Hewlett Packard Enterprise

Codar

Software version: 1.70

For Microsoft Windows® and Linux operating systems

Secure Application Deployment Using Topology Composition

Document release date: August 2016

Software release date: July 2016

Contents

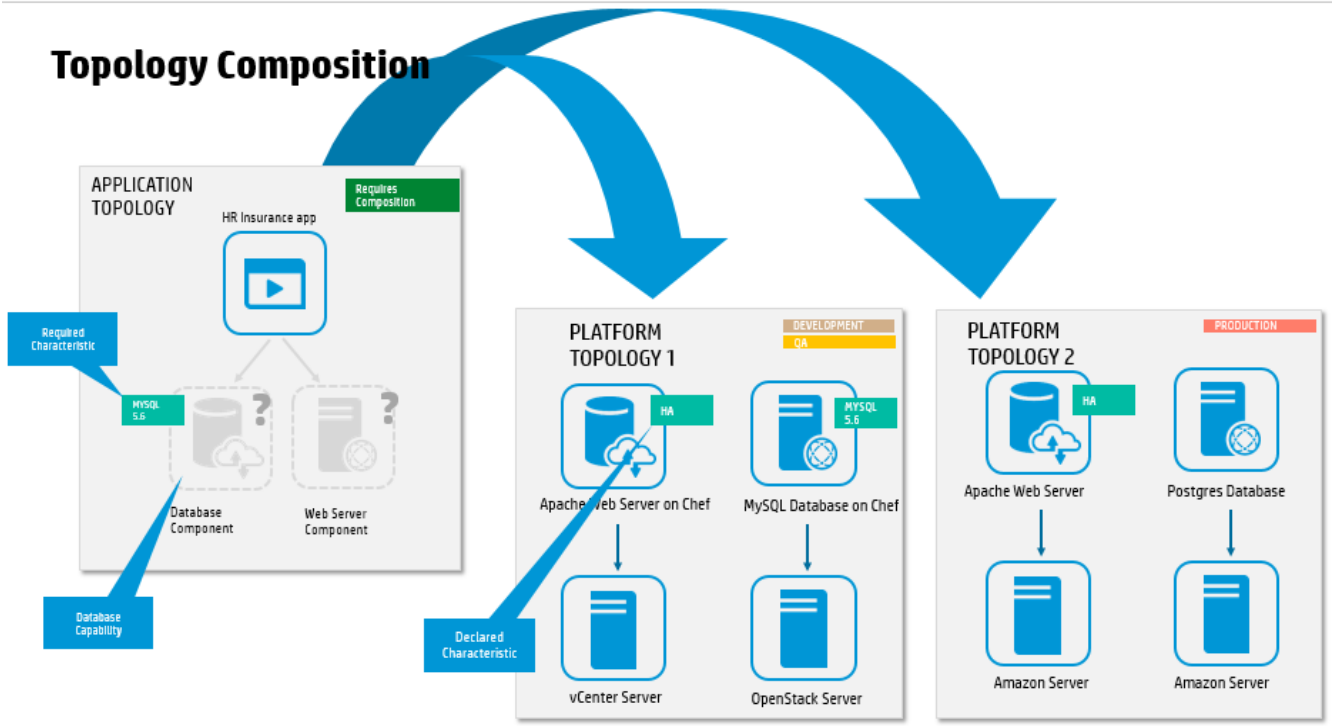
Introduction

The HPE Codar resource provider provides integrations with multiple infrastructure vendors such as VMWare, Amazon Web Services, HPE Helion, and so on. These vendors are used to deploy middleware software such as application servers, database servers, web servers etc. Each of these infrastructure vendors will be used in different lifecycle environments based on the requirement, feasibility and advantage.

Codar provides modelling capability to deploy these middleware software applications on the dedicated infrastructure across different environments. This modeling capability comes from different resources such as HP Operations Orchestration workflow, Opscode Chef Cookbooks, HP Server Automation software policies, etc. An application design includes many components that are catered to provision the infrastructure and install software and application. It is most important to create infrastructure based on the lifecycle environment to avoid misuse of virtual machines by different persona who are part of different teams such as development, testing, and release management.

Topology Composition, a feature introduced in Codar, provides a sophisticated and secure way to deploy applications on elevated and non-elevated environments thus allowing end users to select the infrastructure meant to be used only by them. This proactively secures the elevated environment. This technical documentation helps end users understand this feature based on use cases described and implement the feature based on the samples provided.

