Distributed Design and Orchestrating vRealize Automation

This is an extract from the second edition of VMware vRealize Orchestrator Cookbook. Until January 9th 2017, you can purchase it for just $5 from Packt's website.

But more than that, every single eBook and video on their website is $5. So, start your $5 search today and get prepared for tech in 2017.

Anyone interested in vRealize automation will know how important Orchestrator is. And for Orchestrator to work effectively you need to be thinking about distributed design. Let's take a look at a number of different design elements that it's essential to be aware of.

Distributed design

When we talk about "distributed", we're essentially referring to multiple Orchestrator installations that are neither in the same place nor looking after the same things. As an example, your main corporate data center might be located in Europe while you'd perhaps have others in North America and Asia. It might seem a little counterintuitive to think that you have more control with less centralization, but consider it this way: if you were to have one Orchestrator sitting in Europe that manages all other centers, you could be faced with massive problems from other sources - from bandwidth to time zones, even workflow distribution and versioning, your infrastructure will likely cause you a lot of headaches.

But that's not the only example. One can generally differentiate Orchestrator deployments into Geographically Distributed and Logically Distributed ones:

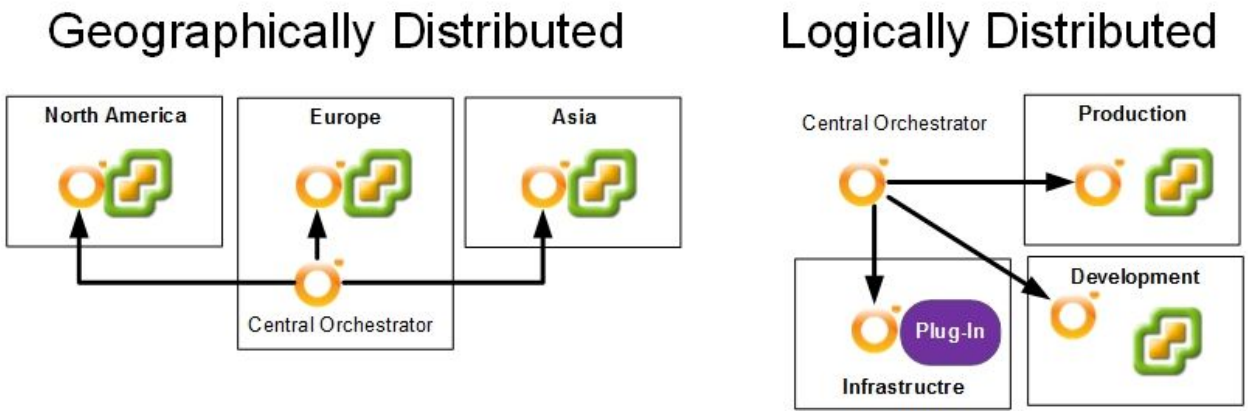
Geographically Distributed

The use of geographically dispersed Orchestrators is common in large companies. Here, a central Orchestrator instance executes workflows on remote environments. The amount of bandwidth used to execute a workflow remotely (using the multi-node plugin) is much less than the amount that would be needed to run the workflows directly. This is especially true when a lot of input variables have to be collected to run the workflow.

Logically Distributed

Logically Distributed means that your Orchestrators are located in different environments, such

as production, development, and so on. In this case, you may have an Orchestrator infrastructure that creates and manages your different infrastructure, or is used for deployments or automation. Central management is then also quite important.

