

**VMware Solution Design**

for



Prepared by

Scott Wolfram

VMware Professional Services

swolfram@vmware.com

Version History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Ver. | Author | Description | Reviewers |
| 1/21/2021 | .1 | Scott Wolfram | Initial Population |  |
|  |  |  |  |  |

© 2020 VMware, Inc. All rights reserved. This product is protected by U.S. and international copyright and intellectual property laws. This product is covered by one or more patents listed at <http://www.vmware.com/download/patents.html>.

VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions. All other marks and names mentioned herein are the trademarks of their respective companies.

VMware, Inc.  
3401 Hillview Ave  
Palo Alto, CA 94304  
[www.vmware.com](http://www.vmware.com)

# Contents

[Version History 2](#_Toc63930831)

[Contents 3](#_Toc63930832)

[1. Document Overview 4](#_Toc63930833)

[1.1 Executive Summary 4](#_Toc63930834)

[1.2 Conceptual Design 5](#_Toc63930835)

[1.3 Business Requirements, Constraints, Assumptions, and Risks 7](#_Toc63930836)

[1.4 Engagement Background 8](#_Toc63930837)

[1.5 Audience 8](#_Toc63930838)

[2. Solution Logical Design 9](#_Toc63930839)

[3. Technology Component Logical Design 11](#_Toc63930840)

[3.1 Configuration Repository 11](#_Toc63930841)

[3.2 Password Repository 14](#_Toc63930842)

[3.3 Orchestration Engine 14](#_Toc63930843)

[3.4 User Forms 15](#_Toc63930844)

[4. Next Steps 16](#_Toc63930845)

[5. References 17](#_Toc63930846)

# Document Overview

This solution design document is created by VMware Professional Services to support the deployment of the VMware Solution for Wells Fargo WAVE SDDC Automation.

This document is accompanied by other documents that focus on implementation and operation subjects, such as installation, configuration, and fundamental technology.

## Executive Summary

This VMware solution enables the Wells Fargo WAVE SDDC team to customize the VCF stack within their environment to meet compliance rules.

The following table illustrates the IT Capabilities and the IT Problems to be considered and addressed within the scope of this engagement.

Table . IT Capabilities that solve IT Problems

|  |
| --- |
| **IT Capabilities** |
| Automated update certificates for components managed by VCF’s SDDC-Manager (ICE-2944) |
| Automated update ESXi host certificates (ICE-3087) |
| Automated update vCenter solution user certificate (ICE-3074) |
| Automated set ESXi certificate mode to custom (ICE-3095) |
| Automated enable vCenter ESXI dump collector service (ICE-3091) |
| Automated configure SDDC Manager’s vRLI (ICE-3080) |
| Automated configure ESXi’s vRLI (ICE-3076) |
| Automated configure vCenter’s vRLI (ICE-3079) |
| Automated configure ESXi Welcome Screen Banner (ICE-3086) |
| Automated configure vCenter port groups for Horizon and vRealize (ICE-3072) |
| Automated update vCenter VAMI certificate (ICE-3096) |
| Automated configure NSX-T’s vRLI (ICE-3078) |
| Automated enable NSX-T IPv6 functionality (ICE-3082) |
| Automated create NSX-T network segments to support VDI-T guest networks (ICE-3069) |
| Automated create NSX-T network segment profile (ICE-3081) |
| Automated configure NSX-T authentication (ICE-3075) |
| Automated configure vSAN encryption using KMS (ICE-3085) |
| Automated configure vSAN storage policy (ICE-3084) |
| Automated configure ESXi dump collector (ICE-3070) |
| Automated configure vCenter statistics interval and logging level (ICE-3090) |
| Automated configure ESXi shell timeout (ICE-3071) |
| Automated add NFS datastore (ICE-3089) |
| Automated deploy vCenter roles and assign AD groups (ICE-3073) |
| Automated set default SSO domain in vCenter VCSA (ICE-3092) |
| Automated configure vCenter AD identify source (ICE-3093) |
| Automated join vCenter to Active Directory (ICE-3094) |
| Automated vCenter system proxy configuration (ICE-3097) |
| Automated VCF SDDC manager poxy configuration (ICE-3100) |
| Automated configure VCF SDDC manager CEIP (ICE-3099) |

Table . IT Problems in scope

|  |
| --- |
| **IT Problems** |
| Manual process causing service delivery delays |
| New deployments out of compliance |