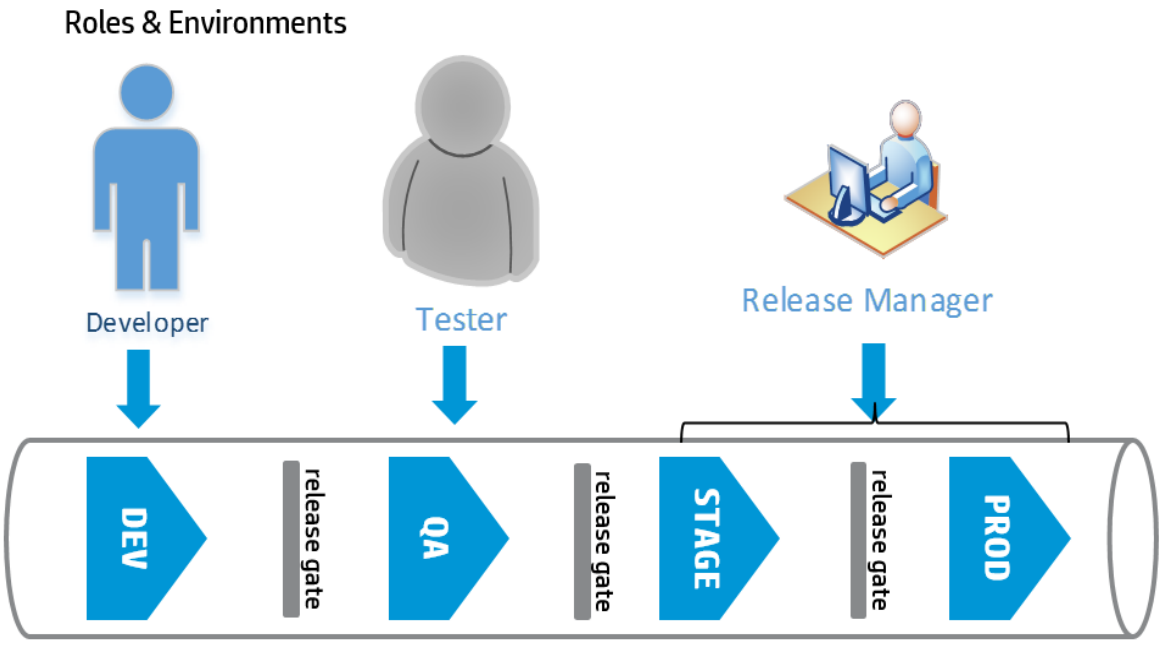
Figure1: Topology Composition



Roles and environments

There are some OOTB roles in Codar like: developer, tester, application release manager, and application architect. Three of these roles (developer, tester, and application release manager) can be associated with different lifecycle environments such as development, testing, staging, and production. Each of these personas can deploy the application to verify and validate whether the application functions as expected and whether the same build version can be promoted to the next stage once it is verified.

Figure2: Roles and environment association

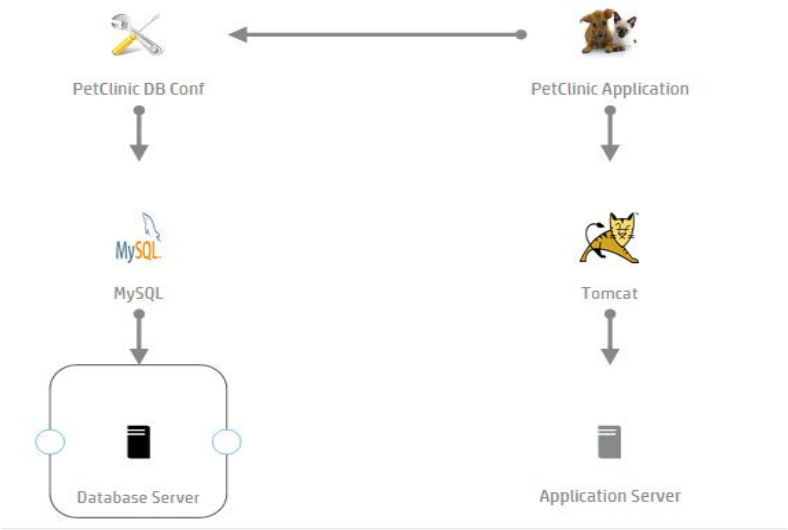


An application architect's role is to create application and infrastructure designs in order to allow other personas to deploy the application on the infrastructure associated with the respective lifecycle environment.

