ORACLE
CPQ Cloud

CPQ Cloud – EBS Configurator Reference Integration

ORACLE IMPLEMENTATION GUIDE

ORACLE

Table of Contents

Introduction

The External Configurator Integration feature available in Oracle Configure, Price, and Quote (CPQ) Cloud 2016 R2 is part of an ongoing effort to integrate CPQ Cloud with other products both internal and external to Oracle. Customers can now integrate and leverage the use of an external configurator while using the pricing and quoting capabilities of CPQ Cloud. CPQ Cloud sales specialists can access the external configurator from the CPQ Cloud Home page and then return to CPQ Cloud to price, discount, propose, and order the configured product.

Integration Overview

The reference integration between Oracle CPQ Cloud and the E-Business Suite (EBS) Configurator expands on the External Configurator Integration functionality available in CPQ Cloud 2016 R2. Oracle recommends the administrator responsible for implementing the reference integration first review the ”External Configurator Integration” section of the *Oracle CPQ Cloud 2016 R2 What's New* document.

By implementing the reference integration between CPQ Cloud and the EBS Configurator described in this implementation guide, CPQ Cloud sales specialists can do the following:

- Log in to CPQ Cloud only once to use both CPQ Cloud and the EBS Configurator. Once in the EBS Configurator, CPQ Cloud sales specialists can seamlessly access CPQ Cloud and write transaction lines directly into a transaction in Commerce.
- Create or reconfigure a transaction in the EBS Configurator. Reconfiguring a transaction involves editing existing line items or adding new line items.
- Use the “_config_extra_info” Commerce attribute in standard BML functions, which include library functions, Commerce functions, and Commerce rules.

Note: For additional information about the functionality available to CPQ Cloud sales specialists when using the reference integration between CPQ Cloud and the EBS Configurator, refer to Appendix A. Oracle CPQ Cloud and EBS Configurator Flow.

Purpose

The purpose of this implementation guide is to provide the steps an administrator must take to implement a reference integration between CPQ Cloud and the EBS Configurator.

Audience

This implementation guide is for the administrator who is implementing the reference integration. This guide assumes the administrator has prior CPQ Cloud and EBS administration experience.

Prerequisites

Administrators must implement the CPQ Cloud and EBS Configurator reference integration in environments that contains the following prerequisites:

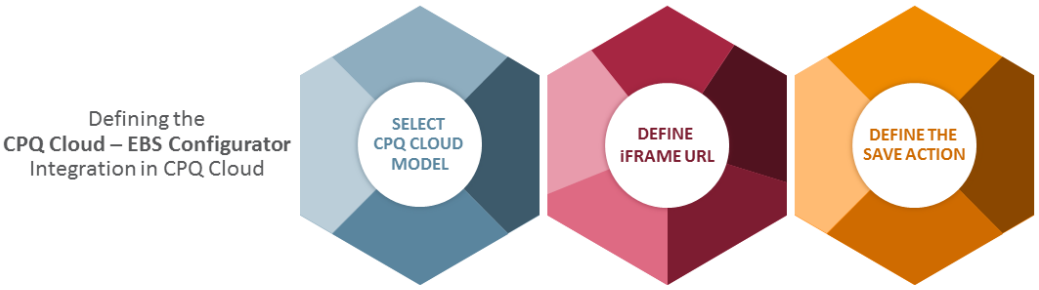
- CPQ Cloud 2016 R2 or later
- EBS 12.1.3/12.2.6 or later with Integrated SOA Gateway setup
  The EBS Web Services invoke the CPQ Cloud External Configurator Advanced Save function to add a configuration or save a modified configuration to a Transaction. The Web Services also support the INIT_MESSAGE and the GET_CONFIG_DETAILS operations.

Note: For additional information, refer to Appendix B. EBS Web Services and the "SOAP API Enhancements" section of the *Oracle CPQ Cloud 2016 R2 What's New* document.

Note: For information about required EBS Configurator patch information, prerequisites, and setup steps for an EBS instance, refer to Doc ID 2241477.1 on My Oracle Support.

Defining the Integration in CPQ Cloud

Administrators must use the CPQ Cloud **Administration Platform** to set up the CPQ Cloud – EBS Configurator integration in CPQ Cloud. When integrating CPQ Cloud with any external configurator, the integration occurs at the CPQ Cloud model level. The following figure illustrates at a high-level the steps an administrator must take to define the CPQ Cloud – EBS Configurator integration in CPQ Cloud.



Note: Customers can integrate multiple external configurators (e.g. EBS and Fusion) to the same CPQ Cloud model using a selector. For additional information, refer to the *Understanding Selectors* section of this implementation guide.

Integrate a CPQ Cloud Model with the EBS Configurator

Administrators must complete the following steps integrate a CPQ model with the EBS Configurator.

1. Go to **Admin** > **Products** > **Catalog Definition**.
   The **Supported Products** page opens with **Product Families** selected by default in the **Navigation** drop-down menu.

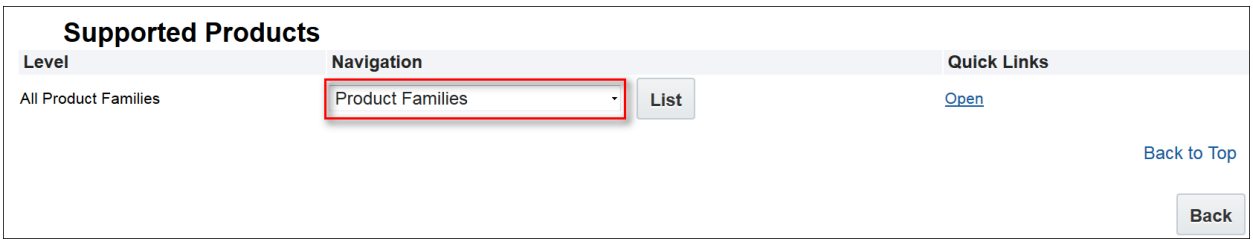


Figure 1: Supported Products Page

2. Click **List**.
   The **Supported Product Families** page opens.

3. Click **List** next to the product family that contains the model you want to integrate with the EBS Configurator. The **Product Line Administration List** page opens with **Models** selected by default for all of the available product lines.

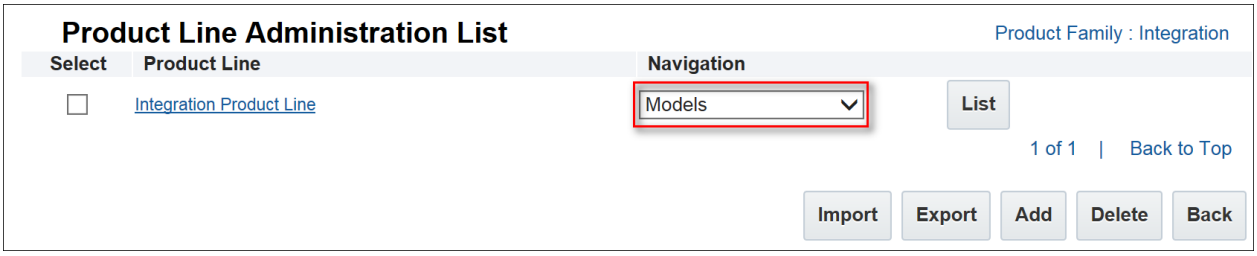


Figure 2: Product Line Administration List

4. Click **List** next to the model you want to integrate with the EBS Configurator.
   The **Model Administration List** page opens.

