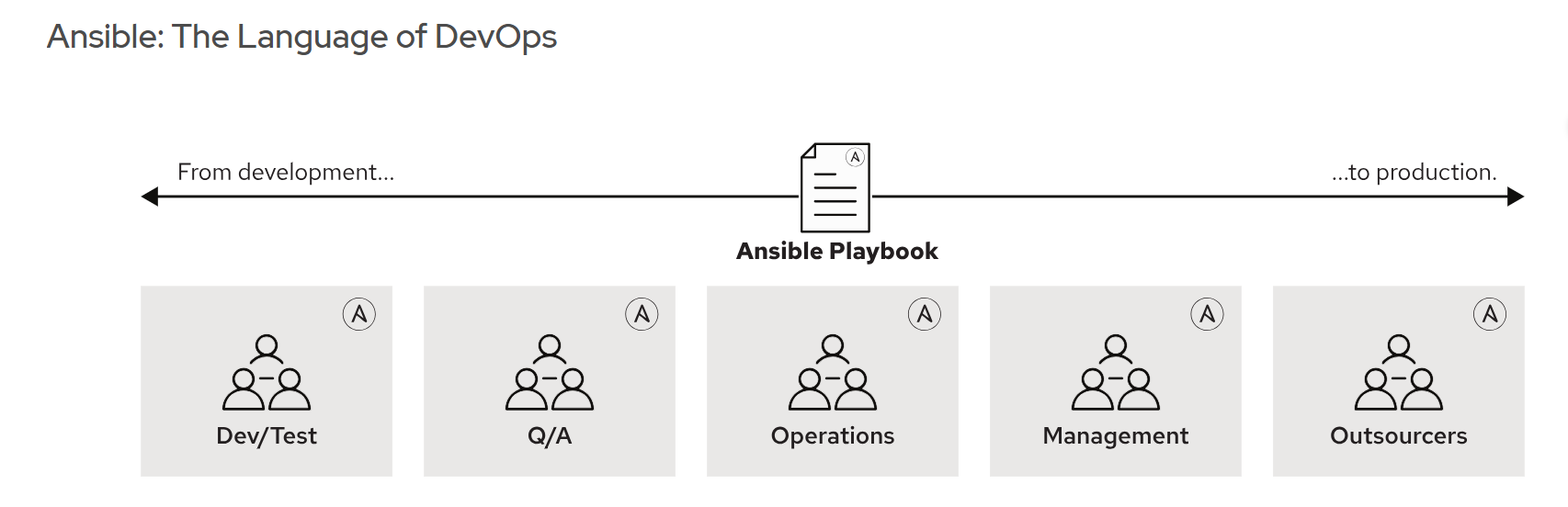
**What is Ansible?**

Ansible is an open-source software provisioning, configuration management, and application deployment tool enabling infrastructure as code. In simple words, Ansible is a free automation tool that can automate IT tasks on a local machine where it is running and on remote machines. For instance, you want to configure http in 5 servers which is easy as you can do it manually on each server but imagine you have 1000 servers to configure that’s when ansible comes in play. You can define ansible in three parts also:

1. **Ansible Is Simple** - Human-readable automation is provided by Ansible Playbooks. The issues brought on by human system administration and infrastructure management can be reduced with automation. System administrators can utilise automation to make sure that all of their systems are deployed and configured promptly and correctly.
2. **Ansible is Powerful** - Ansible can be used to deploy programs for network automation, workflow automation, and configuration management. Ansible can be used to manage the complete lifecycle of an application.
3. **Ansible is Agentless** - Ansible is based on an architecture without agents. Ansible often connects to the servers it administers using OpenSSH or WinRM and executes tasks by pushing out little programs known as Ansible modules to those hosts, though this is not always the case. Since no special agents need to be authorised for use and then deployed to the managed hosts, you can utilise Ansible nearly right away. Ansible is more effective and secure than other options because it lacks agents and any additional customised security architecture.



**Ansible can be used to**

1. **Provision system** - Applications must be installed or deployed on systems. Whether you are PXE booting and kickstarting bare-metal servers or virtual machines, or building virtual machines or cloud instances from templates, Ansible and Automation Controller can help you streamline the provisioning process for systems.
2. **Configure system** - A typical use case for Ansible is centralizing configuration file management and deployment, and this is how many power users are initially exposed to the Ansible automation platform.
3. **Deploy Applications** - Development teams can efficiently manage the full application lifecycle, from development to production, when you specify your application with Ansible and handle the deployment with automation controller.
4. **Orchestration** - Your environment is not defined by configurations alone. You must specify how various configurations work together and make sure that the various components can be managed.
5. **Continuous Delivery** - A CI/CD pipeline needs to be coordinated and approved by many teams. Without a basic automation platform that anyone in your organisation can utilise, you cannot accomplish it. Your applications are appropriately deployed and managed throughout their full lifecycle thanks to Ansible Playbooks.

**History of Ansible**

1. Ansible Project was started by Michael DeHaan in 2012.
2. It is open-source and community driven.
3. It was purchased by Red Hat in 2015.
4. It is available in most of the linux distributions such as Red Hat, Centos, Ubuntu, Fedora, Debian and SUSE

**NOTE - Ansible engine can only be installed in Linux systems. However, you can manage and automate windows system tasks using Ansible.**

**Benefits of Ansible –**

1. It provides pre-written modules.
2. Easy to learn.
3. Great product for orchestration.
4. Ansible can be used not only for systems but also for network, storage, cloud, etc.
5. It provides approx. 1300 modules out of the box and about 4000 modules on galaxy.
6. Huge online ansible resources are available:
   1. [www.ansible.com](http://www.ansible.com)
   2. [www.docs.ansible.com](http://www.docs.ansible.com)
   3. [www.galaxy.ansible.com](http://www.galaxy.ansible.com)

**Terminologies in Ansible**

1. Playbook – It is an automation file in which all the steps are written for the tasks.
2. YAML (Yet another markup language) – A playbook is written in YAML language.
3. Play – It performs a series of *tasks* on the hosts, in the order specified by the play. These plays are expressed in YAML format in a text file.
4. Inventory – A file that has information about the remote hosts that are connected to the Ansible control node and on which servers the tasks are executed.
5. Tag – A reference or alias of a specific task.
6. Variable – It is like containers that hold the defined value which can be used repetitively.
7. Role – Splitting of Playbook into smaller group. After you group your content in roles you can easily reuse them or share them with other users.

**Ansible Concepts and Architecture**

It consists of two types that are control node and managed hosts. Ansible is installed on control node and this machine has copies of ansible project files.

The control node's inventory includes a list of managed hosts. Ansible users build high-level plays to guarantee that a host or collection of hosts is in a specific condition rather than writing intricate scripts. Each job executes a module—a brief bit of code—with particular arguments (written in Python, PowerShell, or another language. In essence, each module is a tool in your toolbox. Hundreds of helpful modules that can handle a wide range of automation tasks are included with Ansible out of the box. They can alter system files, set up applications, or call APIs.

A module usually makes sure that a specific feature of the machine is in a certain state when it is used in a task. One task, for instance, might check if a file exists and has specific rights and content. Plays, tasks, and playbooks are all intended to have equal power. This implies that running a playbook on the same hosts more than once is safe.