Illustrative Example

Consider a three tier application that must be tested across different lifecycle environments by different personas and finally deployed in production. The following example is of a sample application called PetClinic that must be deployed and tested by various roles across development, test, staging, and production environments. This is a three tier application that has an infrastructure layer, middleware layer, and an application layer. The infrastructure can either be provisioned from VMware or can be existing infrastructure or can be from public cloud vendors such as Amazon Web Services or HP Helion based on the requirement. The middleware layer installs Tomcat and MySQL server. The last layer is the application layer that contains components to configure the MySQL database required for this application and to deploy the "petclinic.war" application on the Tomcat server.

Figure3: Sample PetClinic Application Design



Partial Design

The application design shown in Figure3 is a complete design that contains middleware components and the infrastructure component. This design can be used to provision new virtual machines, install middleware software, and deploy the application. In the core of this application design are two components: “PetClinic DB Conf” and “PetClinic Application”. These two components help the user deploy the PetClinic application on the required infrastructure. These two components can be modeled in a separate application design and can be used to set a relationship, which in Codar terminology is called a *partial design*.

Capability components

What are capabilities?

Codar brings in some of the out-of-the-box components that are used to associate a component that are capable of exposing certain characteristics. A characteristic of a component is its capability to install an application server, database server, or a web server. The component can be associated with the out-of-the-box Capabilities tab by adding a supported capability.

A partial design can have these capabilities components set as a relationship with the core components in order to deploy the application on the required infrastructure by selecting the filtered infrastructure during deployment time. This is explained in detail using the following use cases.

Figure 4: Out-of-the-box capabilities components

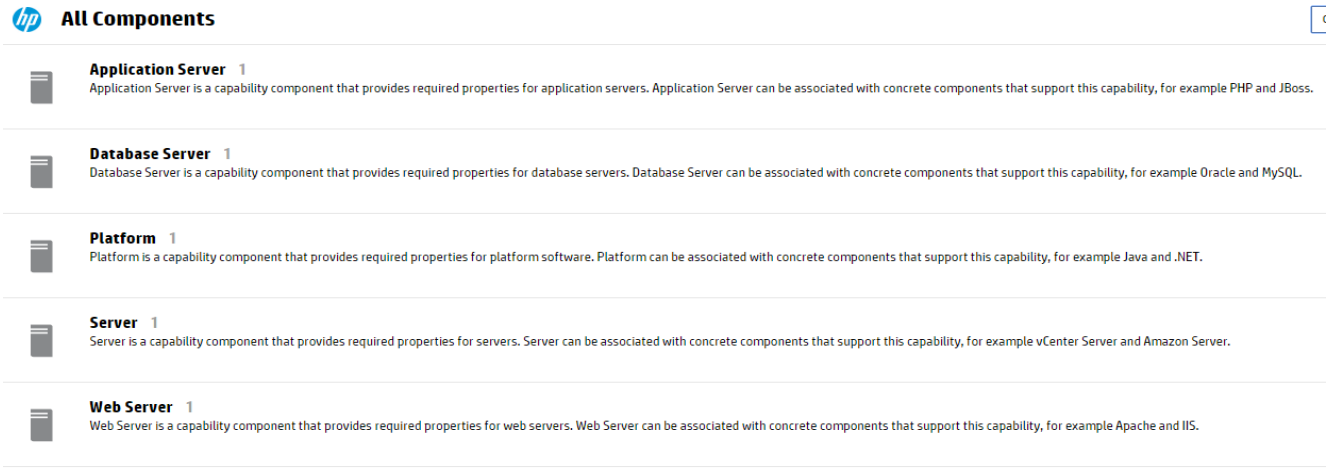
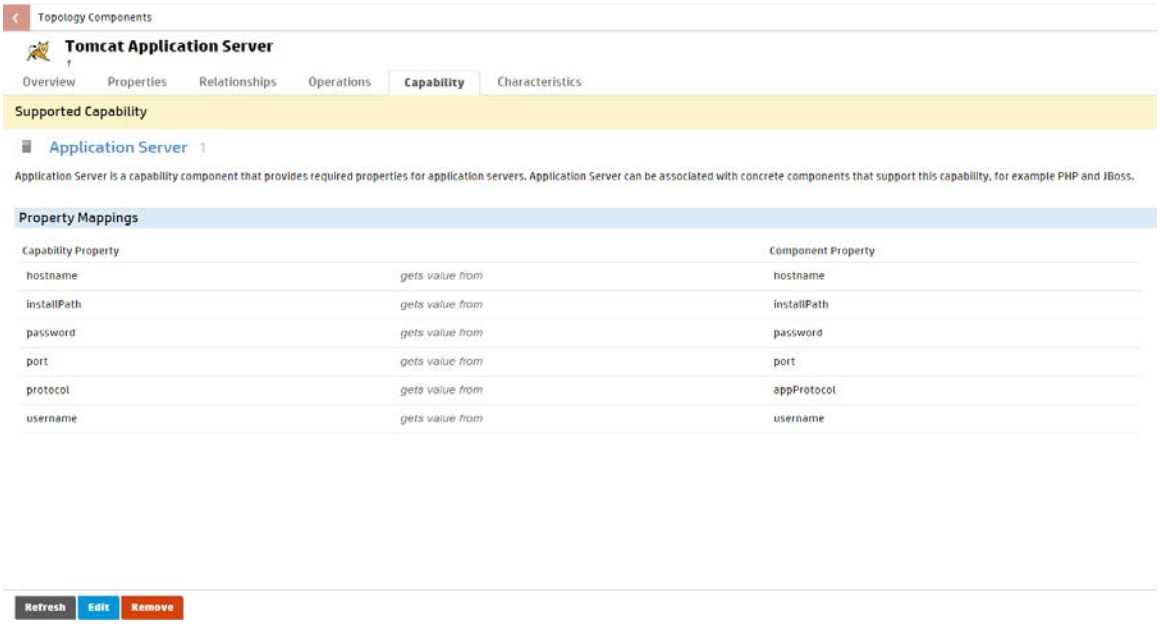