5. Select **External Configuration** from the **Navigation** drop-down menu.

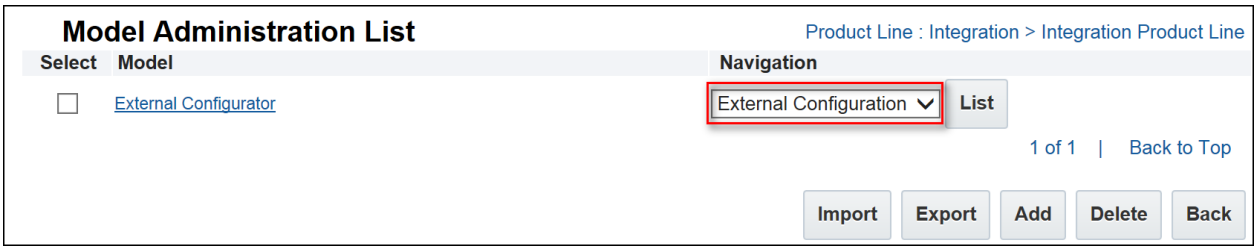


Figure 3: Model Administration List Page

6. Click **List**.
   The **Edit external configuration** page opens with the **Type** option set by default to **None**.
   This indicates there are no external configurators defined for the specified model.

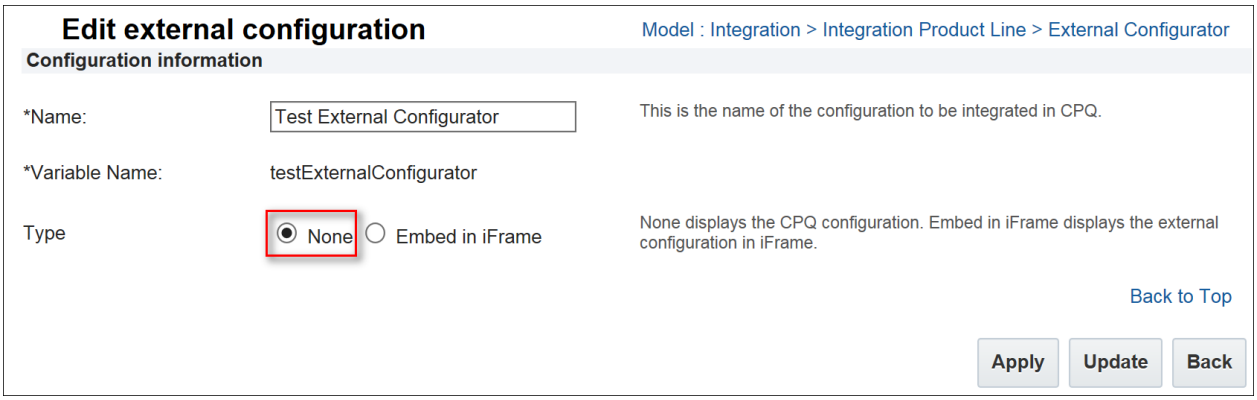


Figure 4: Edit External Configuration Page with None Option Selected

7. Set the **Type** option **to Embed in iFrame**.
   Additional options display in the **Edit external configuration** page.

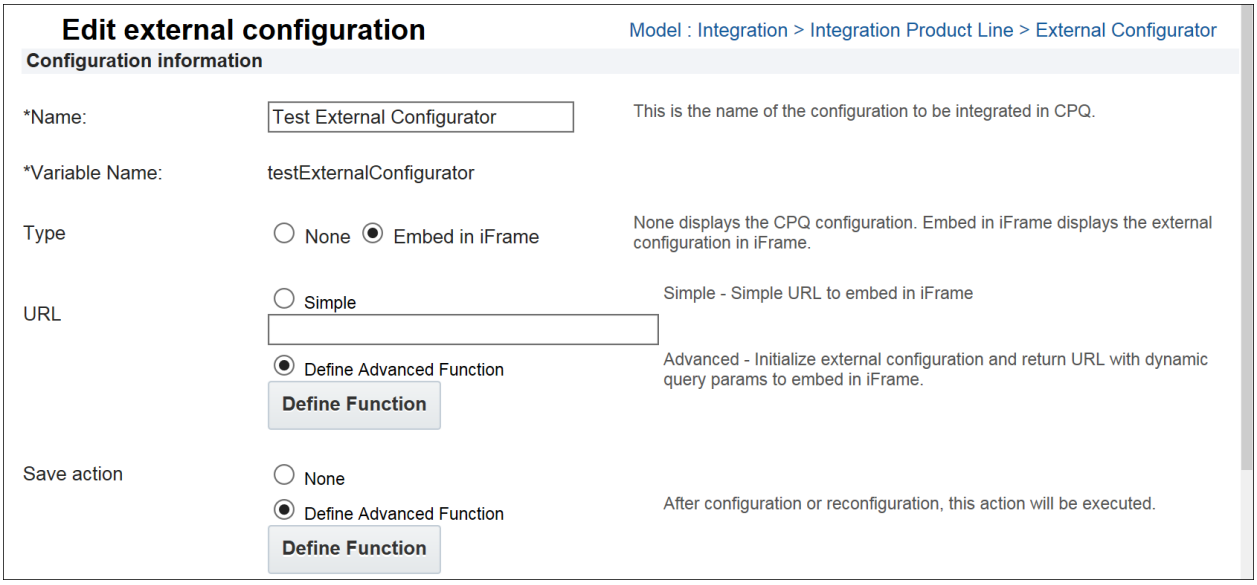


Figure 5: Edit External Configuration Page with Embed in iFrame Option Selected

8. Set the **URL** option to **Define Advanced Function**.

9. Click **Define Function** to open the **BML Editor**. Use the **BML Editor** to define the URL for the EBS Configurator, which is viewable to users via an iFrame in CPQ Cloud.

10. Enter the BML code provided in the “BML_Url.txt” file into the text box that displays to the left of the list of functions. The BML code returns the URL for the iFrame that displays the EBS Configurator.

BML_Url.txt

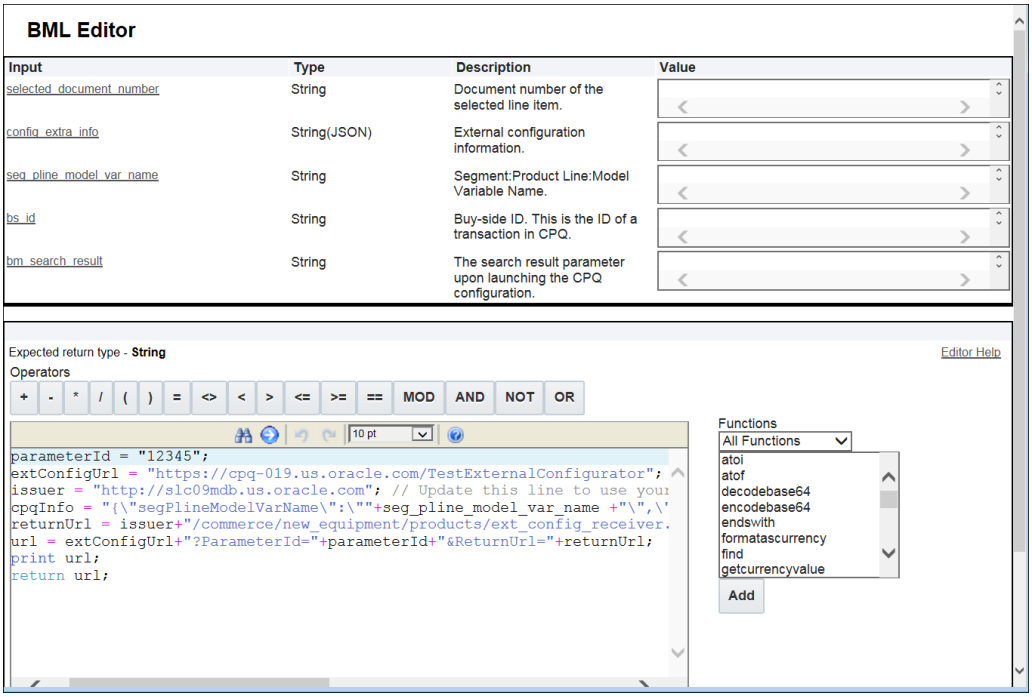


Figure 6: BML for the URL that Displays the EBS Configurator

Note: When an error occurs in the BML, the iFrame does not generate. To debug the BML, administrators can specify runtime values of input arguments using the **Value** fields. The runtime values validate the output of the corresponding BML function.

