

PRODUCT LIFECYCLE MANAGEMENT

PLM for Admins

Release 9.3-MP3 (April 2020)

Table of Contents

Table of Contents	2
PLM for Admins	5
Context, Action Sets, and User Groups	6
Context	6
Action Sets	6
User Groups	7
Line Plan and Schedules	10
Configuration Setup for Line Plans and Schedules	13
Setting Up Line Plans in Workbench	15
Creating Line Plan Object Types	16
Creating Primary Product Classification Object Types	16
Creating Alternative Classification Object Types	17
References Needed for Line Plans	19
Product To Classification Link Type	19
Entity Reference Type	20
Setting Up Attributes for Line Plans	25
Line Plan Setup in Web UI	35
Adding Line Plans to a Links Widget	36
Adding a Children Of Types Screen	40
Adding a PLM Line Plan Screen	50
Adding a Classification Screen	56
Setting Up Schedules in Workbench	66
Creating Schedule Object Types	67

Creating PLM Schedules	67
Creating Schedule Template Tasks	69
References Needed for Schedules	73
Entity Reference Types	73
Schedule Template	73
PLM Schedule Template Task Dependency	75
Schedule Task Dependency	76
Product Reference Type	77
Schedule Task	77
Setting Up Attributes for Schedules	81
Scheduling Method and Scheduling Date	82
Scheduling method	83
Planned Start and Planned End	87
Planned Duration	91
Schedule Is Submitted	94
Actual Start and Actual End	97
Schedule Deadline	99
Workflow and State ID	100
Schedule Setup in Web UI	104
PLM Schedule Template	104
Private Label Food Solution	113
Change Reports	114
Change Report Limitations	114
Set up performed in Workbench	114
Set up performed in Web UI	114
Object Types, References, and Attributes for Change Reports	116
Classifications	116

Products	117
Reference Types	117
Classification Reference Types	118
Attributes	118
Snapshot Configurations	120
Create Snapshot Configurations	121
Snapshot Recorders	122
Create Snapshot Recorders	123
Business Rules and Snapshots	126
Prerequisites	126
Snapshot Bind	126
Creating Snapshots	126
Flagging Changes	129
Elements in a Snapshot	132
Private Label Food Solution Setup in Workbench	134
Creating Private Label Food Solution Object Types	135
Creating Primary Product Classification Object Types	135
Additional Primary Product Hierarchy Object Types	137
Creating Alternative Classification Object Types	139
References Needed for Private Label Food Solution	143
Product Reference Types	144
Additive	144
Ingredient	145
Parameter	146

Requirement	147
Sample	148
Specified Additive	149
Specified Ingredient	149
Sub-Ingredient	150
Sub-Ingredient Additive	151
Supplier's Equivalent	151
Classification Reference Types	153
Additive Class	153
Ingredient Options	153
Selected Supplier	154
Ingredient Specification Group	154
Product to Classification Link Types	156
Additive Classification	156
Ingredient Classification	156
Label Country and Languages	157
PLM Product Category	158
Required for Geography	158
Supplier Link	159
Attributes Needed for Private Label Food Solution	160
Attribute Groups	160
Attributes	162
List Of Values for Attributes	170
Private Label Food Solution Setup in Web UI	172
Adding the Compare BOMs Action Button	173
Adding the Compare BOMs Action	173

Adding and Mapping Tabs for Private Label Food Solution	176
Adding Tabs	176
Mappings	179
Specify Ingredients Tab	179
Supplier Ingredients Tab	181
Compare Tabs	183
Configuring the Specify Ingredients Tab	185
Language Translations	187
Copying Pre-translated Ingredients	188
Post Copy Business Action	190
Configuring the Supplier Ingredients Tab	191
Language Translations	193
Copying Pre-translated Ingredients	194
Post Copy Business Action	196
Configuring the Compare Tabs	197
Compare Ingredients Tab	197
Compare Parameters Tab	199
Compare Requirements Tabs	202
Configuring Project Navigator	205
Active Projects Flipper	209
Current Project Flipper	209
Groups	210
Group Supplier Responses By Supplier	213
Related Supplier Responses	214
No Suppliers Group Name	218
Related Groups Attributes	219
Storyboard Setup	222

Configurations, Object Types, and Business Rules	223
PLM Configurations Setup	223
Creating PLM Storyboard Object Types	224
Adding a Hero Image Business Rule	226
Setting Up Storyboard Attributes	232
Storyboard Attributes	232
Creating Tag Attributes	237
Creating the Tag Multi-Valued Attribute	237
Creating Storyboard Assets	239
Creating Storyboard Collection Filters	242
Option to Use Concatenated Attribute Values as Collection Filters	243
PLM Reference Types	246
Creating a Reference Type	247
Multi-Reference Editor in PLM	249
Prerequisites	249
Configurations	249
PLM Create References Action	251
PLM Edit Reference Action	257
Reference Metadata Flex Value Header Component	264
Prerequisites	264
Defining Attribute	264
Configuration	264
Setting Conditional Settings in a MRE	267

PLM for Admins

Welcome to PLM where concepts can be captured, designed, created, and managed all the way to the product's retirement. PLM allows for quick adaptability to any changes that should arise during the production process, ensuring that timely responses can be made.

The following documentation describes the setup needed in workbench and Web UI. If the steps are not followed (including naming conventions, except where notated), then PLM will not function as expected.

Users of PLM will not see the workbench, and they will not see the designer in Web UI, but will only interact with PLM through the web. Proper workbench and Web UI setup is vital for the user to have a seamless experience while working.

Important: It is recommended that admin users read through the **PLM for Users** documentation before starting setup. It is important to understand how PLM works prior to beginning any configuration. Additionally, it is important to be familiar with standard STEP functionality.

Context, Action Sets, and User Groups

Setting up the proper context, action sets, and user groups are all important for helping to maintain a properly functioning system. Through context, data can be organized, and with action sets and user groups, it is possible to maintain who has access to what data. It is important that before any configuration for PLM takes place, that the following context and any needed user groups are created.

Context

A context is a specific filter placed on data in the workbench which groups a set of dimension points, allowing data to vary based on context. Each context is a combination of different dimensions, such as language and country. Only one dimension point from each dimension can be associated with a specific context. While there can be multiple contexts in workbench, PLM will only operate in the context that is specified in the uploaded configuration document.

Initially, the context ID in the configuration documents is labeled 'Context1.' If this needs to be changed, notify your implementation team.

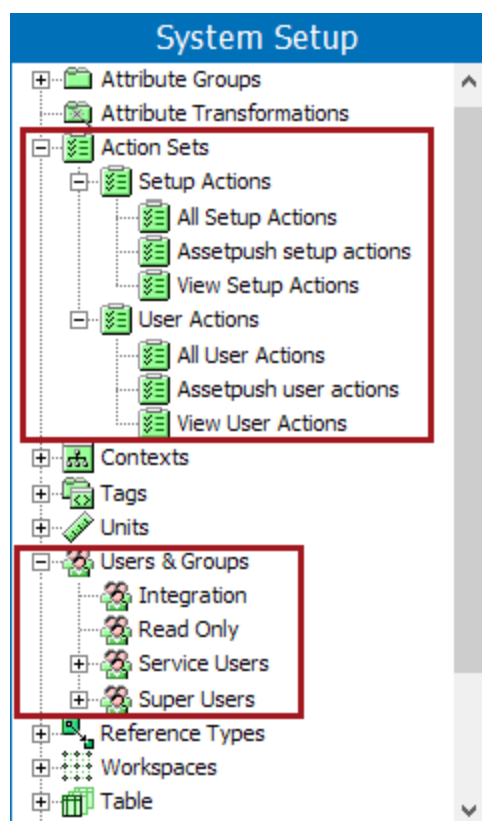
ID	Name	Purpose	Locale	Language	Country
Context1	English US		English - en	English	USA

[Add New Context](#)

To learn more about contexts and how to set one up, see the **Contexts** topic in the **System Setup / Super User Guide** documentation.