Figure 4.1: An application server capability is set for the component that can install the Tomcat server



Use Case 1

Assume that the sample PetClinic application can be deployed in combination with multiple application servers such as Tomcat, JBoss, Oracle WebLogic etc. and databases such as MySQL, Oracle, MSSQL etc. The QA team may need to certify this application across these servers. To certify the application on these server, all these components need not be a part of the same application design. The application architect may create a partial design as shown in Figure 5 and create infrastructure designs as show in Figure 5.1.

Figure 5: A partial PetClinic application design

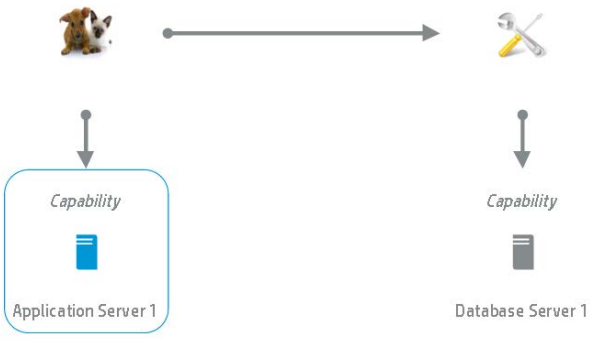


Figure 5.1: Infrastructure designs to certify on multiple servers Tomcat/MySQL and JBoss/MSSQL

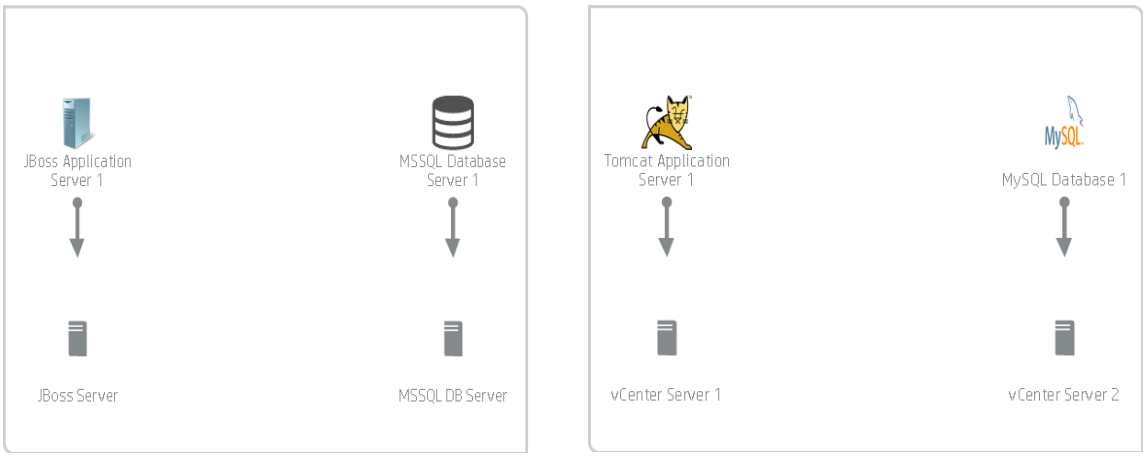
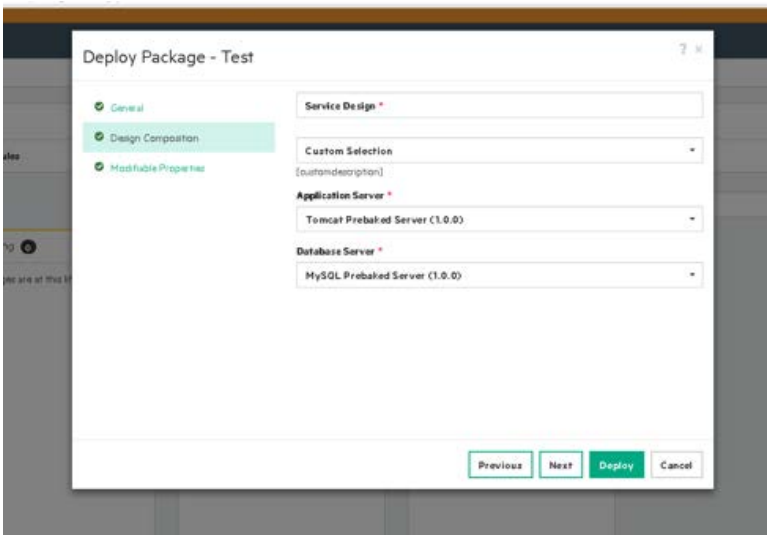


