ORACLE
CPQ Cloud

CPQ Cloud – Fusion Configurator Reference Integration

ORACLE IMPLEMENTATION GUIDE

ORACLE

Table of Contents

Introduction 1

Integration Overview 1
Purpose 1
Audience 1
Prerequisites 2

Defining the Integration in CPQ Cloud 3

Integrate a CPQ Model with the Fusion Configurator 3
Use Configuration Quick Links to Access the Fusion Configurator 7

Implementation Details 8

Add Payload Templates to File Manager 8
Use Data Tables to Populate the Payload Templates 8
Add New Attribute to the Attribute List 11
Add Library Functions to Commerce Process 12
Prepare and Deploy Data Tables and Upload Fusion Parts to CPQ Cloud 16
Get Fusion Parts from the Fusion Database 17
Upload Fusion parts to CPQ Cloud 19

Understanding Selectors 20

Search a Selector 20
Create Search Attributes 21
Create a Selector 22
Deploy Changes 24

Troubleshooting Tips 25

Appendix A. Oracle CPQ Cloud and Fusion Configurator Flow 26

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 Fusion 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 Fusion 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 Fusion Configurator. Once in the Fusion 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 Fusion 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 Fusion Configurator, refer to Appendix A. Oracle CPQ Cloud and Fusion 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 Fusion 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 Fusion administration experience.

Prerequisites

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

- CPQ Cloud 2016 R2 or later
- Fusion Release 13 or later
  The Fusion Web Services invoke the External Configurator Advanced Save function to add a configuration or save a modified configuration to a Transaction. The Web Services also support the Get Configurations and the getURLPayload operations.

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

Defining the Integration in CPQ Cloud

Administrators must use the CPQ Cloud **Administration Platform** to set up the CPQ Cloud – Fusion 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 – Fusion 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 Model with the Fusion Configurator

Administrators must complete the following steps to integrate a CPQ Cloud model with the Fusion 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.

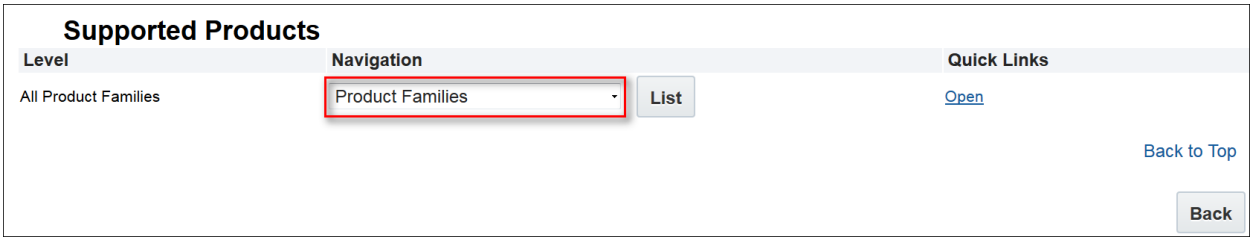


Figure 1: Supported Products Page

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 Fusion Configurator. The **Product Line Administration List** page opens with **Models** selected by default for all of the available product lines.

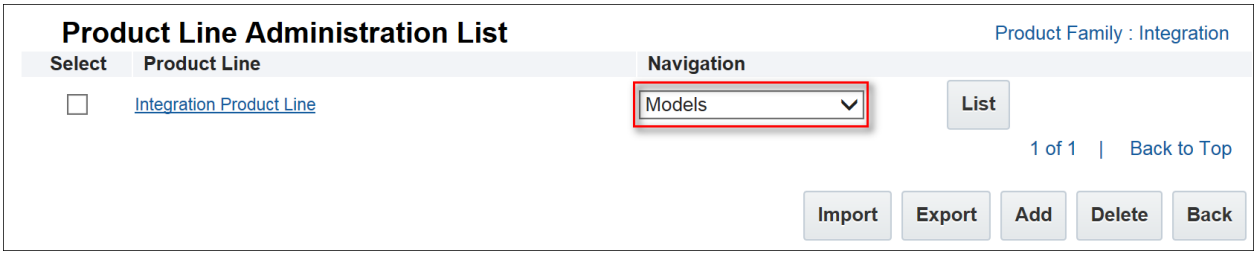


Figure 2: Product Line Administration List

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

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

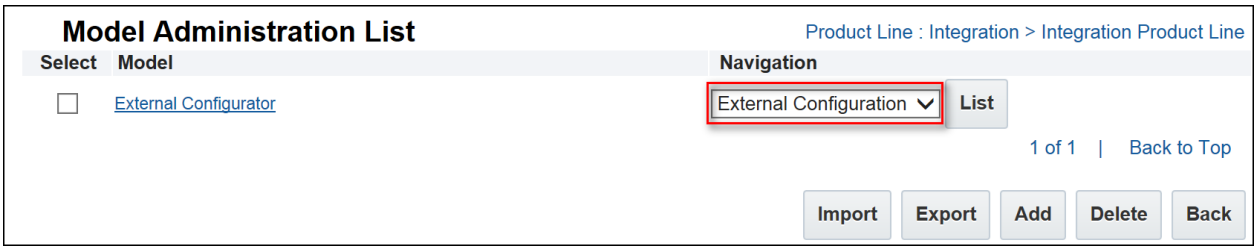


Figure 3: Model Administration List Page

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.