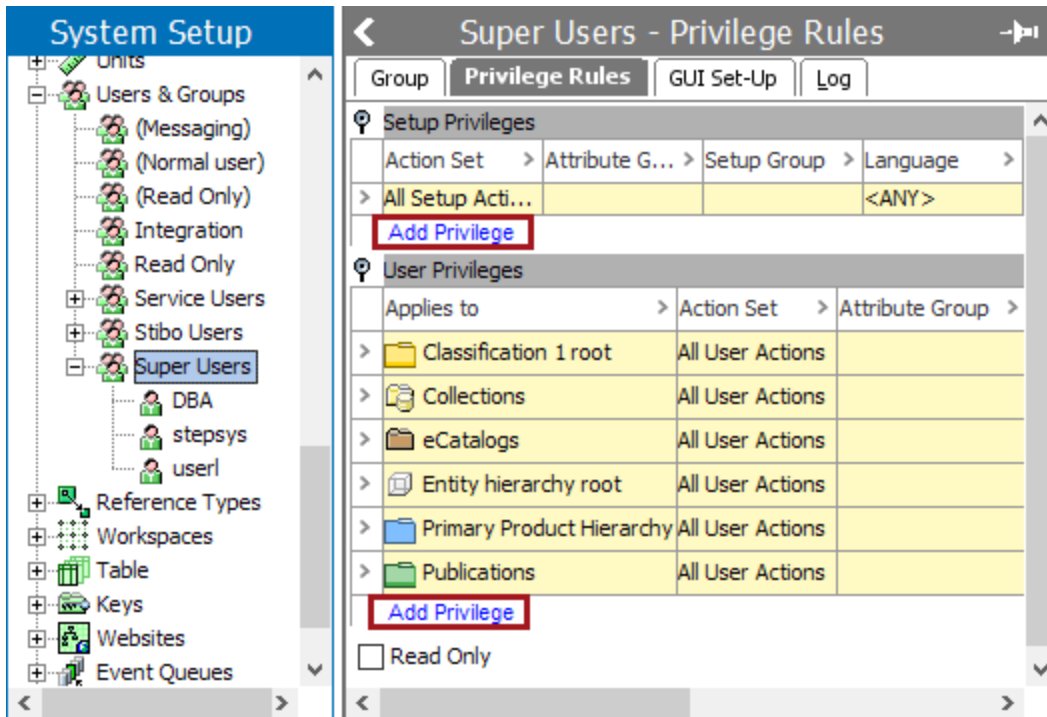
Action Sets

Action Sets are used to define the actions or privileges a user or group of users are permitted to perform in STEP and PLM. Actions are grouped into setup and user Action Sets and can be tied to user group privileges for the user groups under the Users & Groups setup area. For more on how to set up proper action sets and apply privileges, see the **Action Sets** topic in the **System Setup / Super User Guide** documentation.



User Groups

In the Users & Groups hierarchy, you can create STEP roles and authorizations to determine who can access the STEP system and what they are able to do or not do as a user in the workbench, Web UI, or in PLM. Well formed action sets allow administrators to create needed user groups with proper settings.



Make sure that each user in the user group has a valid email address. Various notifications are often sent to the user through their email address. One such notification would be that if a user is using storyboards, they will receive an email each time they are mentioned in a comment on a storyboard. The email notification to the mentioned user will include the message and the storyboard URL so that the individual can quickly navigate to that storyboard and read the comments thread.

Laura Torri - User

User | GUI Set-Up | System Settings | Log

Description

Name	Value
ID	LAURATORRI
Name	Laura Torri
E-Mail	lati@stibosystems.com
Force Authentication via S...	☐
User Information	abc

[Change User Password](#)

Groups

- Stibo Users
- [Add User to Group](#)

Comments

stepsys **2** last month
Laura Torri do you need the new slide deck for this?

Laura Torri last month
 Yes that would be great! Thank you

Notification Card:

Reply | Reply All | Forward | IM

Wed 6/28/2017 2:25 PM

noreply@stibosystems.com

stepsys mentioned you on Nature's Designs

To **Laura Torri** **3**

Laura Torri <lati@stibosystems.com> do you need the new slide deck for this?

Join the conversation at
<http://tom.scloud.stibo.com:80/spireplm/board/board-102986/content>

If there is no email in place under the description flipper on their profile, they will not receive these notifications.

For more information on users and groups see the **Users and Groups** topic in the **System Setup / Super User Guide** documentation.

Line Plan and Schedules

Line planning is a discipline for the development of products. Line plan managers closely manage the development of line plans to ensure that they focus on the right products, meet key deadlines, and achieve goals set for cost and profit margins. Schedules are a way to manage the deadlines and goals set in the line plan.

When making a product for distribution, it is important to have a clear line plan and schedule. PLM assists managers in making decisions about whether or not a product should be produced once market analysis has been completed, and assists managers in making sure that a product goes to market on time. It also aids in identifying which tasks are taking longer than the planned duration, enabling managers to focus on process improvements.

← Christmas 2018

Line Plan Business Cat...

Apparel

Line Plan Year:

2018

Line Plan Season:

Holiday

Line Plan Start Date:

2018-03-06

Schedule Deadline:

2018-04-06

Line Plan Status:

In Progress

Edit Line Plan

707 / 240000

623 / 120

	Item Quantity	Item Unit	Actions
^ Men Jackets (9 styles) Pants (2 styles)	707	623	— + ✓

Christmas 2018

2018

JAN

FEB

MAR

APR

● Men - Jackets →

Design 0/9

Samples 0/9

Delivery 0/9

Design 3/9

● Men - Pants →

Design Brief 0/1

Design 0/1

Samples 0/1

Delivery 0/1

For more information on how to setup line plans and schedules, see the following documentation.

- [Setting Up Line Plans in Workbench](#)
- [Setting Up Schedules in Workbench](#)

The following licenses and components are needed for line plans and schedules to work:

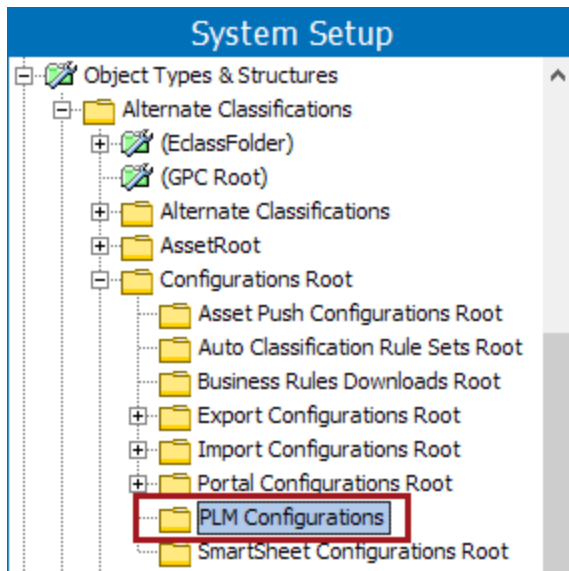
Required License	Components needed
X.STEPIllustrator Note: Only needed if also utilizing the ability to integrate with Adobe Illustrator	illustrator Note: Only needed if also utilizing the ability to integrate with Adobe Illustrator
N/A	spire-plm

Configuration Setup for Line Plans and Schedules

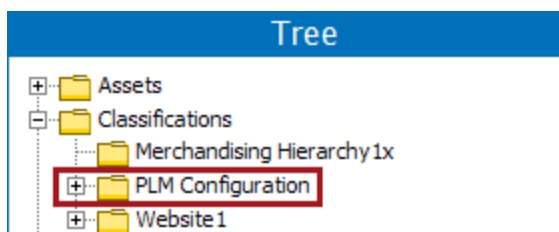
The following steps will upload the configuration file that the line plans and schedules need to function properly, if it has not already been uploaded to your system.