Figure 5.2: Infrastructure designs filtered to list the matched capabilities based on partial design



Benefits

When an application QA engineer deploys the application using partial design, the UI lists two infrastructure designs as shown in Figure 5.1. As per the characteristics of this feature, the UI can list all the infrastructure designs that contain the application and database servers and those that satisfy or match the partial design capabilities as shown in Figure 5.2. This figure shows an additional infrastructure design option: “Two Tier Infrastructure on AWS-Weblogic/Oracle”.

Without this feature, the application QA engineer will be given a list of all the available infrastructure designs that may not match the application design and users may need to search for the matching infrastructure on which they want to deploy the application. Using this feature avoids this complexity**.**

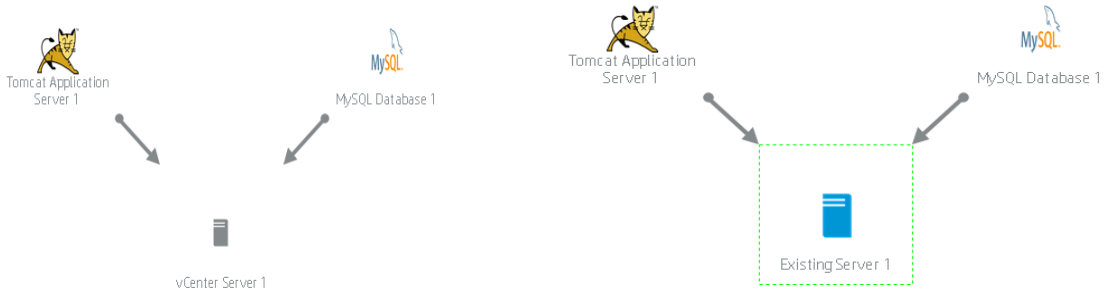
Use Case 2

The second advantage of the topology composition feature in Codar is to prevent the user from deploying the application on the infrastructure design not meant to be used by a persona other than the one it is designed for. For example, an application developer or the application QA team must not deploy the application on the infrastructure design meant for the stage and production environments. An application release manager must not wrongly choose the infrastructure design designated only for development and testing purposes.

Application developer

Consider a developer who wants to deploy the application and execute some unit tests. The developer’s team may not have Tomcat and MySQL on two different servers. They may choose to install them in a single server that can be from VMware or they may use existing infrastructure. In this case, the application architect will create an infrastructure design similar to the following designs.