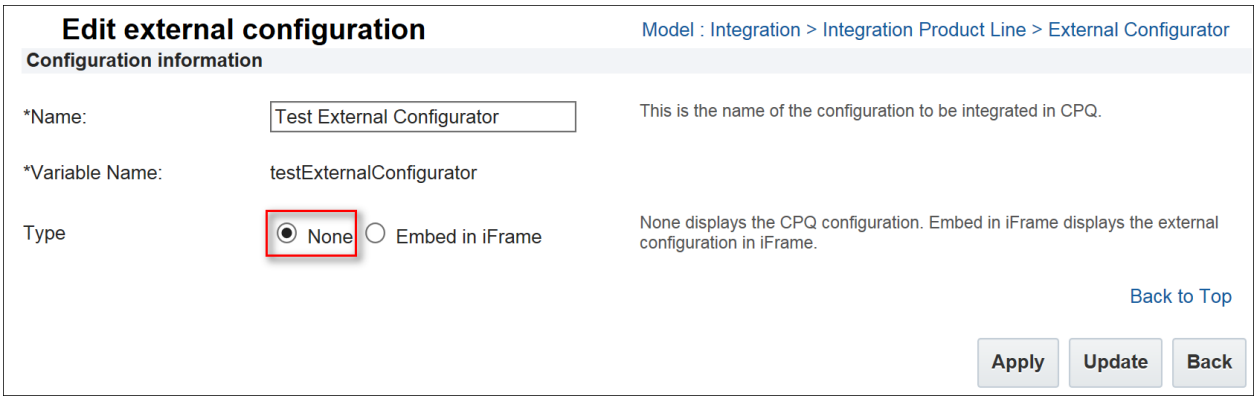


Figure 4: Edit External Configuration Page with None Option Selected

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

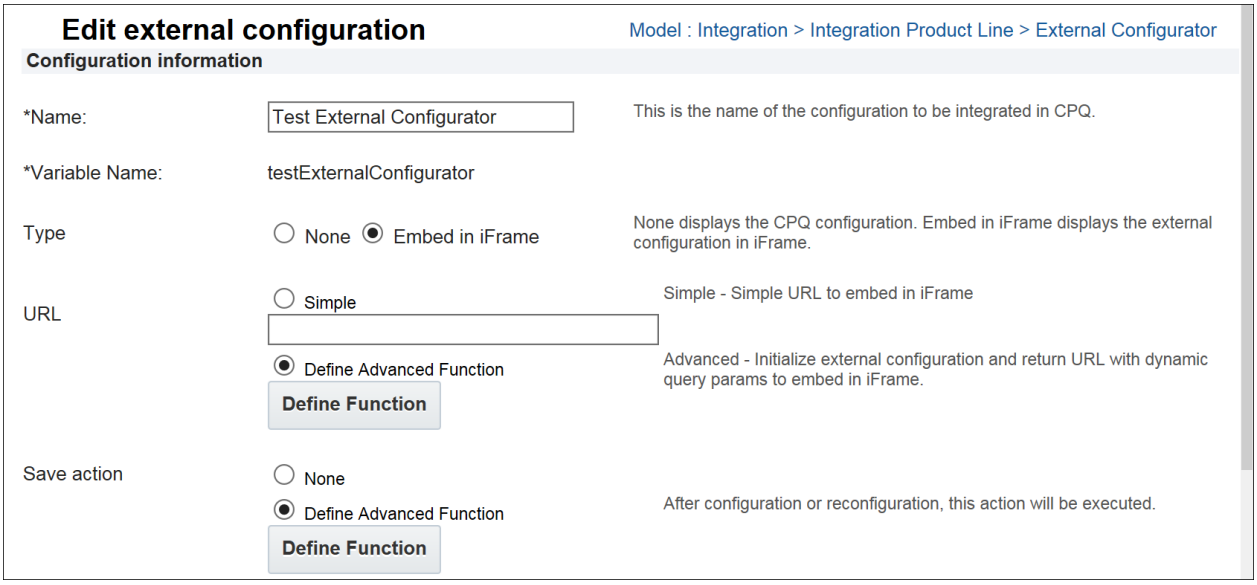


Figure 5: Edit External Configuration Page with Embed in iFrame Option Selected

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 Fusion Configurator, which is viewable to users via an iFrame in CPQ Cloud.

10. Enter the BML code provided in the “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 Fusion Configurator.

Url.txt

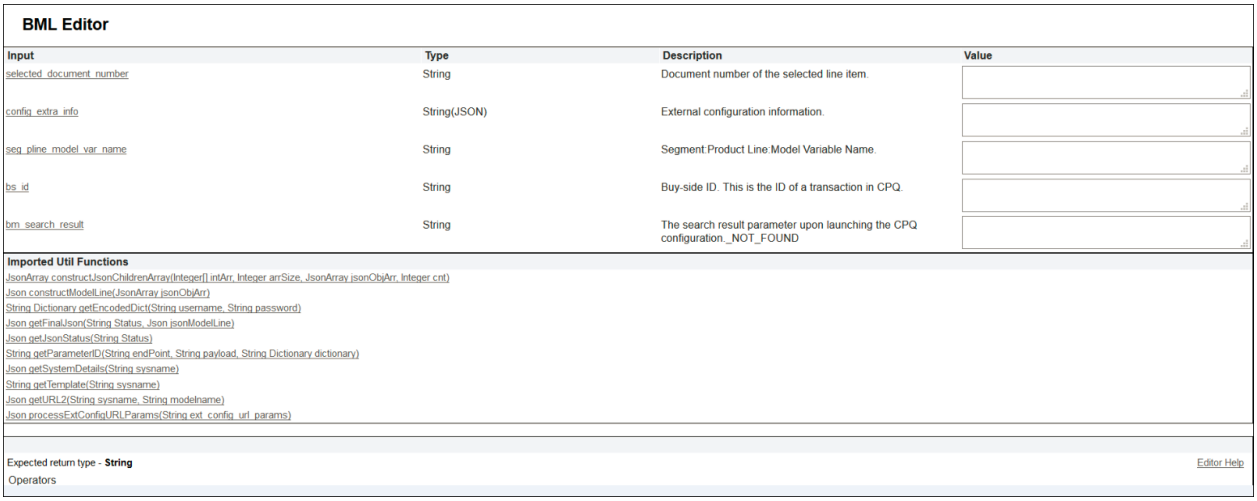


Figure 6: BML for URL that Displays the Fusion Configurator

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 "Save.txt" file.

Save.txt

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

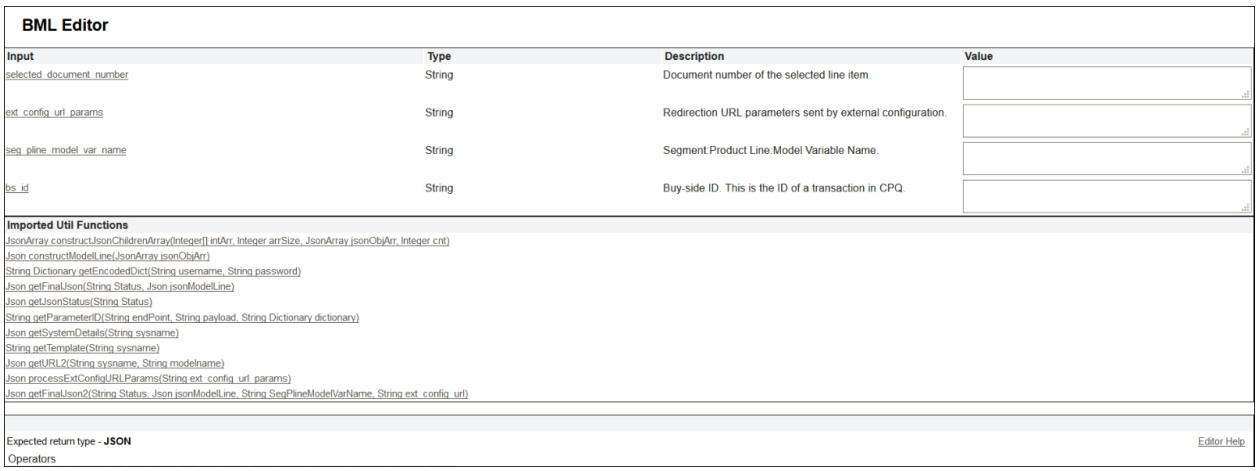


Figure 7: BML for the Save Action

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

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

Note: Administrators can disable the CPQ Cloud – Fusion 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** and **Save** actions, the system deletes the BML from both the **BML Editor** and the database.