11. Set the **Save** action to **Define Advanced Function**.
12. Click **Define Function** to open the **BML Editor**. Use the **BML Editor** to enter the BML code for the **Save** action invoked when a user configures or reconfigures a product. Use the BML code provided in the "BML_Save.txt" file.

13. Use the **Value** fields to specify the runtime values used by the BML code to validate the output of the corresponding BML function.

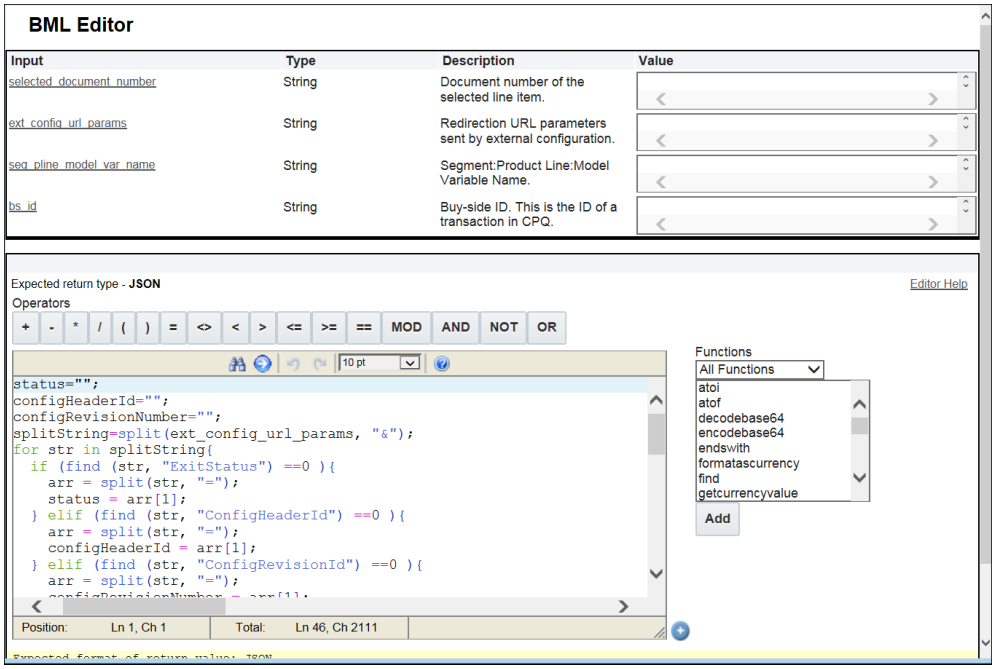


Figure 7: BML for the Save Action

14. Click **Apply** to enable the integration. After defining the integration between CPQ Cloud and the EBS Configurator, the EBS Configurator automatically displays within Configuration Quick Links.

15. Deploy the model you integrated with the EBS Configurator.

Note: Administrators can disable the CPQ Cloud – EBS Configurator integration by returning to the **Edit external configuration** page, setting the **Type** option to **None**, and clicking either **Apply** or **Update**. Since the integration the administrator is disabling uses the **Define Advanced Function** for the **URL** or **Save** actions, the system deletes the BML from both the **BML Editor** and the database.

Use Configuration Quick Links to Access the EBS Configurator

After defining the CPQ Cloud – EBS Configurator integration, the EBS Configurator automatically displays within Configuration Quick Links.

To access the EBS Configurator via Configuration Quick Links:

1. From the **Configuration Quick Links** pane, drill-down to the product family and model associated with the EBS Configurator.

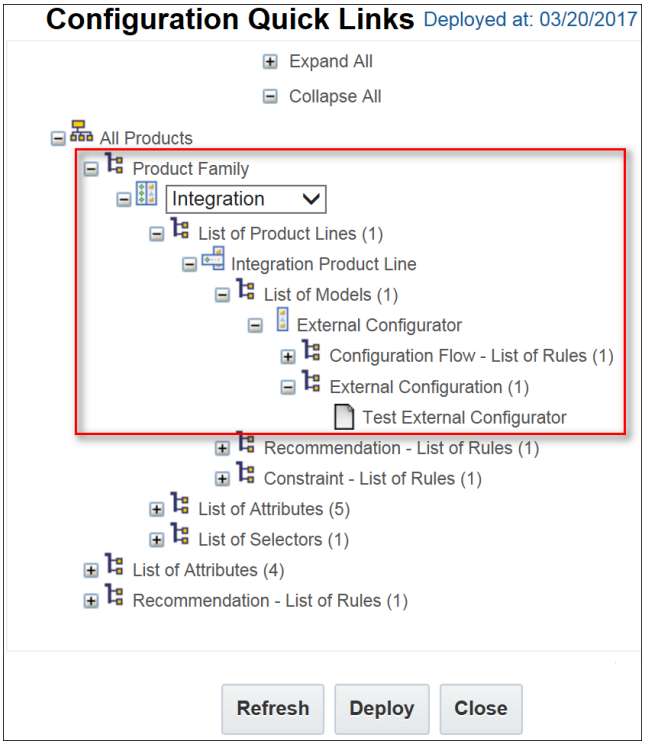


Figure 8: Configuration Quick Links

NOTE: Any un-deployed changes display in red until deployed. Administrators can click **Deploy** from the **Configuration Quick Links** page to deploy any un-deployed changes.

2. Click the **Test External Configurator** link shown in the above figure. The **Edit external configuration** page opens and allows the administrator to view, modify, or disable the EBS Configurator integration.

Implementation Details

Use the information provided in this section of the implementation guide to do the following:

- Add Payload Templates to File Manager
- Use Data Tables to Populate the Payload Templates
- Add New Attribute to the Attribute List
- Add Library Functions to Commerce Process
- Define the Integration in CPQ Cloud

Add Payload Templates to File Manager

The attached File Manager templates contain the payload template file formats for the CPQ Cloud - EBS Configurator reference integration. Administrators must upload the payload template files to File Manager. The payload template files support various EBS Web Service operations. BML reads the template files and replaces the values in brackets, such as {{username}}, with dynamic values. Also attached are sample payload request and response files for the Initiate Message payload, the Configurator payload, and the Reconfigure payload.

System	Operation	File Manager Templates	Sample Payloads
EBS Configurator	getInitMsg	 Init_Msg_Payload.txt	 sample_initpayload.txt  sample_initMsgResponse.txt
EBS Configurator	getConfiguratorDetails	 ConfiguratorPayload.txt	 sample_configuratorPayload.txt  sample_configuratorResponse.txt
EBS Configurator	getReconfig	 Reconfig_Payload.txt	 sample_reconfigInitPayload.txt

System	Operation	File Manager Templates	Sample Payloads
			sample_reconfigSoapResponse.txt

Use Data Tables to Populate the Payload Templates

The data tables described in this section support the reference integration between CPQ Cloud and the EBS Configurator. A sample of each of the data tables (e.g. INT_SYSTEM_DETAILS, INT_SYSTEM_TEMPLATES, and Oracle_ExtCfgDetails) is also included in this section.

INT_SYSTEM_DETAILS

The **INT_SYSTEM_DETAILS** data table stores the usernames (e.g. logins), SOAP endpoints, REST endpoints, and the maximum number of lines to include in large payloads. This data table is queried by the getSystemDetails library function and invokes EBS Web Services to populate the payload template files referred to in the previous section. Administrators should use a secure password type for the INT_SYSTEM_DETAILS data table.