Figure 6: Single-tier VMware infrastructure **Figure 6.1**: Single-tier existing infrastructure



Application QA

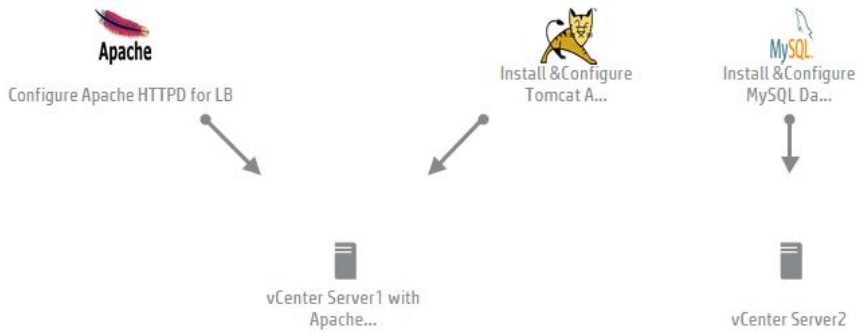
An application QA engineer may deploy the same application to test on multiple infrastructures that may contain Tomcat and MySQL servers, JBoss and MSSQL servers, and Oracle WebLogic and Oracle servers. The application architect can create multiple designs to support these combinations. One of the combination can be created as shown in Figure 7. You can also see Figure 5.1 for more infrastructure designs.

Figure 7: Infrastructure design that comprises Tomcat and MySQL

Application release manager

An application release manager may deploy the same application on a complex infrastructure that involves a load balancer configuration. Typically, in the stage and production environments, a load balancer is available to balance the requests. The application architect may need create an infrastructure design that contains an additional component to configure a software load balancer on the same Tomcat server as shown in Figure 8.

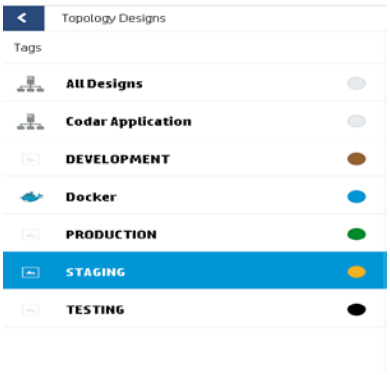
Figure 8: Infrastructure design that contains Tomcat, MySQL, and Apache2 for load balancer



All of the above infrastructure designs must be associated with the following tags. The tags are case-sensitive.

- Development environment infrastructure designs must be associated with a tag named ‘DEVELOPMENT’.
- Test environment infrastructure designs must be associated with a tag named ‘TESTING’.
- Stage environment infrastructure designs must be associated with a tag named ‘STAGING’.
- Production environment infrastructure designs must be associated with a tag named ‘PRODUCTION’.

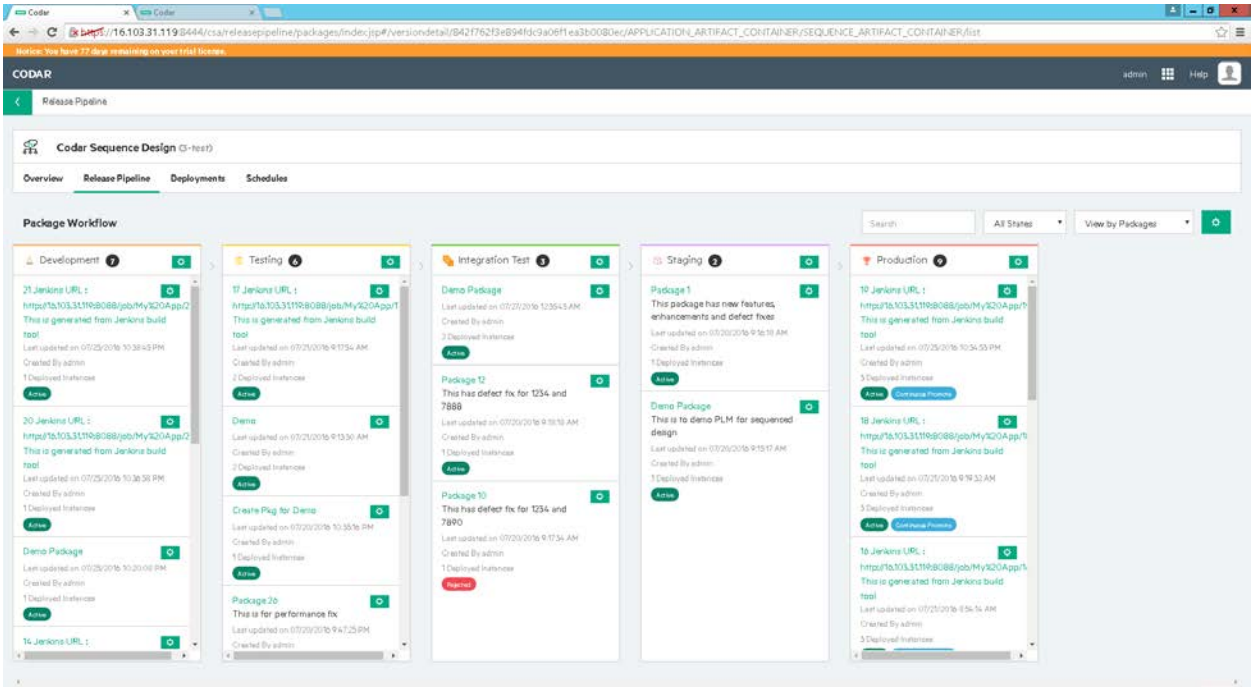
Figure 9: Tags for different environments