Use Configuration Quick Links to Access the Fusion Configurator

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

To access the Fusion Configurator via **Configuration Quick Links**:

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

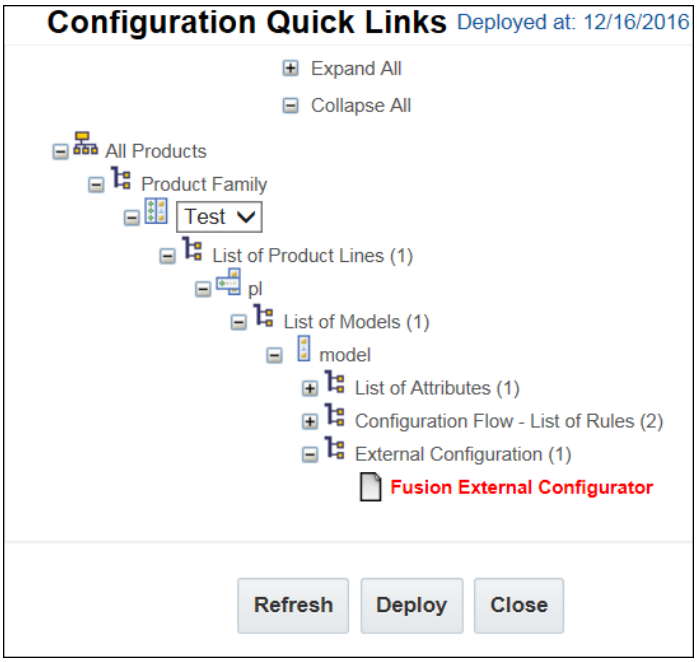


Figure 8: Configuration Quick Links

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

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

Add Payload Templates to File Manager

The attached File Manager templates contain the payload template file formats for the CPQ Cloud - Fusion Configurator reference integration. Administrators must upload the payload template files to File Manager. The payload template files support various Fusion Web Service operations. BML reads the template files and replaces the values in brackets, such as {{InventoryItemId}}, with dynamic values.

System	Operation	File Manager Templates
FusionReconfig	getURLPayload	 Fusion_Reconfig_URL_Payload.txt
Fusion	getURLPayload	 Fusion_URL_Payload.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 Fusion 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 (i.e. logins), REST endpoints, and the configurator REST response URL. Queried by the getSystemDetails library function, this data table invokes Fusion 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	MaxLinesInPayload
Fusion 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 here>	<Enter the value to divide lines for huge payload>

Description	Password	CnfgRestResponseURL
<Enter the description>	<Enter the password for calling the Web Services endpoint>	<Enter the Web Service configurator REST response URL to call the service>

INT_SYSTEM_TEMPLATES

The getURLPayload Web Service operation queries the **INT_SYSTEM_TEMPLATES** data table, which retrieves and populates the associated payload template files.

INT_SYSTEM_TEMPLATES.csv

System	Operations	Template URL
FusionReconfig	getURLPayload	<Enter the template URL path that is uploaded in File Manager>
Fusion	getURLPayload	<Enter the template URL path that is uploaded in File Manager>

Oracle_ExtCfgDetails

Used by many of the library functions identified in this implementation guide, the **Oracle_ExtCfgDetails** data table obtains 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	inventory_item_id
<Enter the variable name of the model>	<Enter the Selector value of the model>	<Enter the external configurator URL>	<Enter the URL of the page to return in CPQ Cloud>	<Enter the inventory_item_id>

modelVarName	organization_Id	System	Invocation_Id	priceBookVarName	Application_Name
<Enter the model var name>	<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 Fusion 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 Fusion Configurator information in JSON format, including the 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 – Fusion Configurator integration are included in this section. Manually add the library functions to your Commerce process by navigating to **Admin** > **Developer Tools** > **BML Library**.

Json getURL2 (String sysname, String modelname)

This function queries ext_Config_URL and return_URL from the Oracle_ExtCfgDetails data table for a particular system and model and returns them both in a JSON object.

getURL2.txt

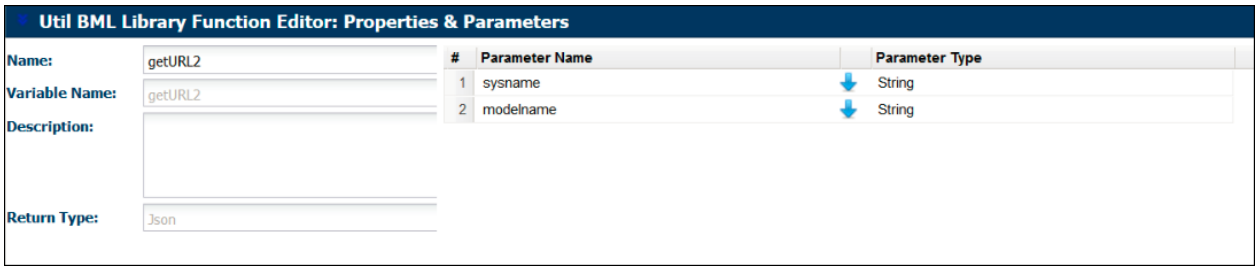


Figure 9: Return Type and Input Information – getURL2

String getTemplate (String sysname)

This function gets the location of the payload template in File Manager to send the payload. This is accomplished by querying the INT_SYSTEM_TEMPLATES data table for a particular system.

getTemplate.txt

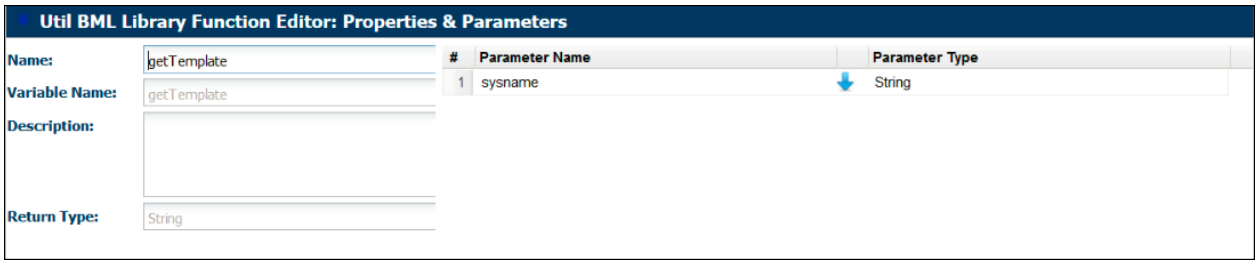


Figure 10: Return Type and Input Information – getTemplate

Json getSystemDetails (String sysname)