INT_SYSTEM_DETAILS.csv

System	Username	Soap Endpoint	REST Endpoint
EBS Configurator	<Enter the username here to call the web service endpoint>	<Enter the endpoint of the SOAP web services to call here >	<Enter the endpoint of the REST web services to call >

Max Lines in Payload	Description	Password
<Enter the value to divide lines for huge payload>	<Enter the description>	<Enter the password for calling the web service endpoint >

INT_SYSTEM_TEMPLATES

The **INT_SYSTEM_TEMPLATES** data table is queried by the getInitMsg, getConfigurationDetails, and getReconfig Web Service operations, which retrieve and populate the associated template files (e.g. getInitMsg, getConfigurationDetails, and getReconfig).

System	Operations	Template URL
EBS Configurator	getInitMsg	<Enter the template URL path that is uploaded in File Manager>
EBS Configurator	getConfiguratorDetails	<Enter the template URL path that is uploaded in File Manager>
EBS Configurator	getReconfig	<Enter the template URL path that is uploaded in File Manager>

Oracle_ExtCfgDetails

Many of the library functions identified in this implementation guide use the **Oracle_ExtCfgDetails** data table to obtain specific information via BML. The following table contains descriptions of the columns in this data table.

Oracle_ExtCfgDetails.csv

SegPlineModelVarName	Selector_Value	ext_Config_URL	return_URL	application_Id	database_Id	responsibility_Id
<Enter the variable name of the model>	<Enter the Selector value of the model>	<Enter the external config URL>	<Enter the URL to the page to return in CPQ Cloud>	<Enter the application Id for the external configurator>	<Enter the database Id for the external configurator>	<Enter the responsibility Id for the external configurator>

inventory_item_Id	organization_Id	System	Invocation_Id	priceBookVarName	Application_Name
<Enter the inventory item id>	<Enter the organization id>	Configurator	<Enter the invocation id for the external configurator>	<Enter the variable name for the default price book used in CPQ Cloud>	<Enter the application Id for the external configurator>

Add New Attribute to the Attribute List

The Reference Integration between CPQ Cloud and the EBS External Configurator makes changes to the "Oracle Quote to Order" Commerce process in attributes, BML, and library functions.

A new Commerce attribute is available in CPQ Cloud 2016 R2 and supports the external configurator feature. Administrators can use the attribute in standard BML functions, which include library functions, Commerce functions, and Commerce rules.

Name	Variable Name	Type	Description
_config_extra_info	Config Extra Info	Text	This Transaction line variable is used to store the EBS Configurator information in JSON format, including configHeaderID, configRevisionNumber, configuratorPath, selectorValue, and partNumber that are passed to the CPQ Cloud Commerce quote from the configuration session.

Add Library Functions to Commerce Process

Descriptions and code for each of the library functions used by the CPQ Cloud – EBS Configurator are included in this section. Manually add the library functions to your Commerce process by navigating to **Admin** > **Developer Tools** > **BML Library**.

Json getSystemDetails (String system)

This function gets the username, password, and endpoint for the external configurator from the System Details data table for use in the SOAP request.

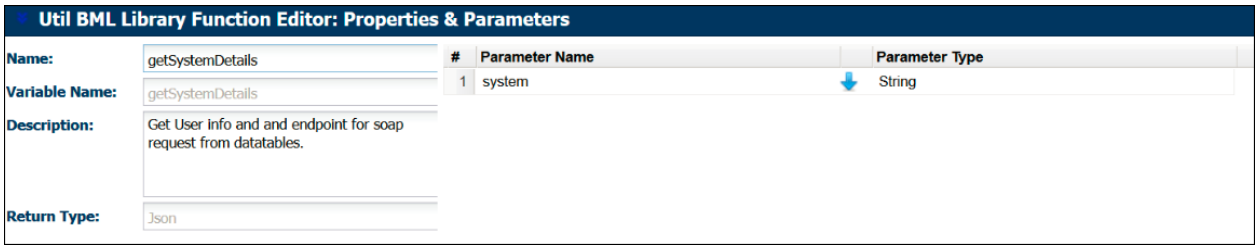


Figure 9: Return Type and Input Information - getSystemDetails

String getSytemTemplate (String system, String operation)

This function gets the location of the payload template in File Manager used in BML to construct the payload to send to the external configurator. This is accomplished by querying the INT_SYSTEM_TEMPLATES data table for a particular system and operation.

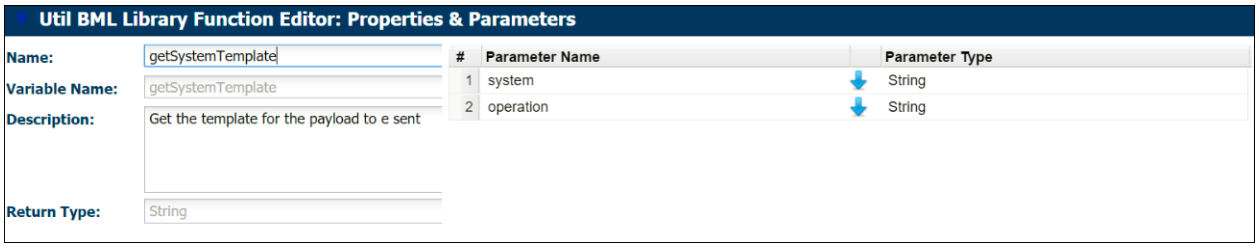


Figure 10: Return Type and Input Information – getSystemTemplate

String getSelectorValue (String reconfigSelectorValue, String bm_search_result)

This function sets the CPQ Cloud configuration selector value based on whether the operation is a configuration operation or a reconfiguration operation. When a value is not passed, the selector value is set to the default value.

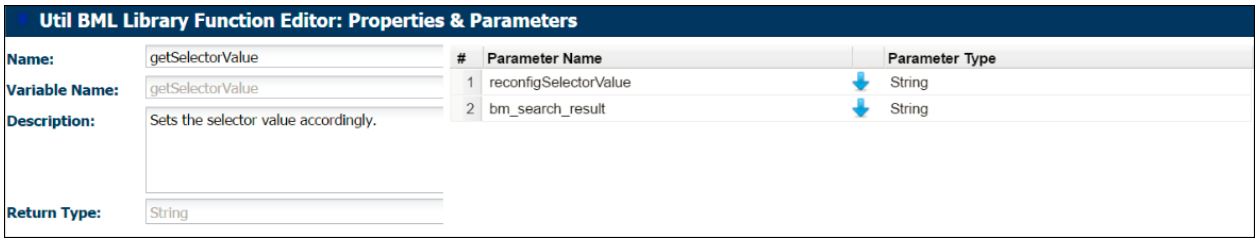


Figure 11: Return Type and Input Information – getSelectorValue

Json getExtCfgDetails (String segPineModelVarName, String bm_searchselector)

This function queries for the following values for a particular CPQ Cloud model and selector: ext_Config_URL, return_URL, database_Id, inventory_item_id, application_Name, organization_id. The function also checks to see if any of values are null. If any of the values are null, an error message displays.