## Conceptual Design

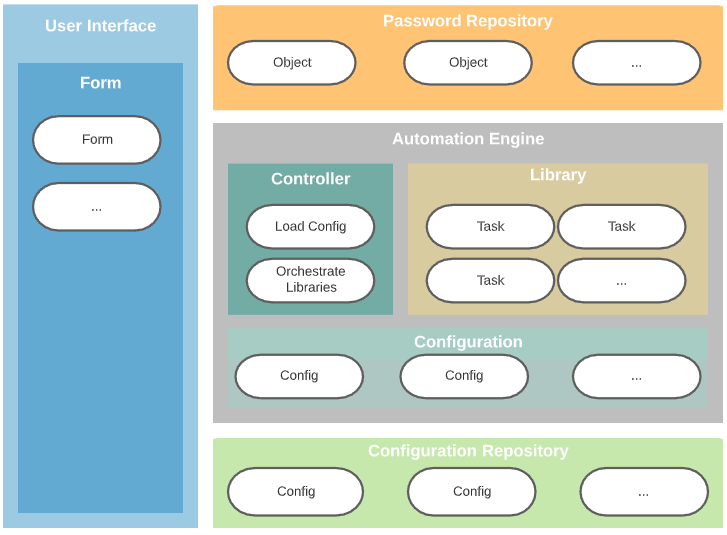
The VMware solution is supported by relevant VMware products and technologies. The key attributes of the solution are:

* Modular libraries written with best in breed languages, that can be driven from a unifying control structure.
* Flexible control structure driven with input from a simple data structure.
* Data structure stored in a simple repository designed to provide traceability.

The following diagram illustrates the logical components of the VMware solution and their relationships. At a high level the system is made up of four components that provide the following:

* User Interface
  + Common interface into the system
  + Security and access controls
* Password Repository
  + Single place for passwords
  + Common secure storage while at rest
* Automation Engine
  + Multi-language programming interface
  + Graphical programming interface
  + Common logging and debugging
* Configuration Repository
  + Common and hierarchal data store
  + Data change management

Figure 1. VMware Solution Conceptual Design



## Business Requirements, Constraints, Assumptions, and Risks

During this engagement, the following business requirements were captured, and technical-specific requirements were discussed and documented along with assumptions, constraints, and risks associated with the solution.

Table . Business Requirements

|  |  |  |
| --- | --- | --- |
| ID | Business Requirements | |
|  | | Compress manual processes from months to days |
|  | | Ensure consistency between environments |
|  | | Ensure consistency between pods |
|  | | Environmental configuration tracking |

Table . Technical Requirements

|  |  |  |
| --- | --- | --- |
| ID | Technical Requirements | |
|  | | Each datacenter should be self sufficient (At Odds with some constraints) |
|  | | Passwords need to be securely stored while at rest |

Table . Constraints

|  |  |  |
| --- | --- | --- |
| ID | Constraints | |
|  | | Must use enterprise Venafi services |
|  | | Must use certified version of VCF |
|  | | Must use certified version of vRealize |
|  | | Must use enterprise KMS servers |
|  | | Must use enterprise GitHub repository |

Table . Assumptions

|  |  |  |
| --- | --- | --- |
| ID | Assumptions | |
|  | | Environment to test code prior to release will be provided |

Table . Risks

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Risks | | Risk Mitigation |
|  | External groups within Wells Fargo not on same timeline as WAVE | Continuously escalate time delays to upper management | |
|  | Automation could ruin an environment, requiring a lengthy rebuild | Proper testing within the VMware lab prior to delivery into the Wells Fargo lab before delivery to more important, shared environments. | |

## 

## Engagement Background

Wells Fargo has engaged the VMware Professional Services Organization (PSO) to design their VMware solution. The design of the components in this engagement aligns to VMware best practices and design guidelines.