Important: Talk to your implementation team to discuss a configuration strategy before uploading the provided configuration files and configuring anything else for PLM line plans or PLM schedules. Proper ID alignment needs to be established when uploading this configuration file and when configuring any other line plan or schedule objects in workbench. If there is improper alignment with IDs, then the line plan or schedule will fail.

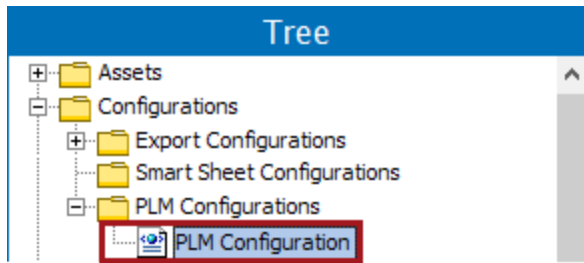
1. In **System Setup** > Object Types & Structures > Alternate Classifications > Configurations Root, create a folder to contain all PLM configurations.



2. Next, in **Tree**, below the classifications node, create a new classification folder to contain the PLM Configuration.



3. In this folder, import the PLM configuration file using the file name as the STEP ID. Make sure that all changes that are needed to be made to the PLM configuration file are made before the file is uploaded. This includes changing any IDs necessary; otherwise, the line plan or schedule will not work properly.



Note: Depending on your implementation team, these steps may or may not be handled by them. If the implementation team does not handle the steps above, they will be the ones to provide the configuration files.

Setting Up Line Plans in Workbench

For line plans to work correctly in the Web UI, proper setup needs to happen in the workbench, and the PLM configuration file needs to be uploaded. If the PLM configuration file has not already been uploaded to your system, see the **Configuration Setup for Line Plans and Schedules** topic in this documentation.

Read the following topics on how to set up line plans properly in the workbench:

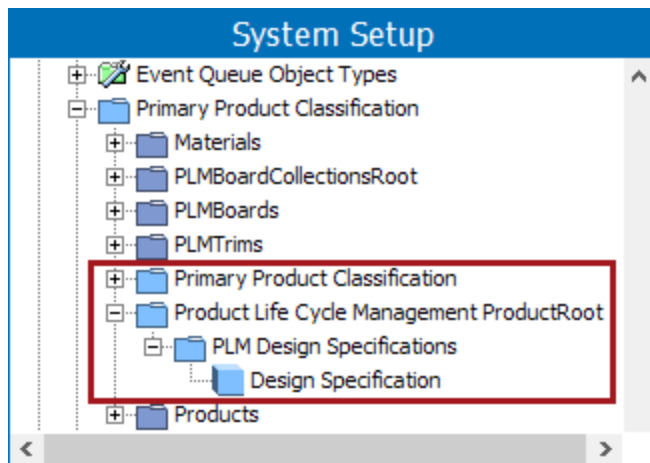
- Creating Line Plan Object Types
- References Needed for Line Plans
- Setting Up Attributes for Line Plans

Creating Line Plan Object Types

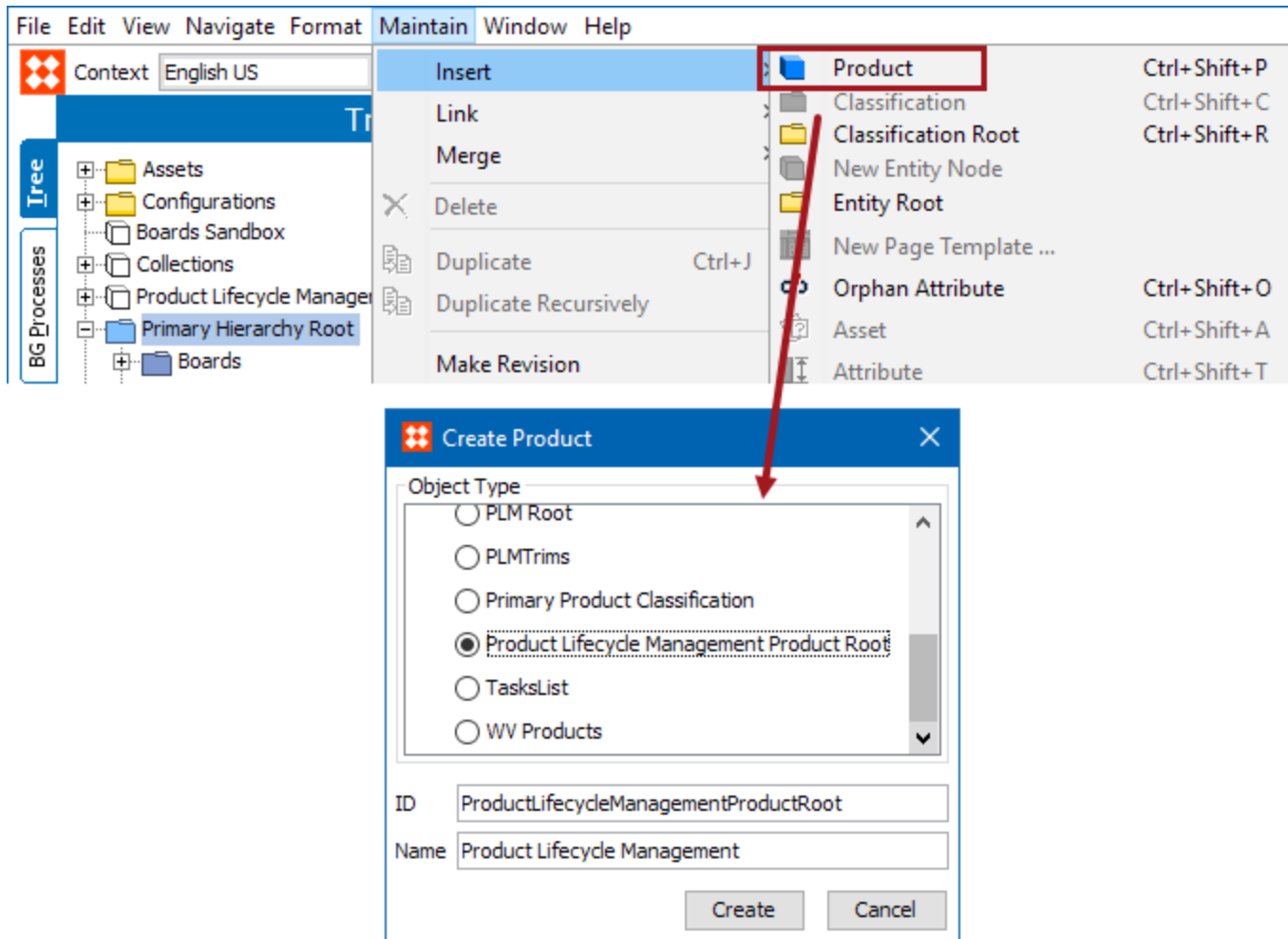
There are a number of object types necessary for line plans to function properly in Web UI. The following sections describe what Primary Product and Alternative Classifications object types are needed.