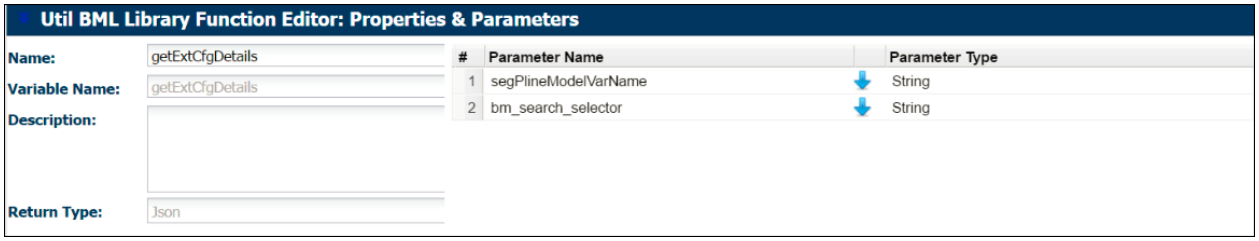


Figure 12: Return Type and Input Information – getExtCfgDetails

String constructPayload (Json jsonobj1, Json jsonuser, Json jsonobj2)

This function is used to construct a payload and read the session ID from the response. If the session ID is null, an error message is returned. If the session ID is not null, the session ID is returned.

Util BML Library Function Editor: Properties & Parameters

Name:	constructPayload
Variable Name:	constructPayload
Description:	
Return Type:	String

#	Parameter Name	Parameter Type
1	jsonobj1	Json
2	jsonuser	Json
3	jsonobj2	Json

Figure 13: Return Type and Input Information – constructPayload

String constructSavePayload (Json jsonUser, Json jsonGetMessageObj, String templateUrl)

This function is used to construct a payload for the save BML and return a response.

Util BML Library Function Editor: Properties & Parameters

Name:	constructSavePayload
Variable Name:	constructSavePayload
Description:	
Return Type:	String

#	Parameter Name	Parameter Type
1	jsonUser	Json
2	jsonGetMessageObj	Json
3	templateUrl	String

Figure 14: Return Type and Input Information – constructSavePayload

Json computeSoapResponse (String configuratorResponse, String seg_pline_model_var_name, String selectorValue, String modelInventoryItemId, String configHeaderId, String revNumber)

This function reads the response from Web Services and constructs a JSON object returned by the Save BML.

Util BML Library Function Editor: Properties & Parameters

Name:	computeSoapResponse
Variable Name:	computeSoapResponse
Description:	
Return Type:	Json

#	Parameter Name	Parameter Type
1	configuratorResponse	String
2	seg_pline_model_var_name	String
3	selectorValue	String
4	modelInventoryItemId	String
5	configHeaderId	String
6	revNumber	String

Figure 15: Return Type and Input Information – computeSoapResponse

String getInventoryItemId (String seg_pline_model_var_name, String selectorValue)

This function queries inventory_item_id from Oracle_ExtCfgDetails for a particular model and selector value.

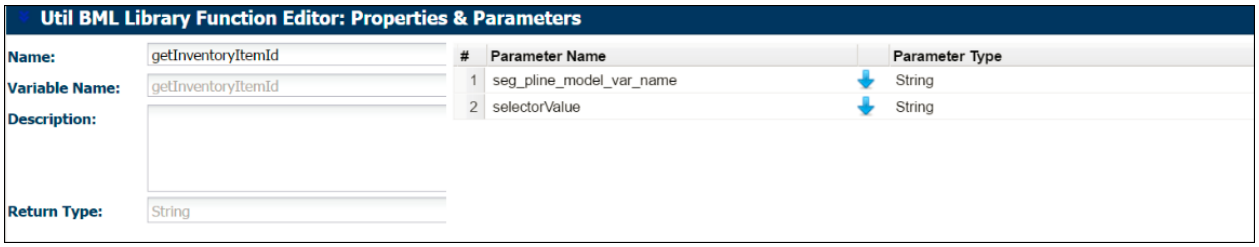


Figure 16: Return Type and Input Information – getInventoryItemId

Json getMessage (String ext_confiig_url_params)

This function splits the parameter ext_config_url_params into ConfigHeaderId, RevNumber, SelectorValue, and StatusMessage and returns them in the form of a JSON object. If any of these parameters are null, the JSON object returns an error.

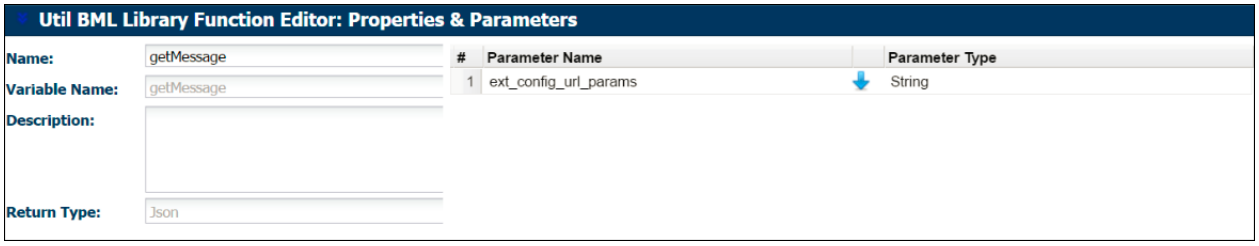


Figure 17: Return Type and Input Information – getMessage

String constructCpqUrlInfo (Json jsonObj, String return Url, String bm_search_selector, Boolean reconfig, String reconfigSelectorValue)

This function encodes the return URL and appends it with the selector value.

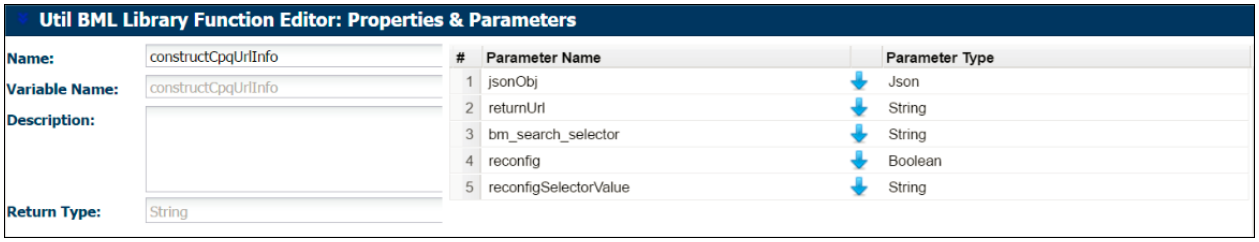


Figure 18: Return Type and Input Information – constructCpqUrlInfo

Prepare and Deploy Data Tables and Upload EBS Parts to CPQ Cloud

Administrators must also complete the following steps:

- Update the entry point of the return URL in the data tables
- Deploy the data tables referenced in this implementation guide
- Use a custom Part attribute (Partner Part ID) to capture the EBS Part Id for each part stored in CPQ Cloud.
- Use standard CPQ Cloud features for parts sync from EBS into CPQ Cloud.

Note: CPQ Cloud and EBS must share a common parts list. Administrators can accomplish this by uploading the EBS parts that correspond to the CPQ models used in the CPQ Cloud – EBS Configurator reference integration.

Understanding Selectors

Administrators can map multiple external configurators (e.g. EBS and Fusion) to the same CPQ Cloud model using a selector. When a selector is not used, administrators can only map one model to one configurator. When administrators have multiple external configurators to integrate with the same CPQ model, implementing a selector saves time during the integration process.

For example: If a customer does not implement a selector and has 10 external configurators to integrate with CPQ Cloud, the administrator would need to create 10 different models.

Search a Selector

After implementing a selector, CPQ Cloud sales specialists can search the selector for a specific external configurator by completing the following steps:

1. From the **Admin Home** page, select the product line that contains the model mapped to two or more external configurators.
2. Under the **Select & Configure** heading, click **Search selector**.