## Audience

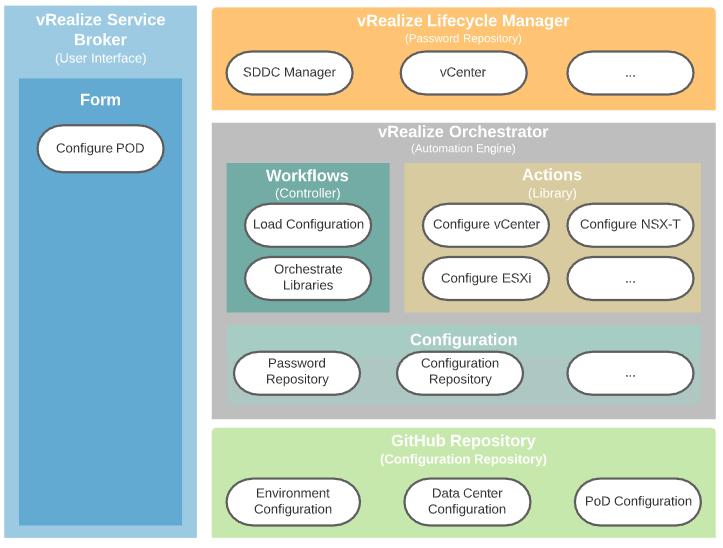
This document is intended for those who plan, design, and implement the VMware solution components. The audience includes the following roles:

* Cloud Architect
* Cloud Service Architect
* Cloud Service Manager
* Director IT Operations
* IT Director
* Network Manager
* Service Administrator

# Solution Logical Design

This section defines the design of a solution from technology perspective. This specific VMware Solution design includes the following technologies:

* User Interface - VMware vRealize Service Broker
* Password Repository – VMware vRealize Lifecycle Manager
* Automation Engine – VMware vRealize Orchestrator
* Configuration Repository – GitHub Enterprise



The table below described the tested software version numbers and build number where applicable.

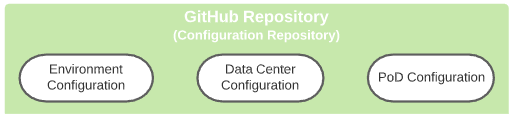
Table 8 Technology Components

|  |  |
| --- | --- |
| **Technology Components** | **Version** |
| vRealize Service Broker | 8.2.0 (16941495) |
| vRealize Orchestrator | 8.2.0 (16941495) |
| vRealize Lifecycle Manager | 3.0.1 (16404613) |
| GitHub Enterprise Server | 2.21.12 |

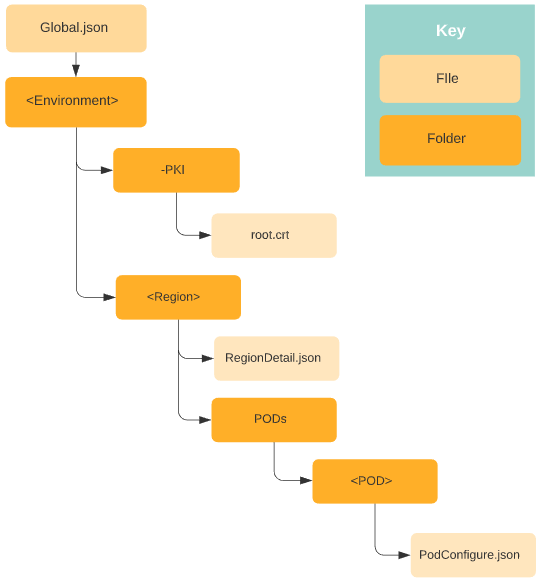
# Technology Component Logical Design

In this section, we go through the logical design of each component included in this engagement. The document begins with the components at the lowest layer of the stack and works methodically up the stack of technology components.

## Configuration Repository



The configuration repository will reside inside a GitHub project repository. This will provide effectively a no SQL database of the various parameters needed to configure each environment. Being a code management repository GitHub will also provide traceability and versioning of the configuration being applied to environments.



The diagram above visually describes the folder and file structure. At the root level there is a file “Global.json” in JSON format that describe the parameters applied to every WAVE environment regardless of which environment they are in. The global configuration will contain the following

* Certificate Details
  + vRealize Lifecycle Manager locker for the username / password
  + Venafi hostname and credentials
  + Certificate Organization, Organization Unit, Country, and email
* SDDC Manager certificate details
  + Certificate Key Size and algorithm
* vCenter details
  + Logging Directory on the vCenter Directory for Solution User logs
  + VM Folder for VDI desktop pools
  + VM Folder for boot devices
* ESXi
  + Shell login welcome banner
  + The kernel vNic for Core Dumps
  + ESXI Shell and Interactive Timeout

Below the root level is each environment such as “Lab”, “Dev”, “UAT”, and “Prod”. Each environment will contain a folder, “-PKI”, containing the root certificates to trust for the environment. The “-“ in front of the name is used to indicate the UI code should not display this folder as will be discussed later. The environment folder will also contain a folder with the name of the region such as “Silas”, “Oxmoor”, “Tempe”, and “Shoreview”.