This function gets the username, password, and REST endpoint for a particular system from the INT_SYSTEM_DETAILS data table.

getSystemDetails.txt

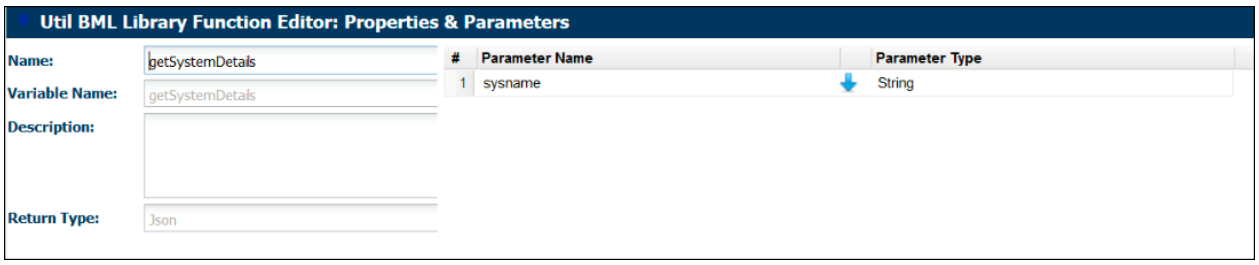


Figure 11: Return Type and Input Information – getSystemDetails

String getParameterID (String endPoint, String payload, String Dictionary dictionary)

This function sends a payload, extracts the parameterID from the response, and returns the parameterID.

getParameterID.txt

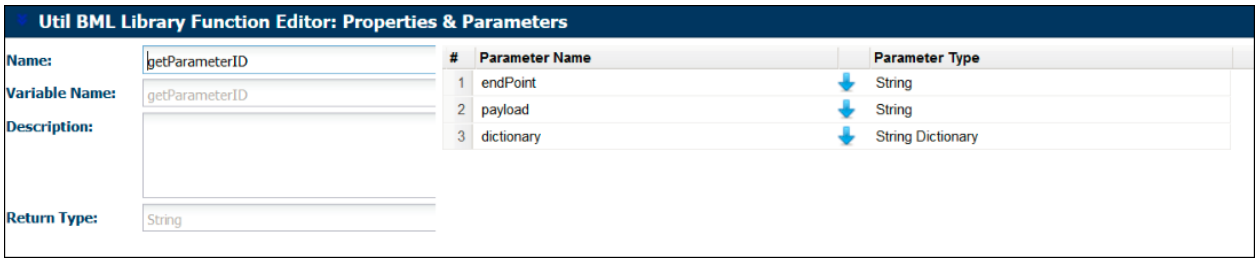


Figure 12: Return Type and Input Information – getParameterID

Json getJsonStatus (String Status)

This function returns JSON objects containing status. If the status contains “cancel” or “error”, the JSON object is returned. Otherwise, the JSON object containing “status” is returned.

getJsonStatus.txt

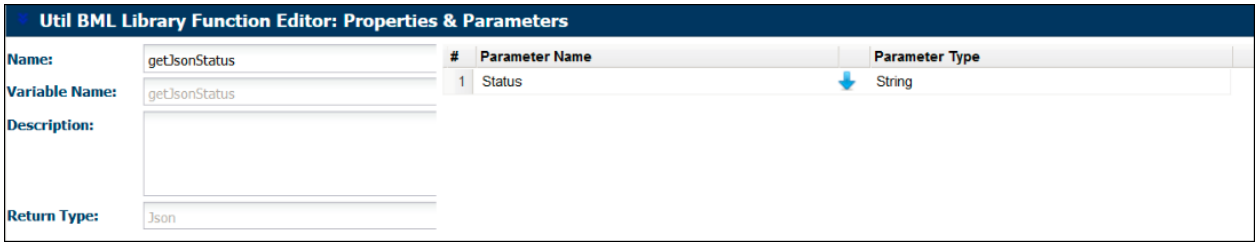


Figure 13: Return Type and Input Information – getJsonStatus

String Dictionary getEncodedDict (String username, String password)

This function returns a dictionary of type String, where the restUsername and restPassword are encoded and mapped into the dictionary.

getEncodedDict.txt

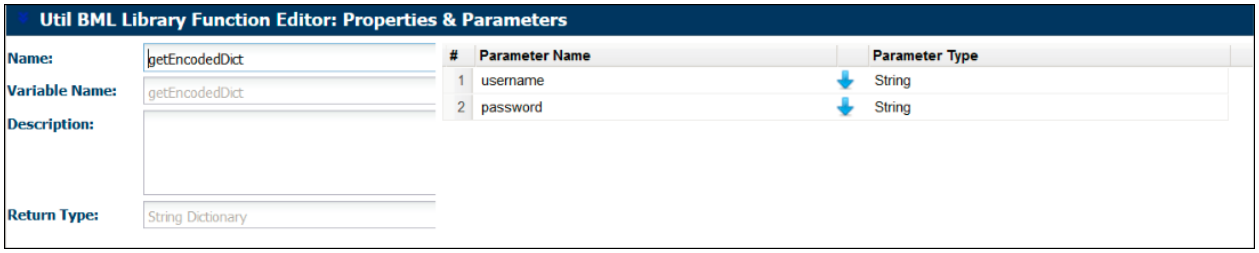


Figure 14: Return Type and Input Information – getEncodedDict

Json constructModelLine (JsonArray jsonObjArr)

This function converts the JSON array “jsonObjArr” into a single “jsonTemp” JSON object and places the id, quantity, and partNumber into a new “jsonModelLine” object. Other mappings such as ConfigHeaderId and ConfigRevisionId are placed into another new “jsonConfigExtraInfo” json object. jsonConfigExtraInfo is mapped into jsonModelLine and returned.