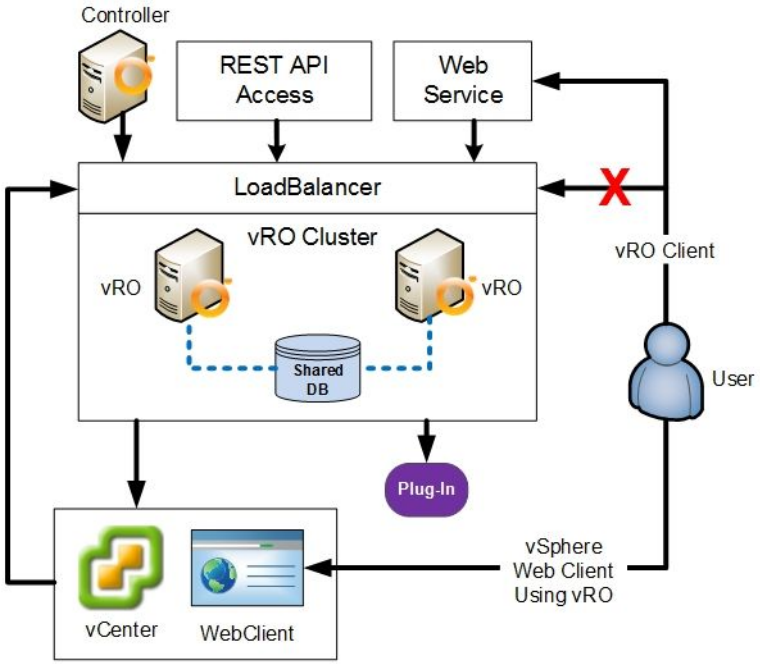


Cluster design

As Orchestrator becomes more and more critical to your infrastructure, it becomes more and more important to cluster Orchestrator to guarantee that it's up and working properly. An Orchestrator cluster is at its most powerful when it is combined with a load-balancer. However, if you are only using Orchestrator to run workflows without any other input (headless), then you can always use an Orchestrator cluster without a load-balancer and ensure that workflows are started or logs are checked using a central Orchestrator that controls all other installations.

A typical situation where a clustered Orchestrator (with a load-balancer) is a very good idea is when the Orchestrator acts as a domain manager. What is meant by that is that Orchestrator is responsible for automating the VMware domain (all things vSphere) and another automation tool (such as Ansible, Chef, or something else) uses the Orchestrator workflows. The domain manager concept is another solution to the automation problem. Instead of using one tool (such as Orchestrator or vRA) to automate everything, you use tools specialized for their domain. Examples of domains are VMware, Microsoft, Red Hat, EMC or NetApp storage, and Cisco networking. In each of these domains, a tool exists that is specialized to deal with the automation of its domain. For Red Hat there is Satellite, for Microsoft there is SMS or SCOM, and so on. Each of these tools has a SOAP or REST interface that can be accessed by a general management tool. Orchestrator would be a domain manager for VMware.

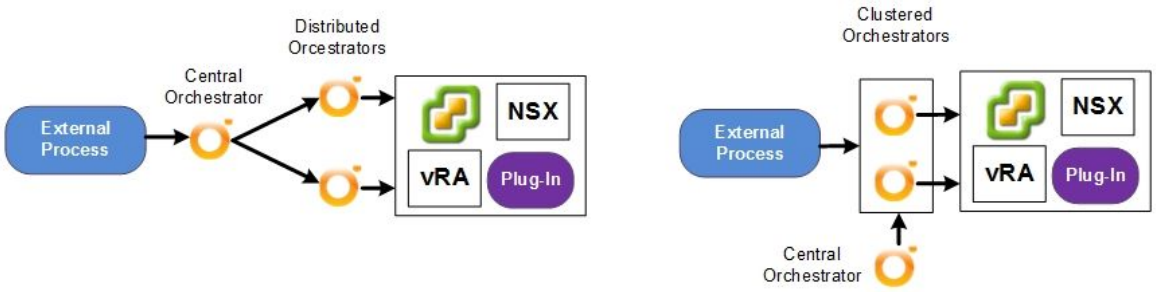
The graphic below demonstrates what an Orchestrator cluster should look and how you can access it. (Please note that the use of the vSphere client isn't supported in cluster mode.)



Scaling out

Sometimes, the number of concurrent workflows you can have running (300) with Orchestrator is simply far too small. One way to deal with this is to increase the limit; however, the smarter option is usually to scale your deployment.

There are two ways you can do this. You either use a distributed design or use a cluster design, as seen in the following figure:



The central Orchestrator in both approaches is responsible for syncing workflows and settings between the actual working Orchestrators.

Central management

A central Orchestrator instance can be used to control all of your distributed installations.

A central Orchestrator server can be connected to all sub-Orchestrators using the multi-node plugin (also known as the VCO plugin). This then allows you to develop and then distribute your workflows centrally.

Using proxy workflows, you can run workflows on geographically remote sites without running into bandwidth or timing problems. You can also schedule the execution of workflows in remote locations to suit time zone differences.

Using the Orchestrator Control Center REST API, you can control the remote/distributed/clustered Orchestrators tidily and even automate their behavior.

In theory, it would be a quite a lot of work, but it's possible to create a workflow that deploys multiple Orchestrator instances using the vSphere plugin, then configure and cluster them using the Control Center API, and then create a load-balancer using the NSX plugin.