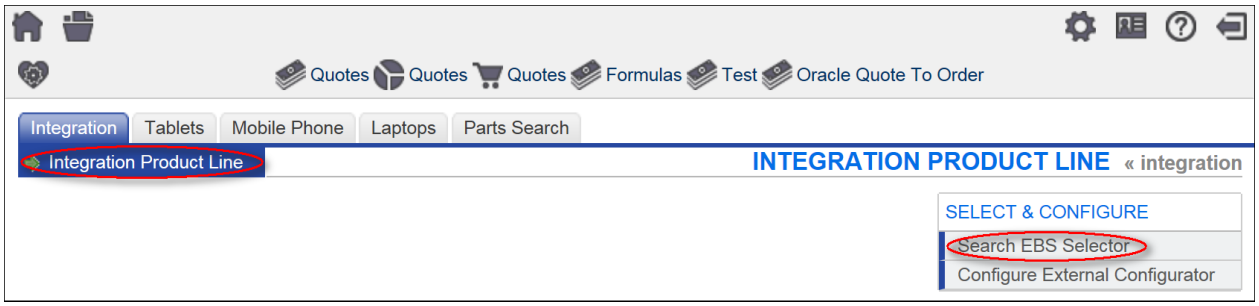


Figure 19: Search Selector

3. A search page opens and displays search attributes. In the below screenshot, the search attributes are “Computer Type” and “Hard Drive Size”.

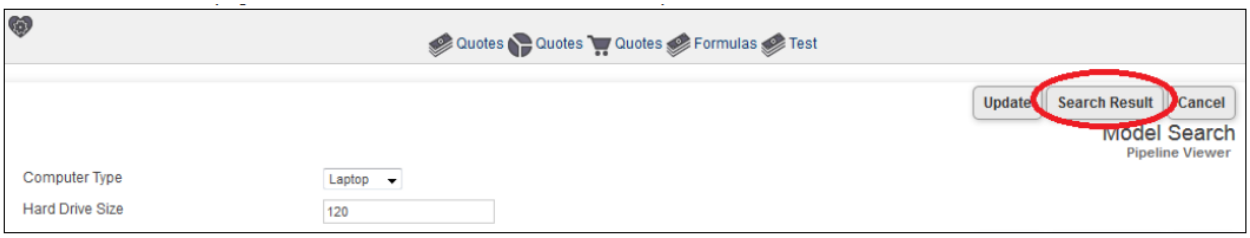


Figure 20: Search Page Displaying Search Attributes

4. From the search attribute drop-down menus, select input values for the attributes.
5. Click **Search Result**.
6. A search results page opens with a link to the model configurator. In the below screenshot, “Computer Type = “Laptop” and “Hard Drive Size”’= “120” are the search results.

Figure 21: Search Results Page with Link to Model Configurator

7. By clicking the **Model 1** button in the above screenshot, the administrator goes directly to the **Model Configuration** page with the search results populated.

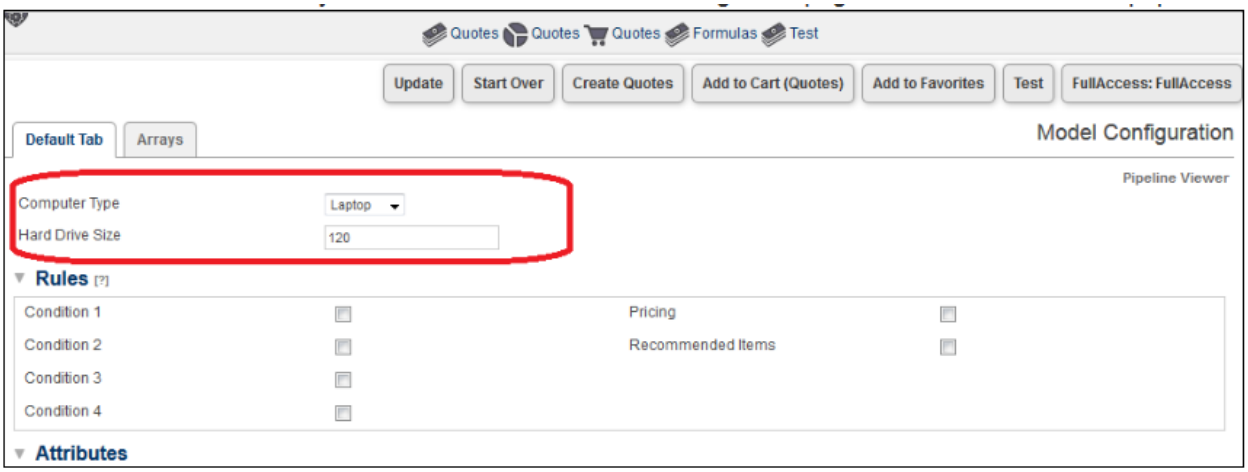


Figure 22: Model Configuration Page

Create Search Attributes

Complete the following procedure to create search attributes for a selector.

Note: If you are using existing attributes as your search attributes or are only integrating with one external configurator, skip this procedure.

1. Navigate to **Admin** > **Products > Catalog Definition**.
   The **Supported Products** page opens.
2. From the **Navigation** drop-down menu, select **Configurable Attribute**.
3. Click **List**.
   The **Configurable Attributes Administrative List** page opens.
4. Click **Add**.
   The **Attribute Editor** opens. Use the **Attribute Editor** to create configurable attributes that function as the search attributes for a selector.

Create a Selector

Complete the following procedure to create a selector:

1. Navigate to **Admin** > **Products > Catalog Definition**.
   The **Supported Products** page opens with **Product Families** displaying by default in the **Navigation** drop-down menu.
2. Click **List**.
   The **Supported Product Families** page opens.

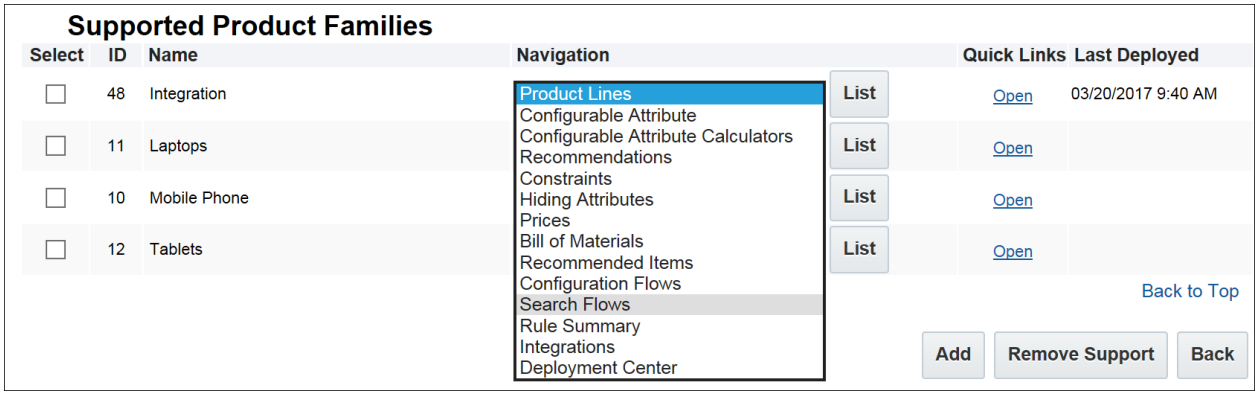


Figure 23: Supported Product Families

3. From the **Navigation** drop-down menu next to the product family for which you set up the external configurator, select **Search Flows**.
4. Click **List**.
   The **Selector List** page opens.
5. Click **Add**.
   The **Selector Administration** page opens.