constructModelLine.txt

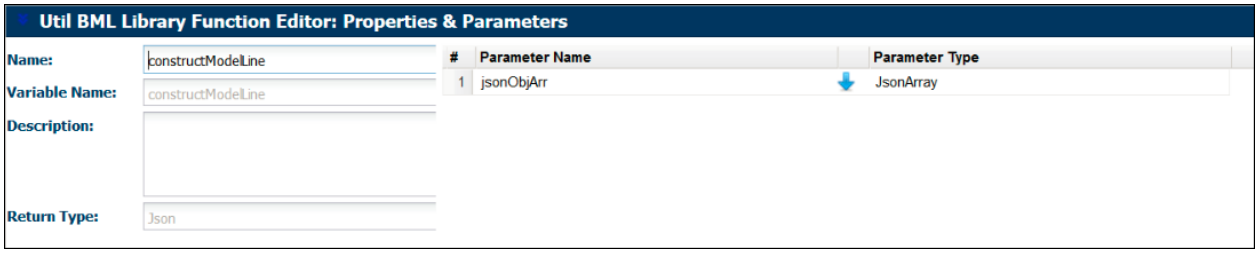


Figure 15: Return Type and Input Information – cconstructModelLine

JsonArray constructJsonChildrenArray (Integer[] intArr, Integer arrSize, JsonArray jsonObjArr, Integer cnt)

This function converts jsonObjArr into jsonTemp objects. In jsonTemp, the id, quantity, parentId, and partNumber are placed into a new “jsonChildren” JSON object. All jsonChildren are appended in the JSON array “jsonChildren”, which is returned.

constructJsonChildrenArray.txt

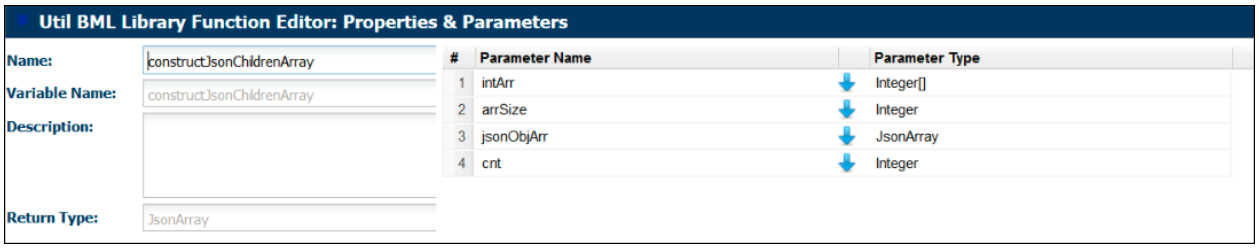


Figure 16: Return Type and Input Information – constructJsonChildrenArray

Json processExtConfigURLParams (String ext_confiig_url_params)

This function extracts ConfigHeaderId and ConfigRevisionId from the String ext_config_url_params. ConfigHeaderId and ConfigRevisionId extract into url_param, and ExitStatus extracts into status.

processExtConfigURLParams.txt

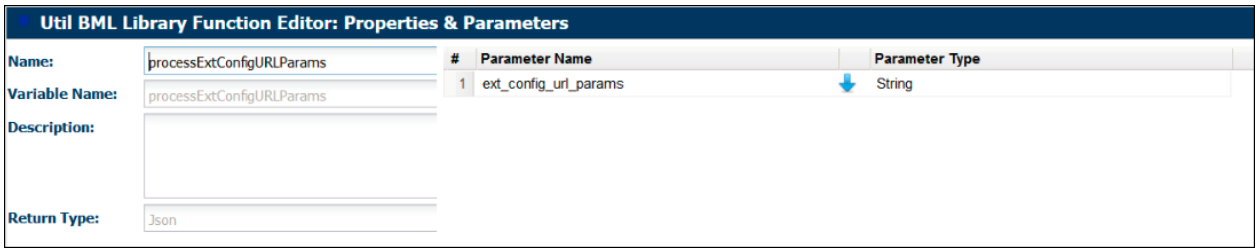


Figure 17: Return Type and Input Information – processExtConfigURLParams

Json getFinalJson2 (String Status, String SegPlineModelVarName, Json jsonModelLine)

This function creates the following three JSON objects: jMsgError, JsonFinal, and jMsgWarning. All error messages map into jMsgError and all warning messages map into jMsgWarning. These two objects append into JSON array “jMessage”. All mappings such as invocationId, priceBookVarName, status, modelline, and jMessage append into jsonFinal, which is then returned.

getFinalJson2.txt

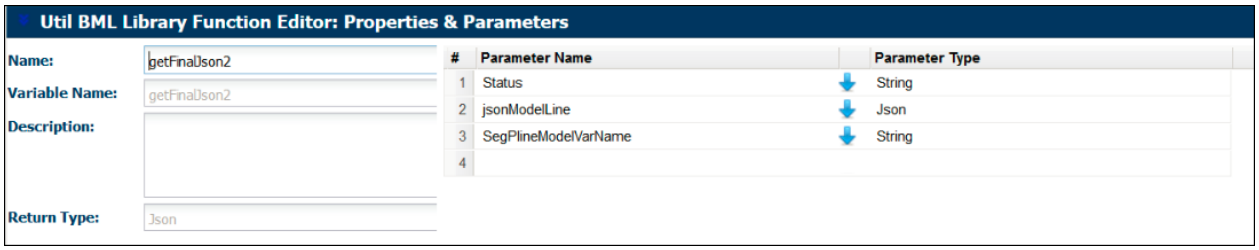


Figure 18: Return Type and Input Information – getFinalJson2

Prepare and Deploy Data Tables and Upload Fusion 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

- Add an inventory Id column to the Parts field
- Use the **Parts Search for Admin** page to upload Fusion parts into CPQ Cloud

Note: CPQ Cloud and Fusion must share a common parts list. Administrators can accomplish this by uploading the Fusion parts that correspond to the CPQ models used in the CPQ Cloud – Fusion Configurator reference integration. For additional information, refer to the *Upload Parts into CPQ Cloud* topic.

Get Fusion Parts from the Fusion Database

Administrators can use the queries provided in this section to get part details from the Fusion database and upload them into the CPQ Cloud reference process. Administrators must query the Fusion database used by their CPQ Cloud – Fusion Configurator integration.

The Fusion database details are as follows:

- Hostname: indl68047.in.oracle.com
- Port: 1522
- SID: in68047
- Username: fusion
- Password: fusion

Queries

Populate the data tables with BOM items from the Fusion database by executing the following BOM explode PL/SQL script:

```
declare
Begin

Egp_Exploder_Pub.Explode(p_inventory_item_id => 137, p_organization_id => 204);

end;
```

Use queries to get parts from the Fusion database.