Inside each region folder there will be a “RegionDetail.json” file in JSON format describing the parameters to be applied to all pods within the region. The region details will contain the following:

* Certificate details
  + Root Certificate location inside the GitHub repository
  + Certificate City, State, and Locality
* Active Directory details
  + vRealize Lifecycle Manager locker for the username / password
  + Domain FQDN
  + Base DN
  + Organization Unit for computer accounts
* vRealize Lifecycle Manager Hostname
  + Note: The credential must be stored in the vRO configuration
* vCenter
  + KMS Servers details
    - vRealize Lifecycle Manager locker for the username / password
    - Signing provider name
    - KMS hostnames, ports
  + vSAN fault tolerance settings
  + Syslog details
  + Proxy host settings
  + NFS Datastore details
  + vCenter role definitions
* vRealize Log Insight
  + vRealize Lifecycle Manager locker for the username / password
  + Server hostname
* vRealize Identity Manager
  + vRealize Lifecycle Manager locker for the username / password
  + Server hostname

Each region folder will contain a PODs folder containing a folder for each pod with a file “PodConfiguraion.json” in JSON format containing the parameters to be applied to the pod. The POD details will contain the following:

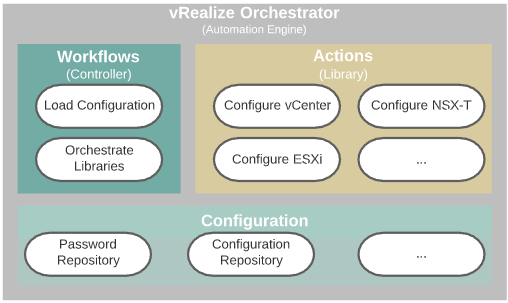
* ID of the pod
* SDDC Manager
  + Hostname
  + Venafi policy DN to store certificates
* vCenter Details
  + Core dump network port
  + Venafi policy DN to store certificates
* ESXi Details
  + Venafi policy DN to store certificates
* NSX-T Details
  + If an external load balancer exists
  + If vIDM should be enabled
  + LDAP user and groups to grant permissions to
  + VDI Segments to create
* List of integrations to execute for the POD

## Password Repository



In order to secure the various passwords needed to configure various systems, the password locker within the vRealize LifeCycle Manager will be utilized. Accounts that are referenced in the configuration repository will be retrieved from the locker feature of vRLCM.

## Orchestration Engine

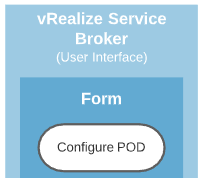


The orchestration engine, vRealize Orchestrator, is where the majority of function happens. vRealize Orchestrator can be described in sections of the system that work together to provide automation. In this case we’ll utilize actions, workflows, and configuration elements to provide the required functions.

Action are basically code snippets organized into containers called modules that provide function. We will create a library of functions to support the IT capabilities described in section 1.

We will then utilize a workflow, which provides a visual representation of code, to load the configuration from the configuration repository and orchestrate the library of calls. The workflow will need to know where the password and configuration repository are located which will be stored in vRealize Orchestrator configuration objects.

## User Forms



vRealize Service Broker will be used to provide a user Interface to present the vRealize Orchestrator Workflow. The user will be presented with a list of Environments, Regions, and PODs derived from the configuration database folder structure.

# Next Steps

The implementation of the VMware solution will enhance the ability to provide and manage access to public and private cloud resources and realize the following outcomes:

* Extend automation tasks beyond minimum viable product by working to organize the silos and expand on capabilities to reduces provisioning and increase quality.
* Refine the code and process as lessons are learned to increase stability and provisioning time.

# References

For more information, see the following supplemental configuration and administration guides, white papers, and best practices documents.

* vRealize Orchestrator Product Documentation
  + <https://docs.vmware.com/en/vRealize-Orchestrator/8.2/com.vmware.vrealize.orchestrator-dev.doc/GUID-B5C0EE02-0E6B-4625-826C-47CD5323488B.html>
* vRealize Automation Product Documentation
  + <https://docs.vmware.com/en/vRealize-Automation/index.html>
* Wells Fargo VCF Configuration Wiki
  + <https://wiki.cloud.wellsfargo.net/display/VE/VCF+Configurations>
* vRealize Orchestrator Coding Design Guide
  + <https://docs.vmware.com/en/vRealize-Orchestrator/7.4/vrealize_orchestrator_coding_design_guide.pdf>