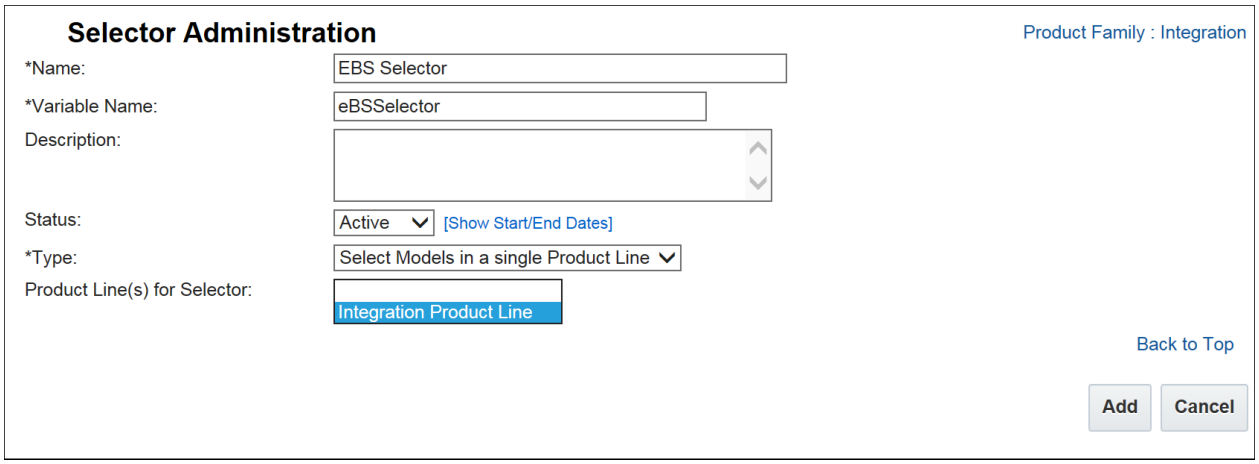


Figure 24: Selector Administration Page

6. In the **Name** field, enter a name for the selector.
7. From the **Type** drop-down menu, select the **Select Models in a single Product line** option.
8. In the **Product Line(s) for Selector** field, select the product lines to show in the selector drop-down menu.
9. Click **Add**.
   The **Selector List** page opens.

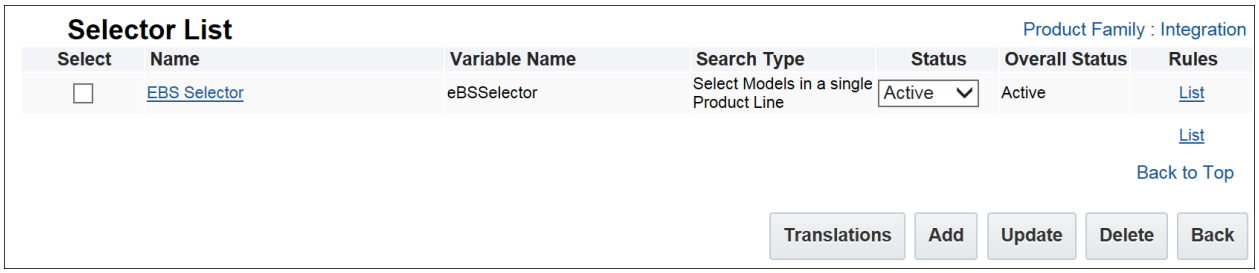


Figure 25: Selector List Page

10. In the **Rules** column, select the **List** link.
    The **Search Flow: Rules List** page opens.

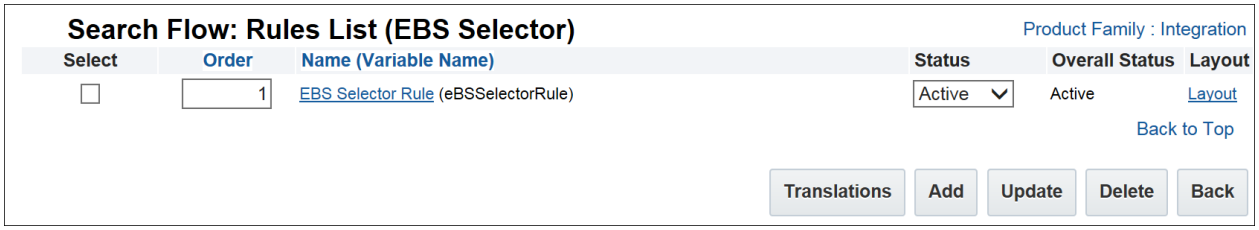


Figure 26: Search Flow: Rules List Page

11. Click **Add**.
    The **Search Flow** page opens.

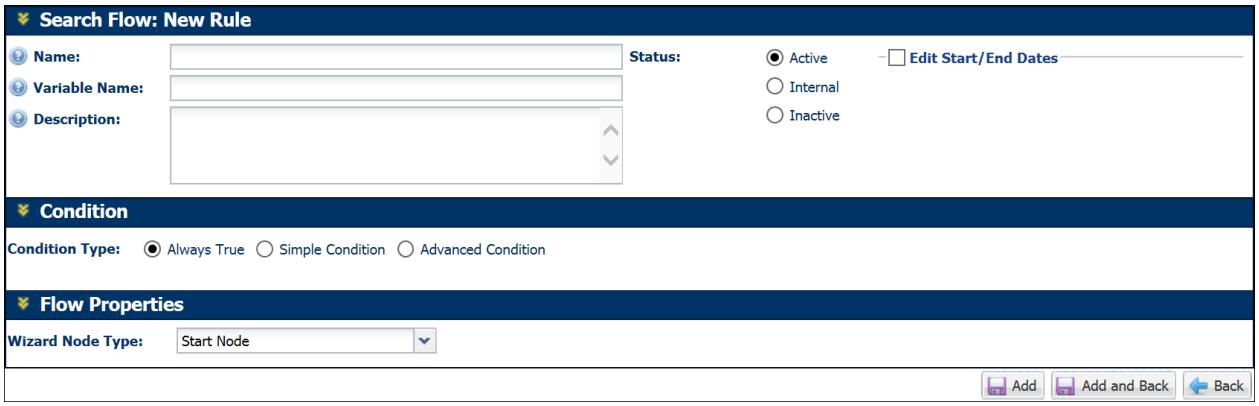


Figure 27: Search Flow Page

12. For the **Condition Type**, select **Always True**.
13. For the **Wizard Node Type**, select **Start and End Node**.
14. For the **Result Type**, select either **External URL** or **Define Function**. The following attachment is a sample function to evaluate the results in use.


sample_selectorpage_function.txt

Note: If defining a function, make sure to add the attributes necessary for the function.

Deploy Changes

Complete the following procedure to deploy the selector you just created.

1. Navigate to **Admin > Products > Catalog Definition.**
   The Supported Products page opens.
2. In the **Navigation** drop-down menu, **Product Families** displays by default.
3. Click **List**.
   The **Supported Product Families** page opens.
4. In the **Navigation** drop-down menu next to the selector you just created, select **Deployment Center**.
5. Click **List**.
   The **Deployment Scheduler** opens, which you will use to deploy your changes.