Creating Primary Product Classification Object Types

Design specifications live in the Primary Product Classification object type structures. A simple hierarchy structure with one parent needs to be created to house the design specifications uploaded from the Web UI. For more on how to create design specifications in Web UI, see the **Creating Line Plans** topic.



After the structure had been created in System Setup, it needs to be added to Tree. In Tree, click on a Primary Product Classification folder and, go to Maintain > Insert > Product > select the object type to hold the design specifications.

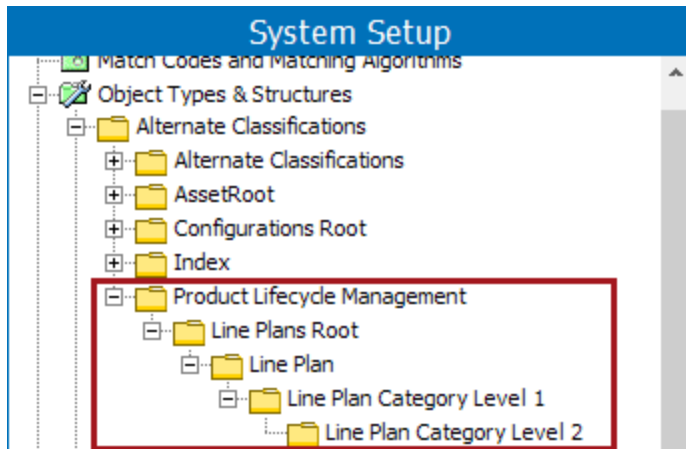


Once the hierarchy structure for design specifications has been created and added to Tree, a reference needs to be created between the line plan and design specifications. See the **References Needed for Line Plans** topic in this documentation.

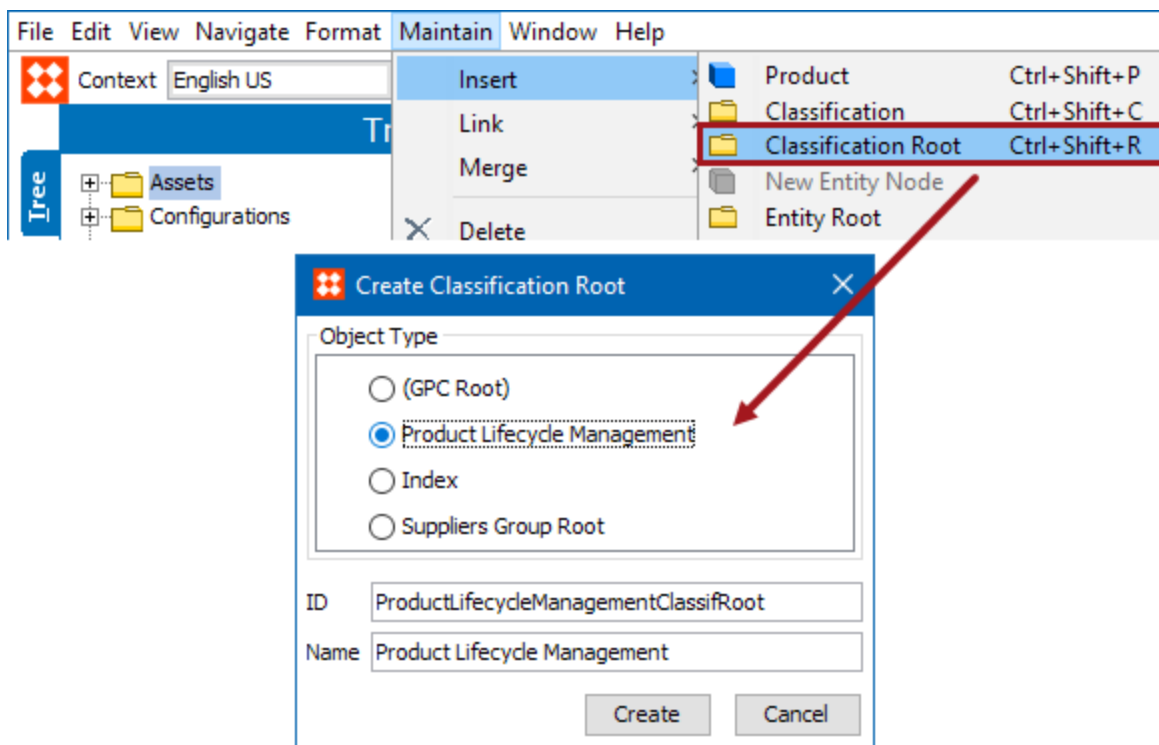
Creating Alternative Classification Object Types

Alternative Classifications are where design specifications are referenced and grouped into a folder for easy line plan use. The Product Lifecycle Management Classification root folder will need to be created for all other line plan roots to live under. Each line plan root can have up to three levels.

When finished, the line plan structure will look similar to the one below.



After the structure had been created in System Setup, it needs to be added to Tree. In Tree, click on a Classification folder and, go to Maintain > Insert > Classification Root > select the object type to hold line plans.



Each line plan will need to have attributes according to business needs. To set up needed attributes for line plans, see the **Setting Up Attributes for Line Plans** topic in this documentation.

References Needed for Line Plans

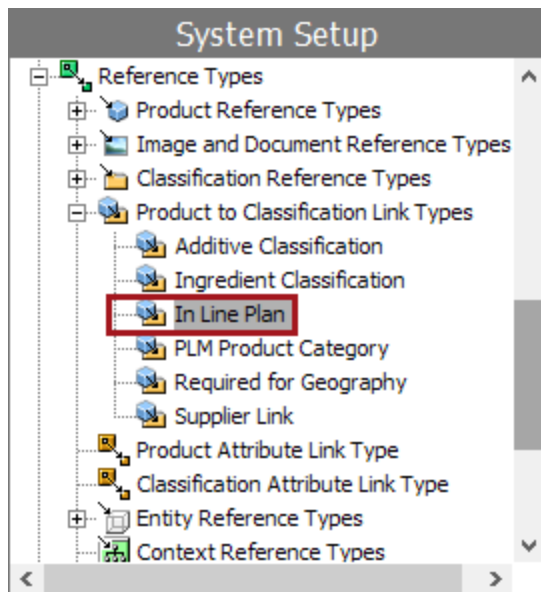
There are two different reference types required for line plans to work properly:

- Product to Classification Link Type
- Entity Reference Type

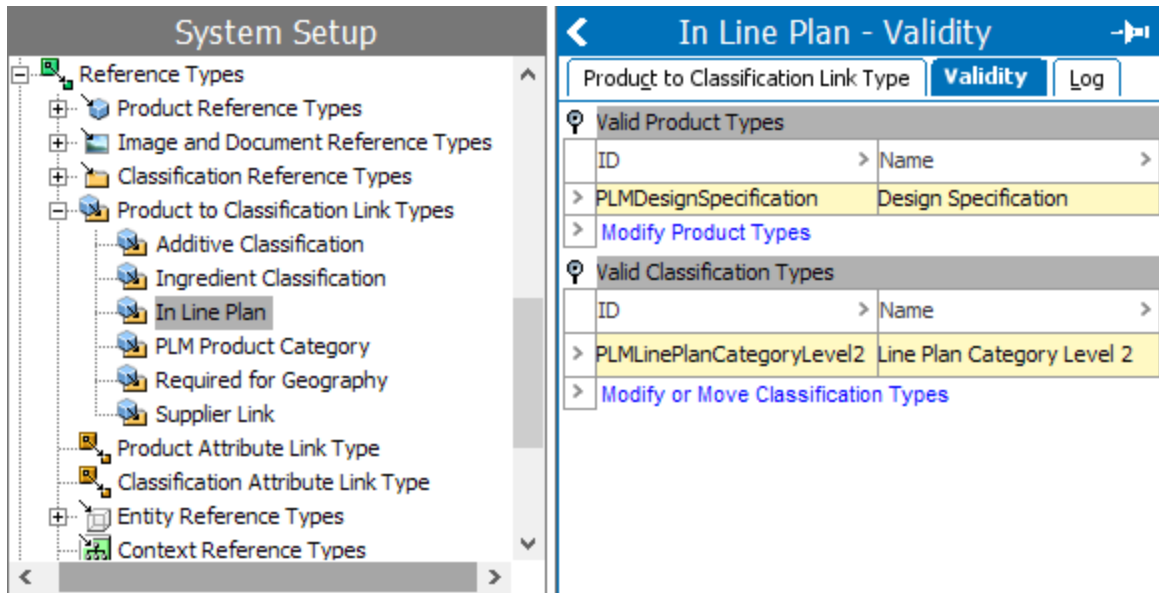
They are needed to link line plans to design specifications and line plans to schedules. This topic describes how to create the necessary reference types.

Product To Classification Link Type

The Product to Classification Link Type is needed as a reference between the line plan category level two and the design specification document. For this Product to Classification Link Type, an 'In Line Plan' reference was created.



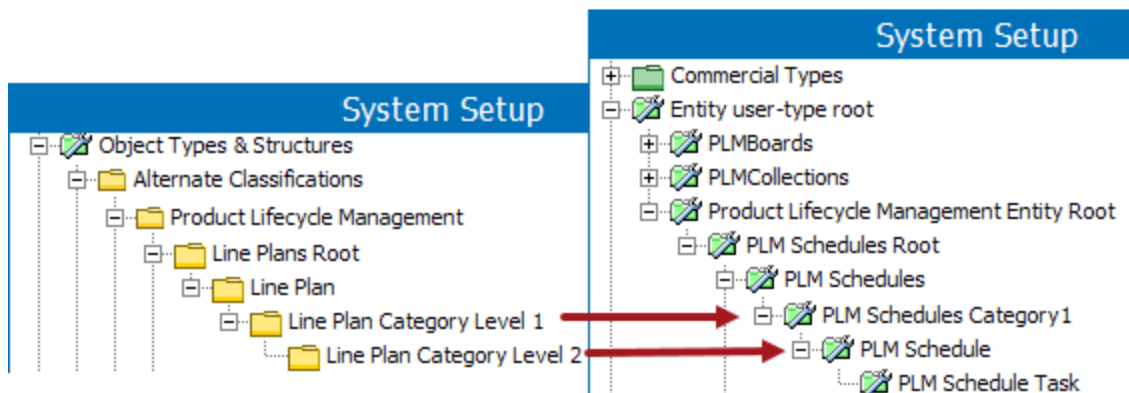
On the Validity tab for this new reference, select the design specification object type for the Valid Product Type and select the Line Plan Category Level 2 object type as the Valid Classification Type.



Entity Reference Type

There are two entity reference types needed to connect the line plan to the schedules:

- A reference to connect the classification object type for Line plan Category Level 1 to entity type PLM Schedules Category 1
- A reference to connect the classification object type for Line Plan Category Level 2 to entity type PLM Schedule



PLM Line Plan Category 1 to Schedule Category 1

A reference to connect the classification object type for Line plan Category Level 1 to entity type PLM Schedules Category 1 is needed when transitioning from line plans to schedules. When an approved line plan displays the 'Schedule' hyperlink, the reference enables the user to navigate to the appropriate schedule.

← Christmas 2018

Business Category: Apparel

Line Plan Start Date: 2018-03-06

Schedule Deadline: 2018-04-06

Line Plan Status: In Progress

Edit Line Plan

1901 / 120

6586

8884 / 2000000 0 (Inc

	Item Unit	Cost	Revenue	Actions
^ Children	24	22	14	Schedule
T-shirts (1 styles)	24	22	14	