Query 1: To get the model's item id list:

```
select listagg (INVENTORY_ITEM_ID, ',')
WITHIN GROUP
(ORDER BY INVENTORY_ITEM_ID) ITEM_ID_LIST
from (select distinct pk1_value as INVENTORY_ITEM_ID from egp_explosions_v);
```

Query 2: To get the price details, substitute ":itemlist" with the output from the first query, substitute ":rootItem" with the model root's inventory item id, and substitute ":priceListId" with the price list ID of the model.

```
(Select 'add' as "update", products.* from(select distinct
egp.item_name as "part_number",etl.description "description: en",
etl.inventory_item_id "inventory_item_id",COALESCE(price.base_price,0) "price: USD"
from egp_system_items_tl etl, egp_explosions_v egp,
(select base_price,root.item_id from fusion.qp_price_list_charges ,(select
price_list_item_id pr,item_id from fusion.qp_price_list_items where
item_id=:rootItem and price_list_id=:priceListId)root where PARENT_ENTITY_ID in
root.pr
UNION
select base_price,b.component_item_id from fusion.qp_price_list_charges ,(select
price_list_comp_item_id,component_item_id
from fusion.qp_price_list_comp_items
where price_list_item_id in (select price_list_item_id from
fusion.qp_price_list_items where item_id=:rootItem and price_list_id=:priceListId))b
where PARENT_ENTITY_ID in b.price_list_comp_item_id)price
where etl.inventory_item_id in (SELECT regexp_substr(:itemlist, '[^,]+', 1, LEVEL)
token
FROM dual
CONNECT BY LEVEL <= length(:itemlist) - length(REPLACE(:itemlist, ',', '')) + 1) and
etl.organization_id=204 and etl.language='US' and
egp.pk1_value=etl.inventory_item_id and price.item_id( + ) =
etl.inventory_item_id)products)
UNION
(Select 'update' as "update", products.* from(select distinct
egp.item_name as "part_number",etl.description "description: en",
etl.inventory_item_id "inventory_item_id",COALESCE(price.base_price,0) "price: USD"
from egp_system_items_tl etl, egp_explosions_v egp,
(select base_price,root.item_id from fusion.qp_price_list_charges ,(select
price_list_item_id pr,item_id from fusion.qp_price_list_items where
item_id=:rootItem and price_list_id=:priceListId)root where PARENT_ENTITY_ID in
root.pr
UNION
select base_price,b.component_item_id from fusion.qp_price_list_charges ,(select
price_list_comp_item_id,component_item_id
from fusion.qp_price_list_comp_items
where price_list_item_id in (select price_list_item_id from
fusion.qp_price_list_items where item_id=:rootItem and price_list_id=:priceListId))b
where PARENT_ENTITY_ID in b.price_list_comp_item_id)price
where etl.inventory_item_id in (SELECT regexp_substr(:itemlist, '[^,]+', 1, LEVEL)
token
FROM dual
CONNECT BY LEVEL <= length(:itemlist) - length(REPLACE(:itemlist, ',', '')) + 1) and
etl.organization_id=204 and etl.language='US' and
egp.pk1_value=etl.inventory_item_id and price.item_id( + ) =
etl.inventory_item_id)products);
```

Note: To query parts for root Sentinel Custom Desktop, give inventory_item_id as 137 and PriceListId as 1002. Add "Parts" on top of the CSV file and upload to CPQ Cloud.

Upload Fusion parts to CPQ Cloud

The following table contains sample Excel spreadsheets for uploading Fusion parts to CPQ Cloud.

Fusion Model	Part File
Envoy Custom Laptop: Inventory Item Id-143	envoyCustomLaptopFusion.csv
Sentinel Custom Desktop: Inventory Item Id-137	SentinelCustomDesktopFusion.csv
Printer: Inventory Item Id-4753	printerFusion.csv
Create Your Own Sentinel System: Inventory Item Id-4723	createyourownSentinelSystemfusion.csv

Note: The CPQ Cloud column names included in the above spreadsheets may vary by customer environment. The column names may not apply to all customers. Administrators must map the field name as per their CPQ Cloud environment.

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**.

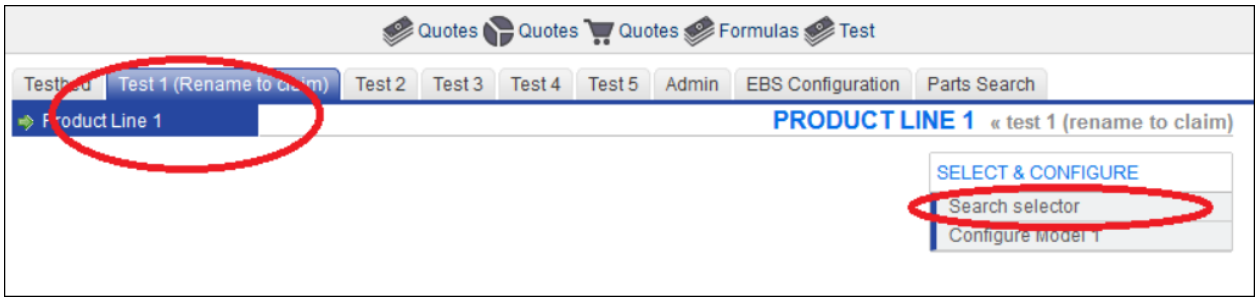


Figure 19: Search Selector

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

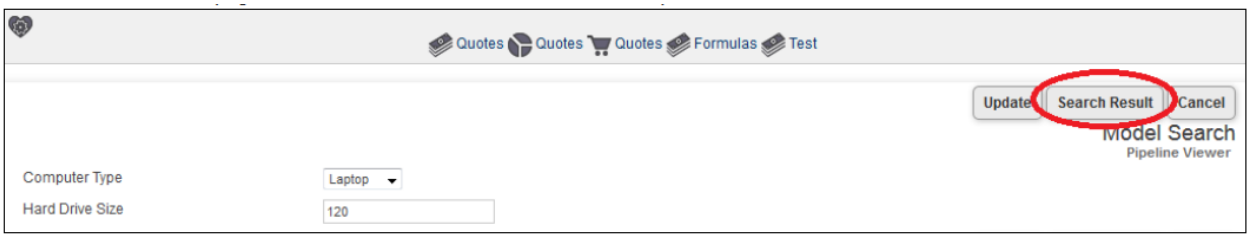


Figure 20: Search Page Displaying Search Attributes

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.

