



**The traditional way of doing application and code releases**

The Data Center would be the gate keeper of changes that were made in the DC. Application and developer teams would be need to get any change approved by the DC team, creating a possible bottleneck for delivering features and services

**Now with Continuous Integration and Delivery**

The Data Center is just another member of the delivery team working together with all parties, business, developers, testers, and operators to get features and services to production to deliver **Infrastructure as Code**.

First we begin by developing our code, in our case the code will consist of:

* Puppet Configuration files. These files will drive what each of the puppet client nodes will look like. For the purpose of this lab, the client will be configured with an apache web server deployed in a Docker container.

### Puppet Terms

* **Puppet Master**: the master server that controls configuration on the nodes
* **Puppet Agent Node**: a node controlled by a Puppet Master
* **Manifest**: a file that contains a set of instructions to be executed
* **Resource**: a portion of code that declares an element of the system and how its state should be changed. For instance, to install a package we need to define a package resource and ensure its state is set to "installed"
* **Module**: a collection of manifests and other related files organized in a pre-defined way to facilitate sharing and reusing parts of a provisioning
* **Class**: just like with regular programming languages, classes are used in Puppet to better organize the provisioning and make it easier to reuse portions of the code
* **Facts**: global variables containing information about the system, like network interfaces and operating system
* **Services**: used to trigger service status changes, like restarting or stopping a service
* The Puppet features leveraged for this exercise are :
  + Modules - are self-contained bundles of code and data. These reusable, shareable units of Puppet code are a basic building block for Puppet.
  + Manifest - manifest in a module’s manifests folder should contain one class or defined type. The file names of manifests map predictably to the names of the classes and defined types they contain.
  + Classes
  + Site Wide/Node Configurations
* Source Code Management (SCM) Git/GitHub
  + Configuring locally and remote repositories
  + Initialize local Git repository
    - Create remote repository
    - Connecting local to remote repository – GitHub.com
    - Add code and commit to local repository, then push to remote repository
* Continuous Integration/Delivery (Jenkins)
  + Create a Jenkins project
    - Tie the project to an SCM repository
    - Create steps to be executed for the build of the project
    - Verify integration and deployment
* Configuration Management (Puppet)
  + Review of Puppet topics leveraged in lab (Manifests, Modules, Classes, Site/Node configurations)
  + Verify deployment based upon a change of code in the remote repository

# Git and GitHub Commands

Here we will go over a few of the git commands we will use

git init

Git and GitHub

**git branch testing** #Create a new branch

**git checkout testing** # Switches to the testing branch

**git log --oneline --decorate**

90842c8 (HEAD -> master, origin/master) touching to trigger abuild puppet/modules/my\_apache/manifests/init.pp

99414be Removing the .init files

b2557e3 Update 2 to the site.pp file to have /var/src the / was missing in front of the var as a prerequsite for running the docker\_root

8e03d63 Update to the site.pp file to have /var/src as a prerequsite for running the docker\_root

7f46afa Fixed the puppet build

**git log --oneline --decorate --graph --all**

\* 0c611a3 (origin/testing, testing) addedd logout script lo

\* 90842c8 (HEAD -> master, origin/master) touching to trigger abuild puppet/modules/my\_apache/manifests/init.pp

\* 99414be Removing the .init files

\* b2557e3 Update 2 to the site.pp file to have /var/src the / was missing in front of the var as a prerequsite for running the docker\_root