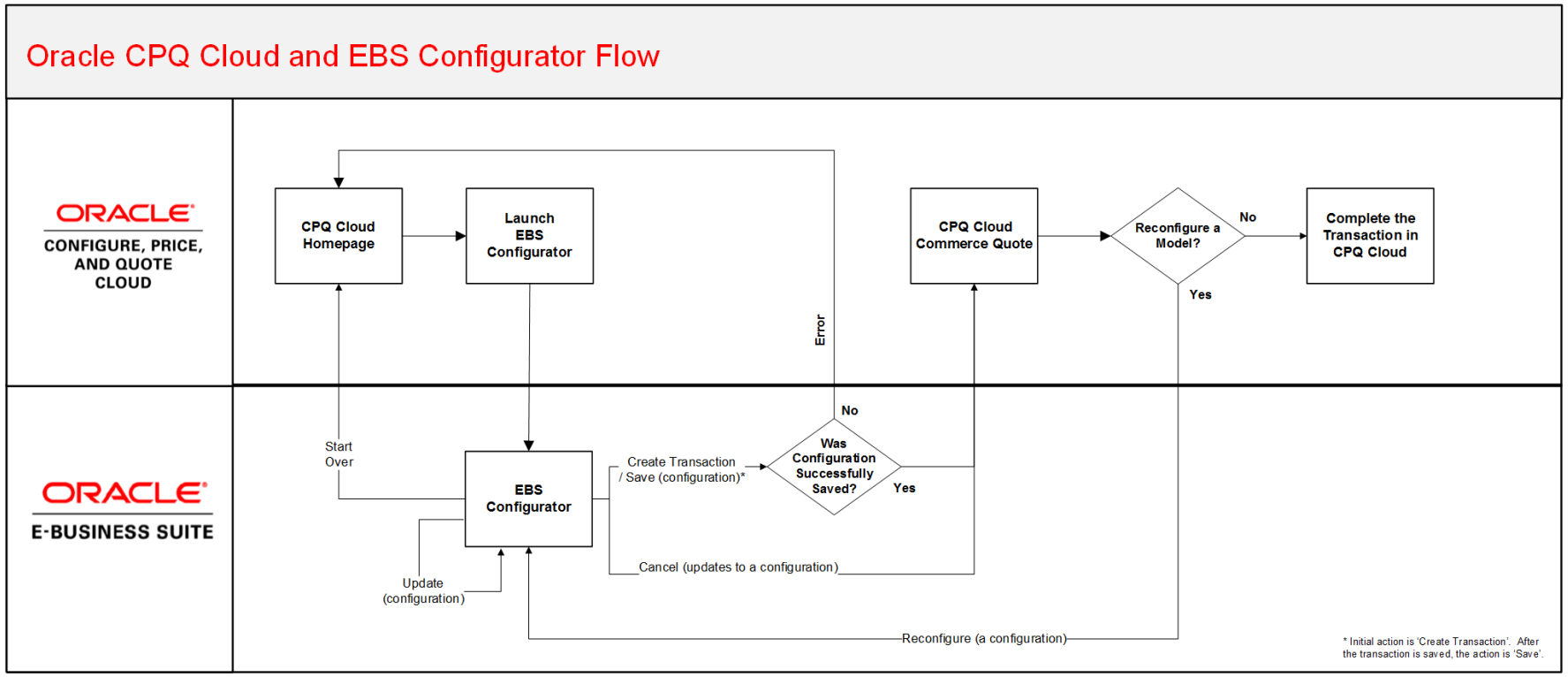
Troubleshooting Tips

The following table contains error messages that administrators may potentially encounter when implementing the CPQ Cloud – EBS base reference integration.

Error Message	Resolution
Unable to establish a connection to the external configuration site.	Check bm_logs. The error most likely occurs when a data table is incomplete or the entered selector value is not present in the data table.
Error in reconfiguration please try again.	Check the values of configRevisionNumber or configHeaderId

Appendix A. Oracle CPQ Cloud and EBS Configurator Flow

The following diagram illustrates the functionality available to CPQ Cloud sales specialists using the reference integration between CPQ Cloud and the EBS Configurator.



Appendix B. EBS Web Services

The following WSDL describes the web services invoked by the CPQ Cloud – EBS Configurator integration.

```
<definitions
   xmlns:tns="http://xmlns.oracle.com/apps/cz/soaprovider/plsql/cz_nb_ws_test/"
   xmlns="http://schemas.xmlsoap.org/wsdl/"
   xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns:tns1="http://xmlns.oracle.com/apps/cz/soaprovider/plsql/cz_nb_ws_test/get_config_details/"
   xmlns:tns2="http://xmlns.oracle.com/apps/cz/soaprovider/plsql/cz_nb_ws_test/init_message/" name="CZ_NB_WS_TEST"
targetNamespace="http://xmlns.oracle.com/apps/cz/soaprovider/plsql/cz_nb_ws_test/">
   <types>
      <schema
         xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
targetNamespace="http://xmlns.oracle.com/apps/cz/soaprovider/plsql/cz_nb_ws_test/get_config_details/">
         <include
schemaLocation="http://rws60124rems.us.oracle.com:8085/webservices/SOAProvider/plsql/cz_nb_ws_test/APPS_CZ_NB_WS_TEST_GET_
CONFIG_DETAILS.xsd"/>
      </schema>
      <schema
         xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
targetNamespace="http://xmlns.oracle.com/apps/cz/soaprovider/plsql/cz_nb_ws_test/init_message/">
         <include
schemaLocation="http://rws60124rems.us.oracle.com:8085/webservices/SOAProvider/plsql/cz_nb_ws_test/APPS_CZ_NB_WS_TEST_INIT
_MESSAGE.xsd"/>
      </schema>
      <schema
         xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
targetNamespace="http://xmlns.oracle.com/apps/cz/soaprovider/plsql/cz_nb_ws_test/">
```

```
         <element name="SOAHeader">
            <complexType>
               <sequence>
                  <element name="Responsibility" minOccurs="0" type="string"/>
                  <element name="RespApplication" minOccurs="0" type="string"/>
                  <element name="SecurityGroup" minOccurs="0" type="string"/>
                  <element name="NLSLanguage" minOccurs="0" type="string"/>
                  <element name="Org_Id" minOccurs="0" type="string"/>
               </sequence>
            </complexType>
         </element>
      </schema>
   </types>
   <message name="GET_CONFIG_DETAILS_Input_Msg">
      <part name="header" element="tns:SOAHeader"/>
      <part name="body" element="tns1:InputParameters"/>
   </message>
   <message name="GET_CONFIG_DETAILS_Output_Msg">
      <part name="body" element="tns1:OutputParameters"/>
   </message>
   <message name="INIT_MESSAGE_Input_Msg">
      <part name="header" element="tns:SOAHeader"/>
      <part name="body" element="tns2:InputParameters"/>
   </message>
   <message name="INIT_MESSAGE_Output_Msg">
      <part name="body" element="tns2:OutputParameters"/>
   </message>
   <portType name="CZ_NB_WS_TEST_PortType">
      <operation name="GET_CONFIG_DETAILS">
         <input message="tns:GET_CONFIG_DETAILS_Input_Msg"/>
         <output message="tns:GET_CONFIG_DETAILS_Output_Msg"/>
      </operation>
      <operation name="INIT_MESSAGE">
         <input message="tns:INIT_MESSAGE_Input_Msg"/>
         <output message="tns:INIT_MESSAGE_Output_Msg"/>
      </operation>
   </portType>
   <binding name="CZ_NB_WS_TEST_Binding" type="tns:CZ_NB_WS_TEST_PortType">
      <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="GET_CONFIG_DETAILS">
         <soap:operation
soapAction="http://rws60124rems.us.oracle.com:8085/webservices/SOAProvider/plsql/cz_nb_ws_test/"/>
```

```
         <input>
            <soap:header message="tns:GET_CONFIG_DETAILS_Input_Msg" part="header" use="literal"/>
            <soap:body parts="body" use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
      </operation>
      <operation name="INIT_MESSAGE">
         <soap:operation
soapAction="http://rws60124rems.us.oracle.com:8085/webservices/SOAProvider/plsql/cz_nb_ws_test/"/>
         <input>
            <soap:header message="tns:INIT_MESSAGE_Input_Msg" part="header" use="literal"/>
            <soap:body parts="body" use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
      </operation>
   </binding>
   <service name="CZ_NB_WS_TEST_Service">
      <port name="CZ_NB_WS_TEST_Port" binding="tns:CZ_NB_WS_TEST_Binding">
         <soap:address location="http://rws60124rems.us.oracle.com:8085/webservices/SOAProvider/plsql/cz_nb_ws_test/"/>
      </port>
   </service>
</definitions>
```

ORACLE

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US


facebook.com/oracle


twitter.com/oracle


oracle.com

Integrated Cloud Application & Platform Services