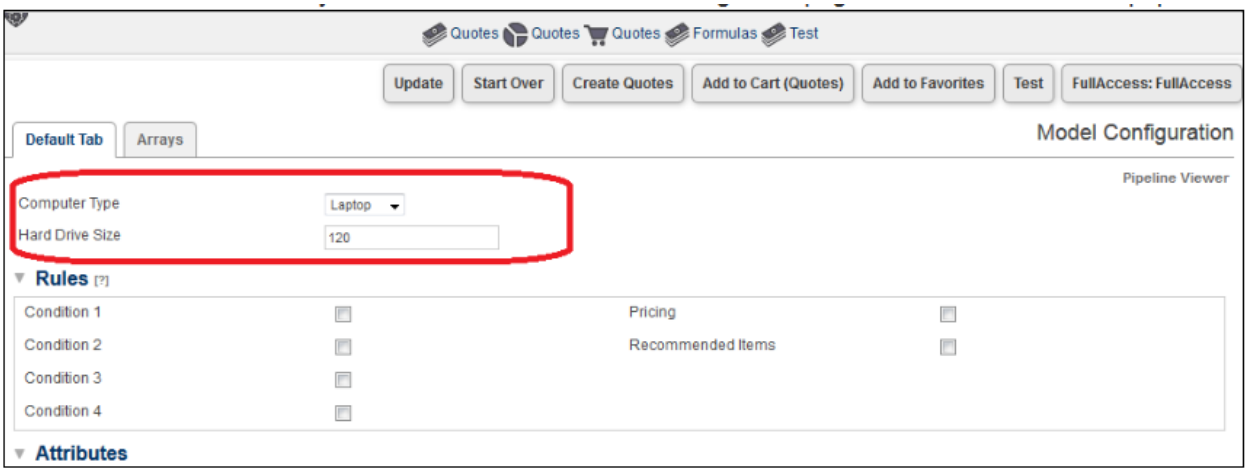


Figure 22: Model Configuration Page

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. In the following screenshot, **Computer Type** and **Hard Drive Size** are the created attributes.

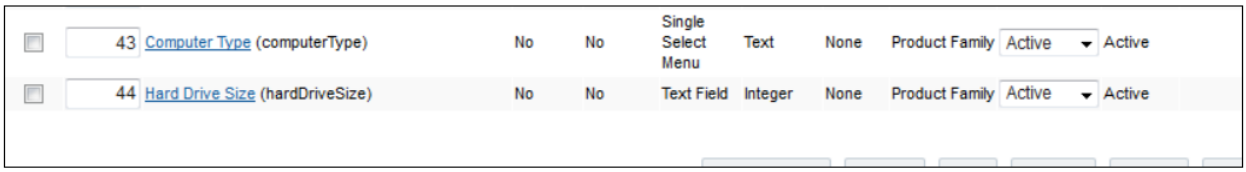


Figure 23: Attribute Editor for Creating Search Attributes for a Selector

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.

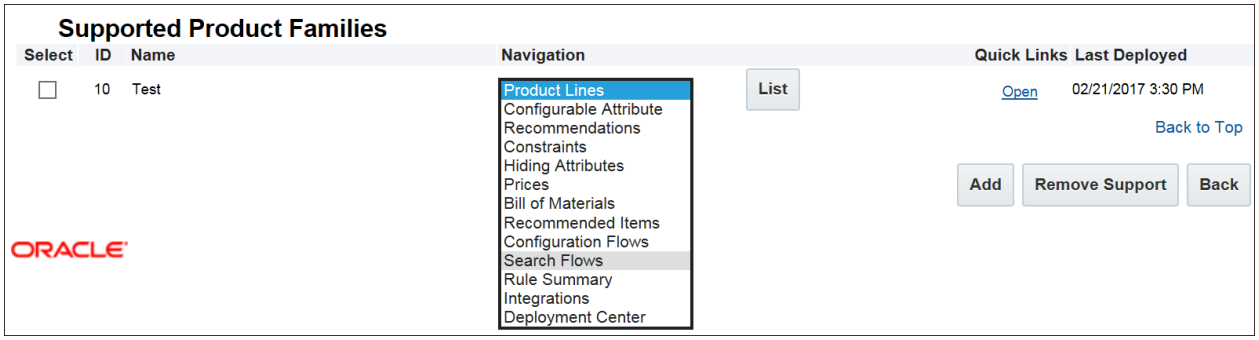


Figure 24: Supported Product Families Page

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.

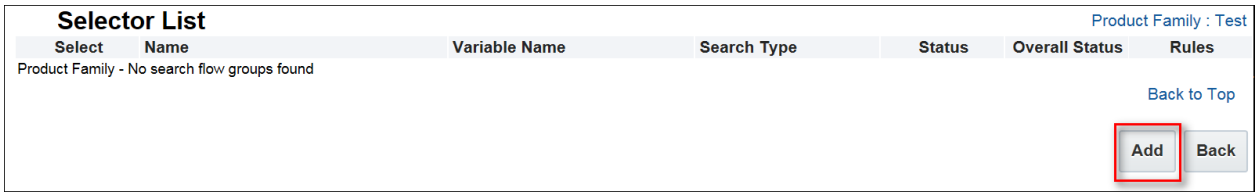


Figure 25: Selector List Page

5. Click **Add**.
   The **Selector Administration** page opens.

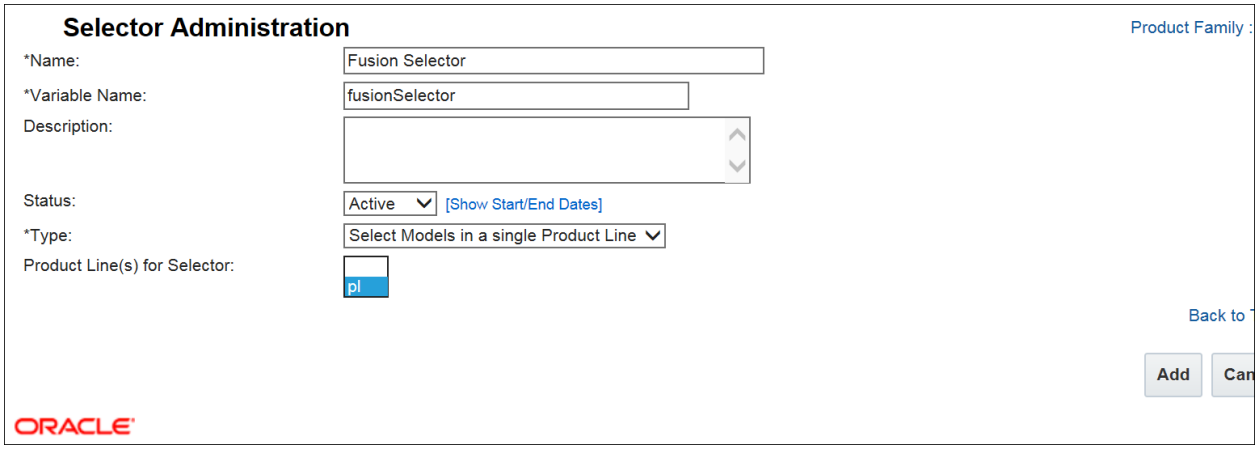


Figure 26: Selector Administration Page

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.