← T-shirts

Start from date ▼ 19. April, 2038

Apparel Template (Make) ▼

Design Brief

19/04/2018 - 02/05/2018

10

Design

03/05/2018 - 30/05/2018

20

Samples

31/05/2018 - 11/07/2018

30

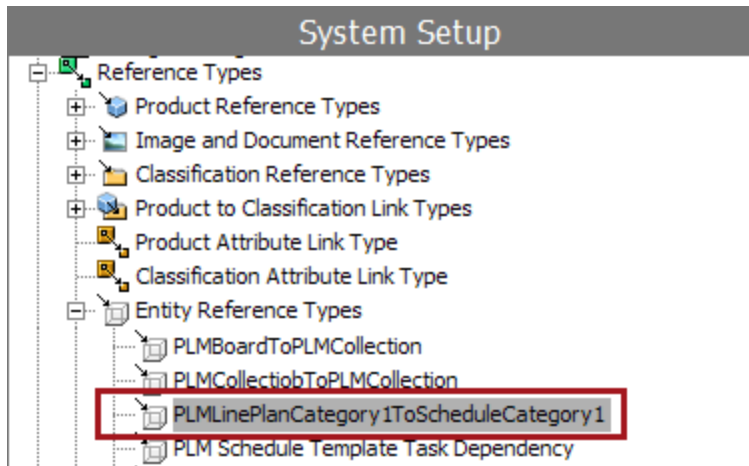
Submit

2018

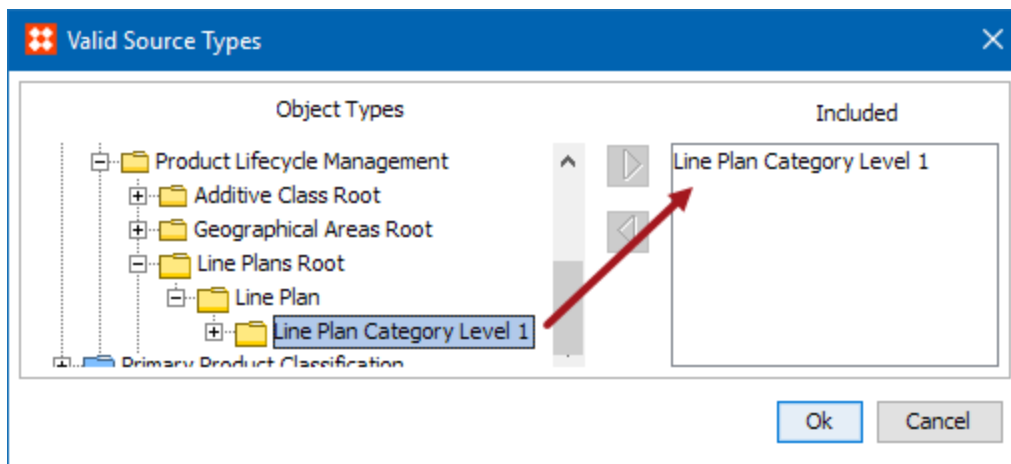
MAY JUN

Design	Design	0/0	Samples

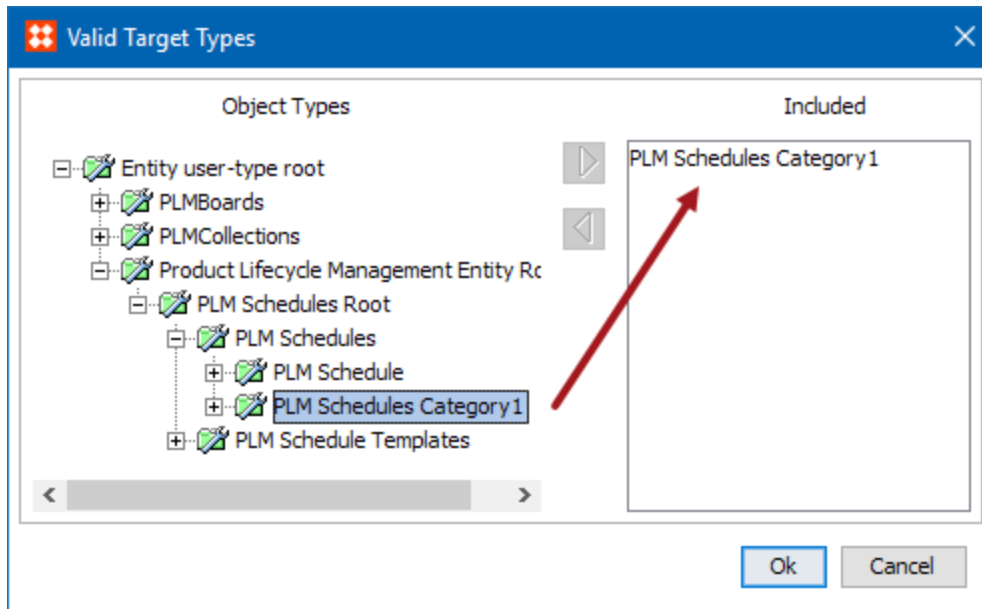
For this example it is 'PLMLinePlanCategory1ToScheduleCategory1.'



This new reference should have the first level of all line plans as a valid source type. In this example it is 'Line Plan Category Level 1.'



For the target type, make sure that it is valid for the first level of schedules. In this example it is 'PLM Schedule Category1.'



It is important that the ID of this Entity Reference is correctly put into the uploaded PLM configuration file. If the uploaded information is incorrect, then PLM may not work properly.

PLMLinePlanCategory1ToScheduleCategory1 - Reference Type

Reference Type	Validity	Log
Description		
Name	>	Value
ID		PLMLinePlanCategory1ToScheduleCategory1
Name		PLMLinePlanCategory1ToScheduleCategory1
Last edited by		2018-04-10 13:12:13.0 by STEPSYS

```

{
  "category": {
    "approvedAttributeId": "PLMLinePlanCategoryApproved",
    "scheduleCategoryReferenceId": "PLMLinePlanCategory1ToScheduleCategory1"
  }
},
  
```

A red arrow points from the 'ID' field in the table to the 'scheduleCategoryReferenceId' value in the JSON snippet.

Note: You may need to talk to you implementation team to set the proper ID in the configuration file.

Line Plan Category Level 2 to Schedule level 2

This reference connects the classification object type for line plan category level 2 to entity type schedule level 2. It is needed to correctly create the subcategory from the approved line plan to the schedule subcategory.

← 2019 Summer Mens

Business Category: Apparel

Line Plan Year: 2019

Line Plan Season: Summer

Line Plan Start Date: 2018-02-05

Schedule Deadline: 2018-11-15

Line Plan Status: In Progress

Edit Line Plan

926037200

3222982000 / 150000000

65.29 / 46

	Cost	Revenue	Gross Margin %	Actions
^ Work From Home Collection Bottoms (72 styles) Shirts (32 styles) 11384000 58409000 937278000 349758000 51.15 47.01 Schedule				

← Work From Home Colle...

2018

JAN

FEB

MAR

● Bottoms

● Shirts

→

→

0

Samples 0

To create this entity reference type, make it have the second level of line plans as a Valid Source Type, and the second level of schedules as the Valid Target Type. In the example below, the reference created is called 'Line Plan Schedule.' The second level for line plans was alternate classification Line Plan Category Level 2, and the second level for the schedule entity type was PLM Schedule.

System Setup

Entity Reference Types

Line Plan Schedule

PLMBoardToPLMBoard

PLMBoardToPLMCollection

PLMCollectionToPLMCollection

PLMLinePlanCategory1ToScheduleCategory

PLM Schedule Template Task Dependency

Schedule Task Dependency

Schedule Template

Context Reference Types

Workspace Reference Types

← Line Plan Schedule - Validity

Reference Type

Validity

Log

Valid Source Types

ID	Name
PLMLinePlanCategoryLevel2	Line Plan Category Level 2

Modify Source Types

Valid Target Types

ID	Name
PLMSchedule	PLM Schedule

Modify Target Types

Setting Up Attributes for Line Plans

When creating line plans, there are a number of attributes that need to be set up properly both in workbench and in the uploaded configuration file, in order for line plans to work in Web UI. The sections below detail what attribute setups are need.

Note: Attributes created for line plans can be calculated attributes, regular attributes, or updated through business rules based on your business needs. For the examples that follow, calculated attributes and regular attributes are used.

Line Plan Approval

When viewed in the Web UI, the line plan approval attribute is an attribute that confirms if a line plan is considered complete by clicking the checkmark under the Actions column in the line plan itself.

← Christmas 2018		Line Plan Year:	2018	Business Category:	Apparel
		Line Plan Start Date:	2018-03-06	Line Plan Season:	Holiday
		Line Plan Status:	In Progress	Schedule Deadline:	2018-04-06
Edit Line Plan		1439 / 240000	2861 / 120	39226	56884 / 2000000
		Item Quantity	Item Unit	Cost	Revenue
^ Men		1107	1827	6562	8866
Jackets (9 styles)		602	1226	6562	8866
					— + ☒

This designated attribute for line plan approvals needs to be a description attribute and have an 'List Of Values' as the validation base type. The LOV used for the line plan approval attribute needs to have 'text' as the validation base type and a value of 'Y' in its Values field. In the example below, the 'PLMLinePlanCategoryApproval' attribute is used.

Attribute	References	Attribute Transformation
Description		
Name	>	Value
ID		PLMLinePlanCategoryApproved
Name		Category Is Approved
Last edited by		2018-03-08 13:36:18 CHI
Full Text Indexable		No
Externally Maintained		No
Hierarchical Filtering		None
Calculated		No
Type		Description
Mandatory		No
Attribute Validation		
Name	>	Value
Validation Base Type		List Of Values
List Of Values		Line Plan Approved
Multi Valued		No
Mask		
Minimum Value		N/A
Maximum Value		N/A
Maximum Length		N/A
Edit Validation Rule		

List of Values	References	Log	State Log	Tasks
Description				
Name	>			Value
ID				PLMLinePlanApprovedLOV
Name				Line Plan Approved
Edited by				2018-03-08 13:28:37 by CHI
Path				Lists of Values / LOVs/!Produ...
Dimension Dependencies				Language;
Use Ids on values				Yes
Use Ids for sorting				No
In Attribute Groups				
List of Values Validation				
Name	>			Value
Validation Base Type				Text
Allow Users to Add Values				No
Mask				
Minimum Value				N/A
Maximum Value				N/A
Maximum Length				5
Values				
Values	>			Value ID
Yes				Y
Add Value				

It is imperative that the ID for the line plan approval attribute is entered in correctly to the uploaded PLM configuration file. If it does not match, line plan approvals will not work properly in Web UI.

```
"category": {
  "approvedAttributeId": "PLMLinePlanCategoryApproved",
  "scheduleCategoryId": "PLMLinePlanCategory1ToScheduleCategory1"
},
```

Note: Talk to your implementation team to make sure that the correct attribute ID is used in the uploaded PLM configuration file.

