rollout.

Examples of CM and how it could potentially benefit an organization

Scenario 3 :

Randy is working hard to help the company's documentation efforts by documenting the recipes of their cloud computing infrastructure.

This documentation involves great care while recreating the individual components of the system to ensure that new hires have the proper information they need to set up a local working copy of the computing infrastructure

Randy has been able to save the numerous hours needed to document this setup solution by simply reading the Configuration Management automation already created and stored in the source control.

Basic Principles of Configuration Management

- Automate where automation is possible
- Provide traceability within the enterprise
- Provide developers, QA, operations, and management with a reproducible infrastructure that is managed through software-development best practices
- Develop a strategy for how new hardware will be provisioned and configured (in an automated way)
- Manage hardware configurations effectively and with strategy
- Develop mechanisms that provide a self-service model for deploying infrastructure changes
- Educate the organization on Configuration Management practices

Configuration Management Best Practices

Some of the more popular tools for Configuration Management

- Ansible
- Chef
- Puppet
- CFEngine

- Most of the tools are open source
- Provide the ways to keep and maintain infrastructure in code form, or IaC (Infrastructure as Code)

Infrastructure as Code

Infrastructure as Code is the process of **managing** and **provisioning computing infrastructure** (processes, bare-metal servers, virtual servers, and so on) and their **configuration** through **machine-processable definition files**, rather than physical hardware configuration or the use of interactive configuration tools.

How Ansible Simplifies DevOps Implementations

The initial implementation of Ansible was derived in 2012 with following principles

- Minimal development required
- Consistent in execution
- Secure
- Scalable
- Highly reliable in nature
- Easy to learn

Users of Ansible

Twitter

Logitech,

NASA,

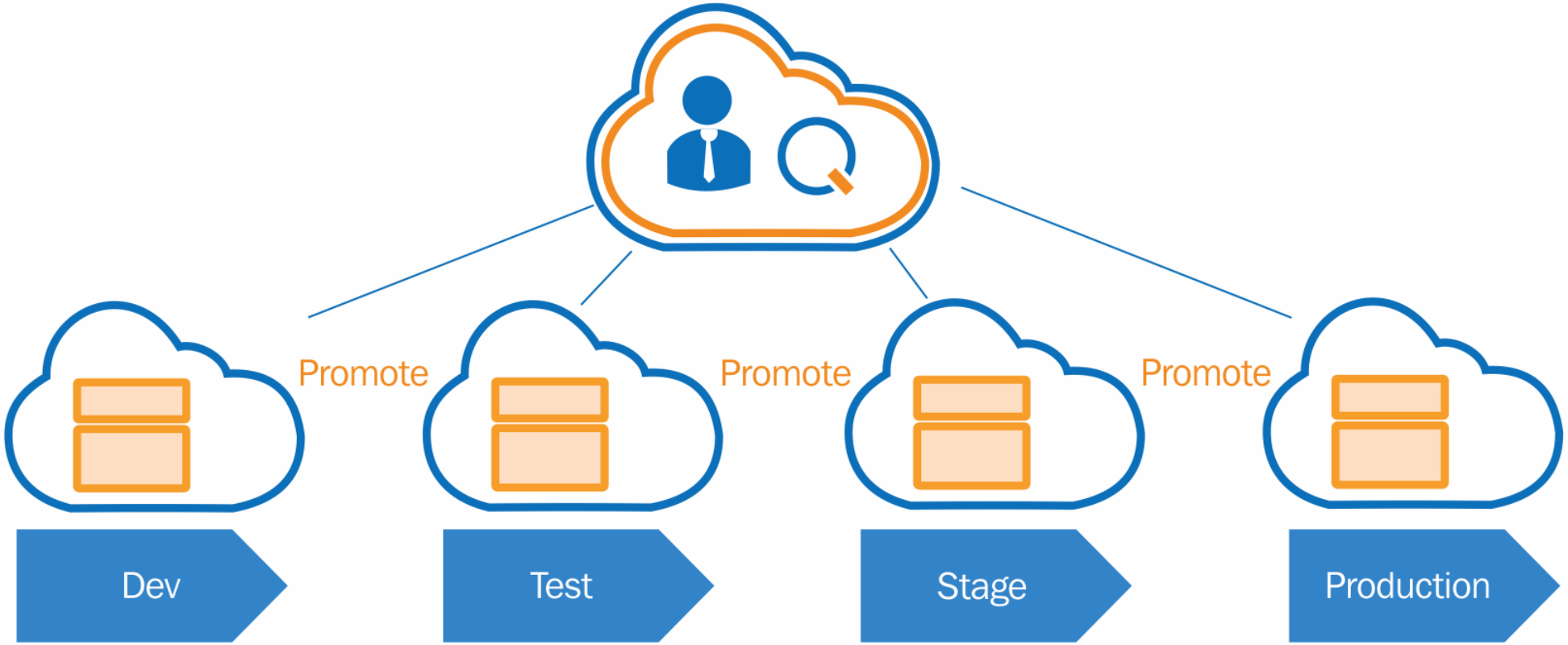
NEC,

Microsoft,

and

hundreds more.

How ansible fits in DevOps envirnment ?



Thank You !