Benefits

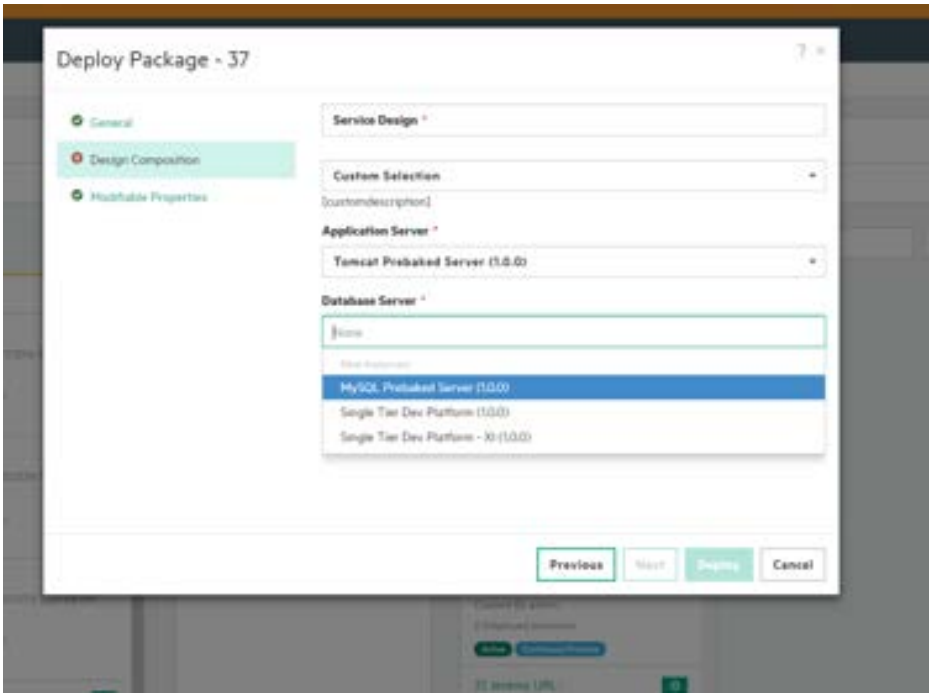
At the time of deploying an application, a developer is given an option to choose the infrastructure design that is created for the development environment, a tester is given an option to choose the infrastructure design that is created for the test environment, and an application release manager is given an option to choose between the stage and production environments**.**

Figure 10: Packages available across different lifecycle stages



When a developer deploys the application, only the infrastructure design created for the development environment is listed as shown in Figure 11.

Figure 11: Infrastructure designs listed for developers deploying on a development environment



Similarly, when the QA engineer and release manager try to deploy the application, infrastructure designs created in the test, and stage and production environments respectively are listed for these roles**.**

Figure 12: Infrastructure designs listed for a tester deploying on a test environment