Line Plan Page Headers

In Web UI, the line plan page headers are attributes that provide extra information about the currently viewed line plan. These attribute values are typically filled out when the line plan is being created, and can be tweaked and edited until the line plan is approved.

Edit Line Plan

Line Plan Year
2019

Line Plan Season
Holiday

Line Plan Start Date
05 / 01 / 2018

Schedule Deadline
11 / 30 / 2018

Line Plan Status
In Progress

Target Item Quantity
250

Target Unit Quantity
150

Target Revenue
300

Target Gross Margin %
90

Create

← Formal Mens

Line Plan Year:

2019

Line Plan Season:

Holiday

Line Plan Start Date:

2018-05-01

Schedule Deadline:

2018-11-30

Line Plan Status:

In Progress

Edit Line Plan

0 / 250

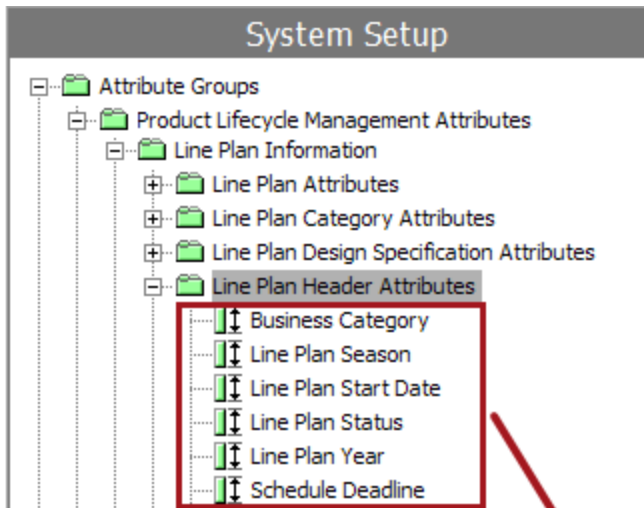
0 / 150

Item Quantity

Item Unit

Actions

Proper validity setup in the workbench needs to take place in order for the correct attributes to appear on the line plan page header in Web UI.



← Christmas 2018

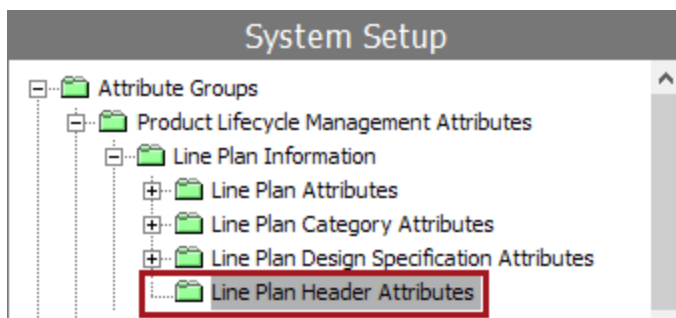
Line Plan Year:	2018	Business Category:	Apparel
Line Plan Start Date:	2018-03-06	Line Plan Season:	Holiday
Line Plan Status:	In Progress	Schedule Deadline:	2018-04-06

Edit Line Plan

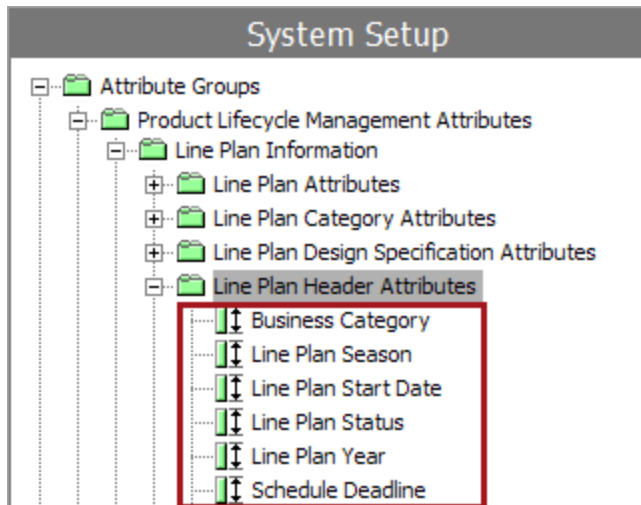
799 / 240000	701 / 120	586	
Item Quantity	Item Unit	Cost	Actions

Men	707	627	562	- + ✓
-----	-----	-----	-----	-------

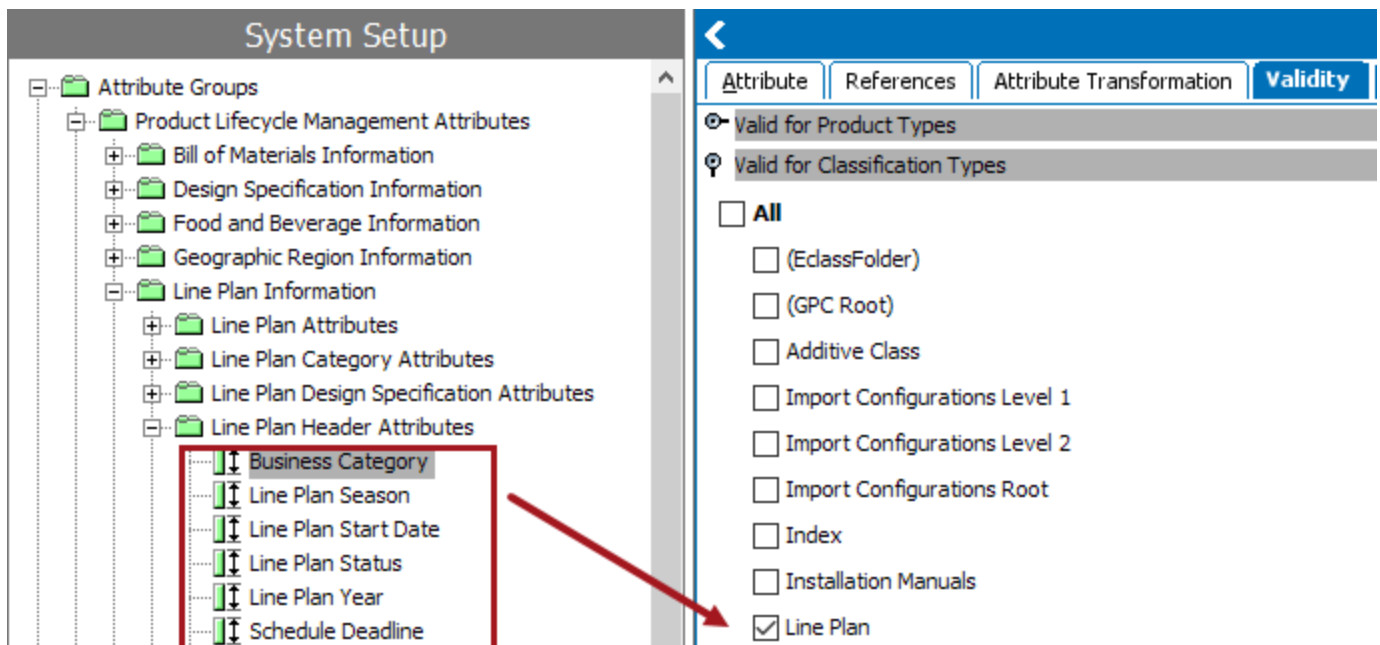
1. In System Setup, create a **PLMLinePlanHeaderAttributes** group to hold all line plan header attributes.