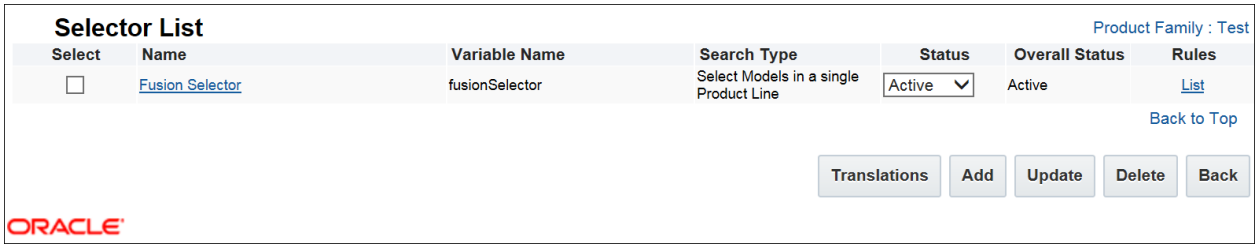


Figure 27: Selector List Page

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

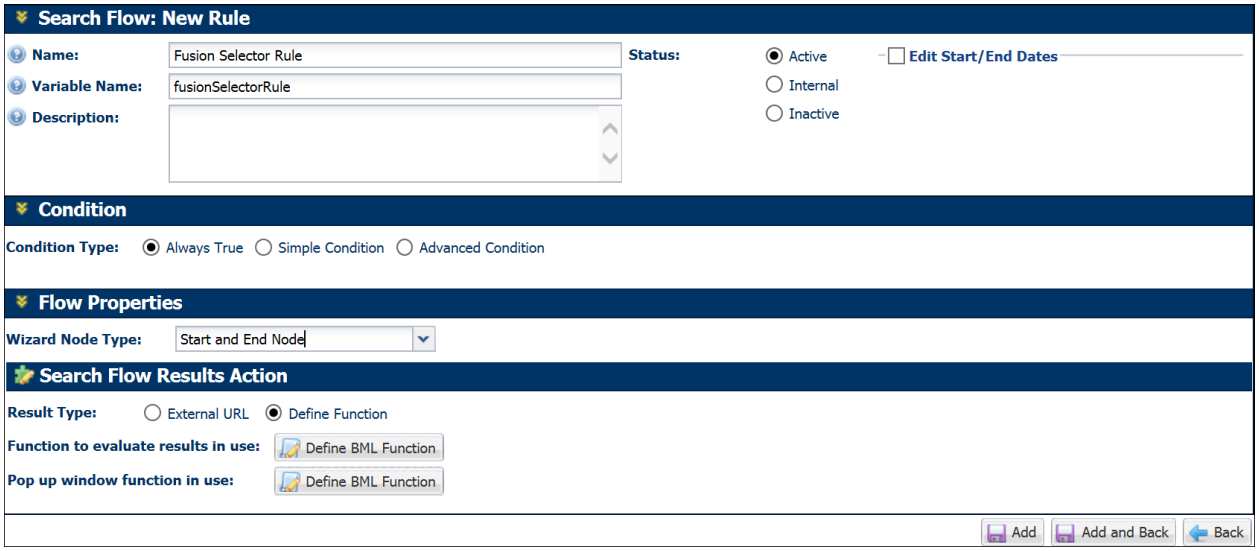


Figure 28: Search Flow Page

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.

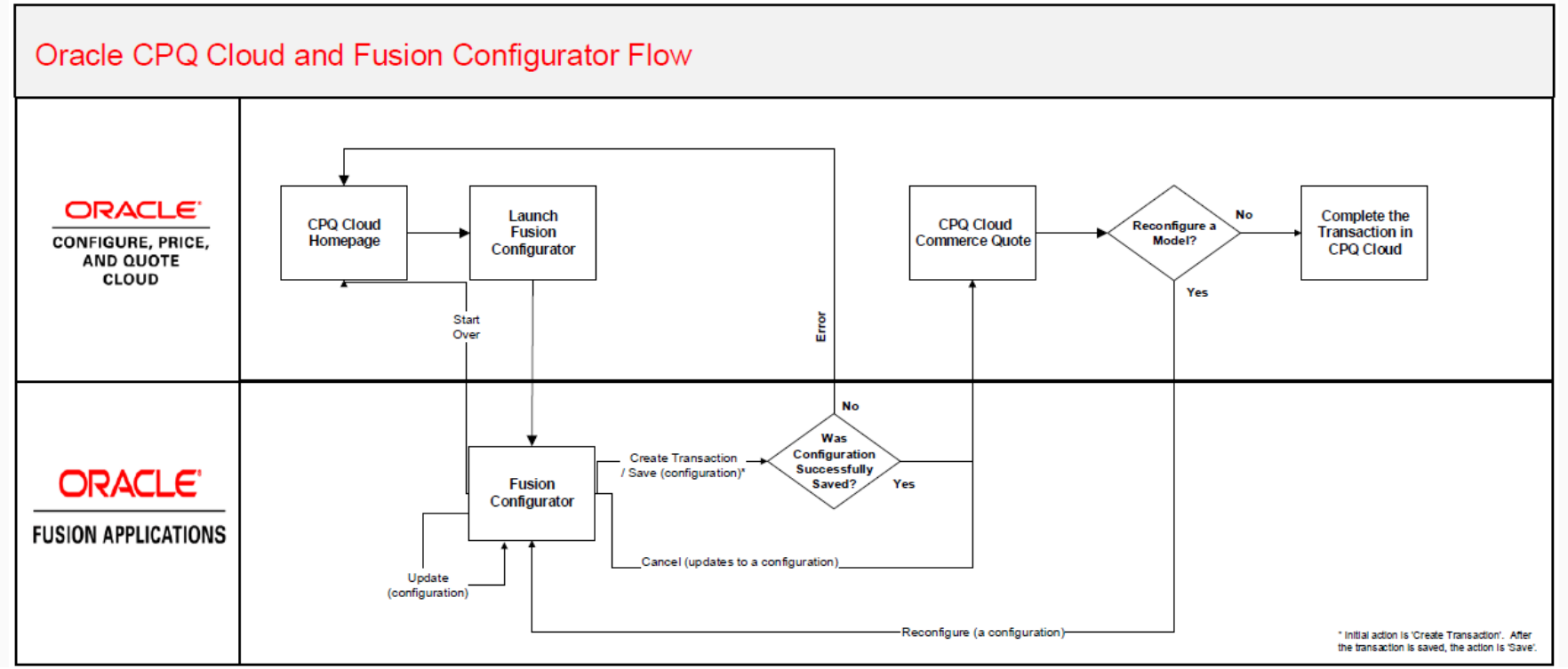
Troubleshooting Tips

The following table contains error messages that administrators may potentially encounter when implementing the CPQ Cloud – Fusion 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.
Partnumber <partnumber> is either not added or associated with the pricebook <pricebook>.	Make sure that all the required parts are added. Also all used parts must be associated with the pricebook mentioned in the code.

Appendix A. Oracle CPQ Cloud and Fusion Configurator Flow

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



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