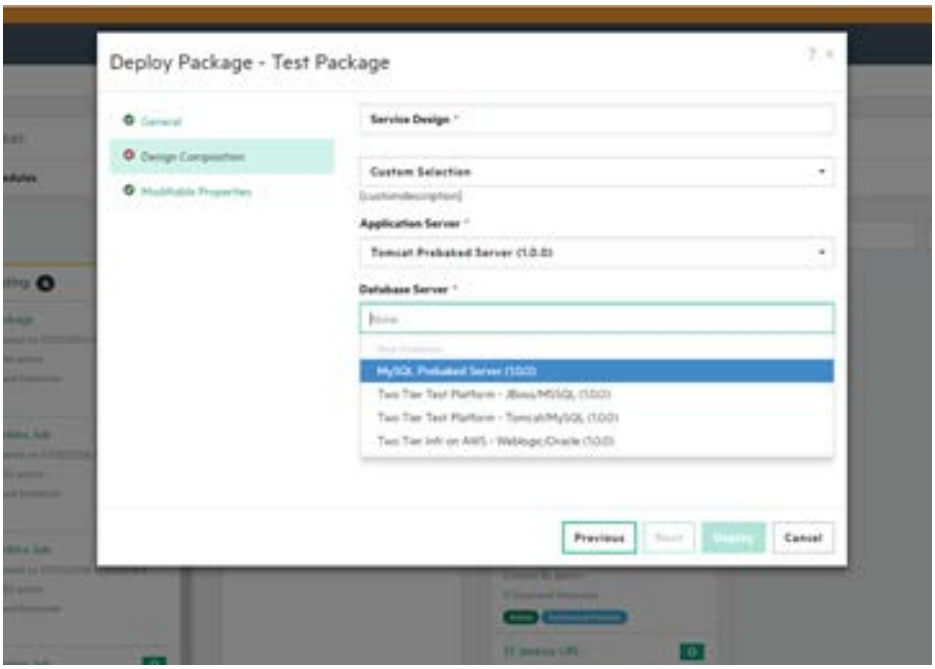
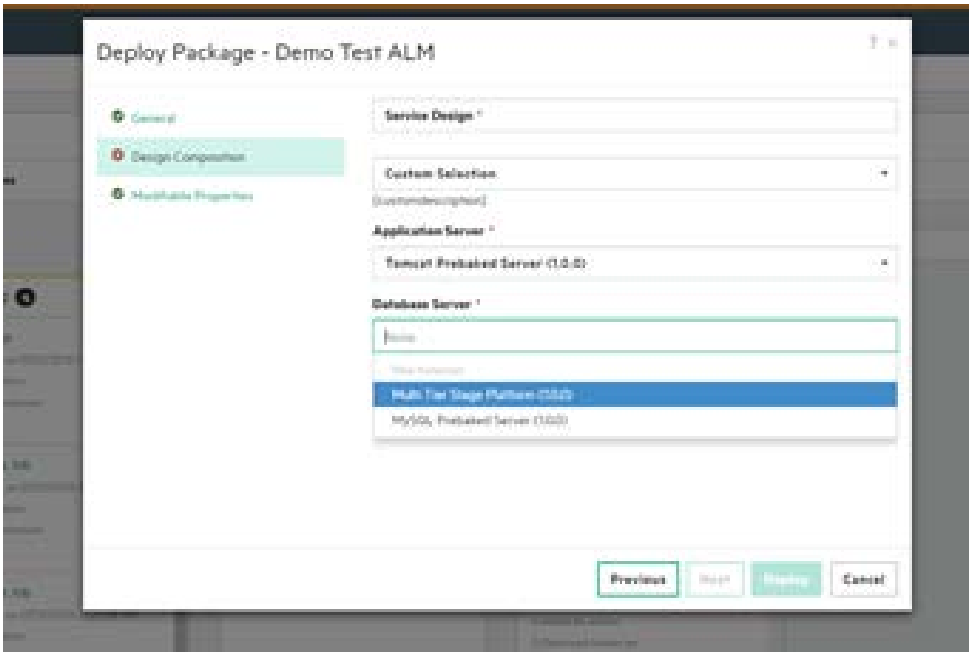


Figure 13: Infrastructure designs listed for application release manager deploying on Stage environment



Conclusion

The topology composition feature helps the end user choose the right infrastructure that must be used for the corresponding lifecycle environment and prevents developers and testers from deploying their application on elevated environments in a more secured manner. It can also match infrastructure design based on the capabilities configured in partial design. This allows users to select from the infrastructure design that is filtered based on the capabilities match found in partial design.

Send documentation feedback

If you have comments about this document, you can send them to clouddocs@hpe.com.

Legal notices

Warranty

The only warranties for Hewlett Packard Enterprise products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Hewlett Packard Enterprise shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Restricted rights legend

Confidential computer software. Valid license from Hewlett Packard Enterprise required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright notice

Trademark notices

Adobe® is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

RED HAT READY™ Logo and RED HAT CERTIFIED PARTNER™ Logo are trademarks of Red Hat, Inc.

The OpenStack word mark and the Square O Design, together or apart, are trademarks or registered trademarks of OpenStack Foundation in the United States and other countries, and are used with the OpenStack Foundation's permission.

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to the following URL and sign-in or register: https://softwaresupport.hpe.com.

Select Manuals from the Dashboard menu to view all available documentation. Use the search and filter functions to find documentation, whitepapers, and other information sources.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your Hewlett Packard Enterprise sales representative for details.

Support

Visit the Hewlett Packard Enterprise Software Support Online web site at https://softwaresupport.hpe.com.