Configuration Management Terms and Definitions

work in progress.....

> This document is mainly a technology specific term dictionary for Eliza/HMS Enterprise CM Automation Framework Saltstack Implementation

Product

a service, application, shared library, 3rd party app/lib/service etc...Products often, but not always map 1:1 to 'roles' in configuration management systems. Roles in CM refers to a specific instance configuration or part of an instances configuration. I.E. instances can have multiple roles, aka composite roles

Common Product

products that are not product group specific (3rd party applications for example). These are not 'implemented', but are 'included' within configuration when consumed by any product group. Typically a common product is something that if configured on an instance by itself, would not provide a function. Example JAVA. Java is rarely installed on a system without being required by something else running on the system.

Shared Product

products that are not product group specific (3rd party applications for example). These are 'implemented' when consumed by any product group. Implemented refers to the configuration created by a product group that would reference (or invoke) the shared product configuration with implementation specific key value overrides etc... An example would be... A decision is made that Eliza engineering will support ProductX. Common default configuration is developed to deploy ProductX. TeamA and TeamB (both are product groups) both want to implement ProductX as part of the system each team is deploying. TeamA want to set the version to 1.0.0 and have a non-default install location, and TeamB wants version 2.0.0 and wants to use the default install location, but they also run some other configuration and setup of ProductX after its installed. The core setup of ProductX is contained in the /common/productX config file with appropriate way of allowing each implementation to be modified (version, install location etc...) using Jinja maps. TeamA simply creates a config /TeamX/ProductX that references (invokes) the common ProductX configuration and passes installdir and version parameter arguments. TeamB does the same with it's required configuration. Now each Team that has 'implemented' ProductX has re-used the main configruation of ProductX and added their own use case to their configuration. Thus ProductX is shared.

Team

business classification of function and people. Team members can be spread across multiple product groups if the organization operates that way.

Product Group

implementation specific group of 'products' and a set of people who are responsible for delivering those products. A product group could be made of of members of different teams, could implement delivery of shared products that other product groups deliver, could implement dedicated product group technology not shared with other product groups. Common 'Products' are not owned by a single product group, they are implemented

Group

Same as Product Group, but sometimes group is used in the automation code.

Role

a logical name applied to an entire function set, or a specific function set via configuration that is deployed to one or more instances (nodes, servers machines etc....). A role cannot span multiple instances, but multiple instances can and definitely will have the same role. Instances may also have other roles. Roles are somewhat of a Base configuration in that they can be one of multiple roles in a Composite Role. Roles usually

map to a single Product. But Roles do not have to map to a single product, they can be comprised of multiple products, functions, actions, tasks, configurations etc... Each Product Group can define Roles as they see fit for their particular use cases. However common/shared roles and product group implementations of common/shared roles will all follow the same structure consistency in configuration. Role is a base building block in configuration data. They can be stacked (aka composite roles), re-used (aka product group implementations of common/shared roles), but cannot span more than once instance. For simplicity, sanity and re-usability, it is best practice to create roles at the lowest level. I.E. 1 to 1 mapping of product to role. A 'Role' should be able to perform some function by itself when deployed to a node/instance. There are special cases where a role may not really do much on its own. For example Java. By itself on an instance it doesn't do much. But it can still be implemented as a role. These special cases are TBD by the cloud and product engineering teams.

Server Type

same as Role

Composite Role

a collection Roles defined as a higher level role in configuration deployed to instances/nodes as a single role. Composite roles cannot span multiple instances. However, multiple instance can have the same composite role.

Common Role

See Common Product

Provision

Provisioning is the process of create a bare metal server, or virtual machine in or out of a cloud. However, the exact definition of provisioning can differ from place to place. In the context of our automation framework, provisioning means the process of creating the instance (vm etc..) with initial base OS and networking configuration. In the old days, ping power and pipe. For the purpose of clarity, provisioning does not include deploying software, applying configurations or anything beyond OS and networking.

System

definition coming soon....