2. Create any children attributes needed for the line plan header in this group. For more on how to create attributes, see the section on **Attributes** in the **System Setup / Super User Guide** documentation.



3. Make the validity for these attributes valid for the object type that represents line plans. In this example, Line Plan is the needed validity to ensure that these attributes display correctly in Web UI.



4. Make sure that the proper attribute group ID is used in the uploaded PLM configuration file. If the IDs do not match, then the attribute group headers will not appear in Web UI correctly.

```
"lineplans": {
  "lineplan": {
    "headerAttributesGroupId": "PLMLinePlanHeaderAttributes"
  },
  "category": {
    "approvedAttributeId": "PLMLinePlanCategoryApproved",
    "scheduleCategoryReferenceId": "PLMLinePlanCategory1ToScheduleCategory1"
```

Note: You may need to talk to your implementation team to ensure that the correct attribute group ID is used in the configuration file. Additionally, should any line plan page headers need to be changed, talk to your implementation team.

Line Plan Column Headers

Note: Attributes created for line plan column headers can be calculated attributes, regular attributes, or updated through business rules based on your business needs. For the examples that follow, calculated attributes and regular attributes are used.

Column headers for line plans are a group of calculated attributes that aid line plan managers in deciding if a particular line plan should be approved and brought to market based on the values in these fields.

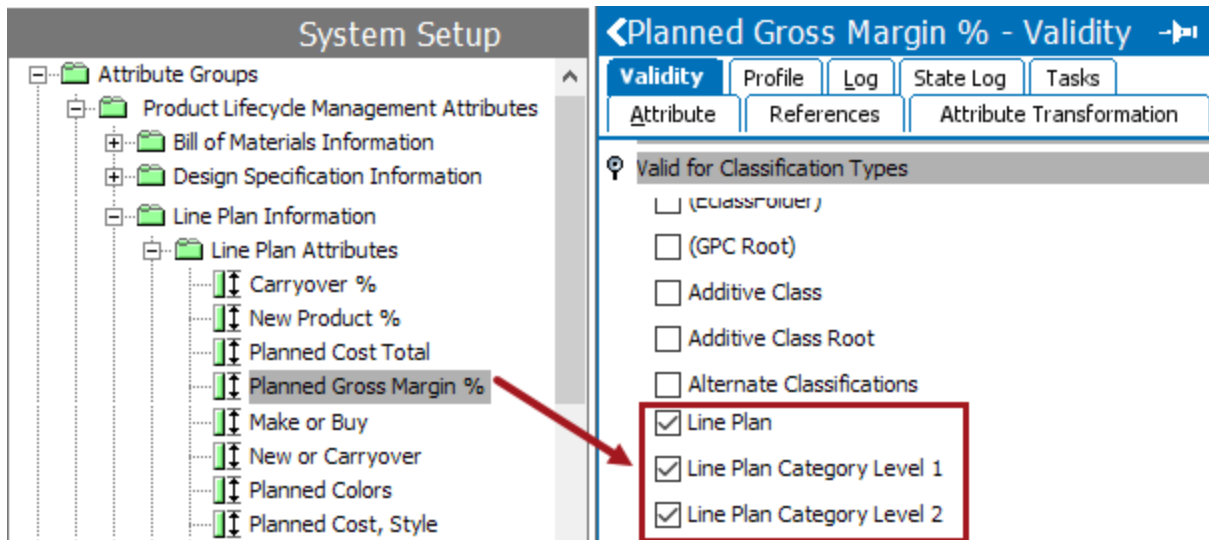
← Christmas 2018		Line Plan Season: Holiday	Business Category: Apparel
		Line Plan Status: In Progress	Line Plan Start Date: 2018-03-06
		Schedule Deadline: 2018-04-06	Line Plan Year: 2018
Edit Line Plan		1439 / 240000	2861 / 120
		39226	56884 / 2000000
		Item Quantity	Item Unit
		Cost	Revenue
		Actions	
Shirts (1 styles)		56	23
		0	0
T-Shirts (7 styles)		327	285
		0	0

To create column header calculated attributes:

1. Go to System Setup > Attribute Groups > create an attribute group to hold line plan column header attributes.
2. In this folder, create the necessary line plan column header calculated attributes.

Note: Some calculated attributes may be uploaded via a PLM data package, and may need to be tweaked for your business needs. If calculated attributes need to be created, talk to your implementation team.

3. Make each calculated attribute valid for all levels of the line plan.



When all needed calculated attributes are added, check that the IDs used for the calculated attributes match those in the uploaded PLM configuration file. Each calculated attribute is grouped in the PLM configuration file to make a complete line plan column header. If the IDs are not entered correctly in the PLM configuration file, the line plan column header(s) will not display or work properly.

Note: Some calculated attributes may be uploaded via a PLM data package, and may need to be tweaked for your business needs. Talk to your implementation team should you need different calculated attributes for the line plan column headers, or if the line plan column headers are not working properly. For more on how to create calculated attributes see the **Calculated Attributes** topic in the **Attributes** section of the **System Setup / Super User Guide** documentation.

Titles for line plan column headers are only able to be set in the uploaded PLM configuration file. In the example below, the Item Quantity line plan column header is represented by the following calculated attributes:

- PLMPlannedItemQuantityTotal - a value of 4079000
- PLMTargetItemQuantity- a value of 100000

```
"lineplans": {
  "lineplan": {
    "headerAttributesGrp": {
      "category": {
        "approvedAttributeId": "PLMApprovedItemQuantity",
        "scheduleCategoryRe": "PLMScheduleCategoryRe"
      },
      "metrics": [
        {
          "title": "Item Quantity",
          "targetAttributeId": "PLMTargetItemQuantity",
          "plannedTotalAttributeId": "PLMPlannedItemQuantityTotal",
          "differenceAttributeId": "PLMTargetVersusPlannedItemQuantity"
        }
      ]
    }
  }
}
```

← Christmas 2018 Line Plan Season: Holiday
Line Plan Status: In Progress

Edit Line Plan 4079000 / 100000
Item Quantity

^ Men 1107

- PLMTargetVersusPlannedItemQuantity - when hovered over, a value of 3979000

```
"lineplans": {
  "lineplan": {
    "headerAttributesGrp": {
      "category": {
        "approvedAttributeId": "PLMApprovedItemQuantity",
        "scheduleCategoryRe": "PLMScheduleCategoryRe"
      },
      "metrics": [
        {
          "title": "Item Quantity",
          "targetAttributeId": "PLMTargetItemQuantity",
          "plannedTotalAttributeId": "PLMPlannedItemQuantityTotal",
          "differenceAttributeId": "PLMTargetVersusPlannedItemQuantity"
        }
      ]
    }
  }
}
```

← Christmas 2018 Line Plan Season: Holiday
Line Plan Status: In Progress

Edit Line Plan 4079000 / 100000
Item Quantity

^ Men 1107

In addition to calculated attributes being used in line plan column headers, they are also used in the design specification documents. See below for more on calculated attributes used in design specifications.

Attributes for Design Specifications

Note: Attributes created for design specifications can be calculated attributes, regular attributes, or updated through business rules based on your business needs. For the examples that follow, calculated attributes and regular attributes are used.

Design specifications are created by filling in various attributes and calculated attributes that are valid for the design specification object type.

	Title	Planned Item Units	Planned Cost, Style	Planned Item Revenue, Each	Planned Revenue, Style	Planned Item Gross Margin %
☐	Style 115952	105000 <i>fx</i>	3045000 <i>fx</i>	46.00	4830000 <i>fx</i>	36.96 <i>fx</i>

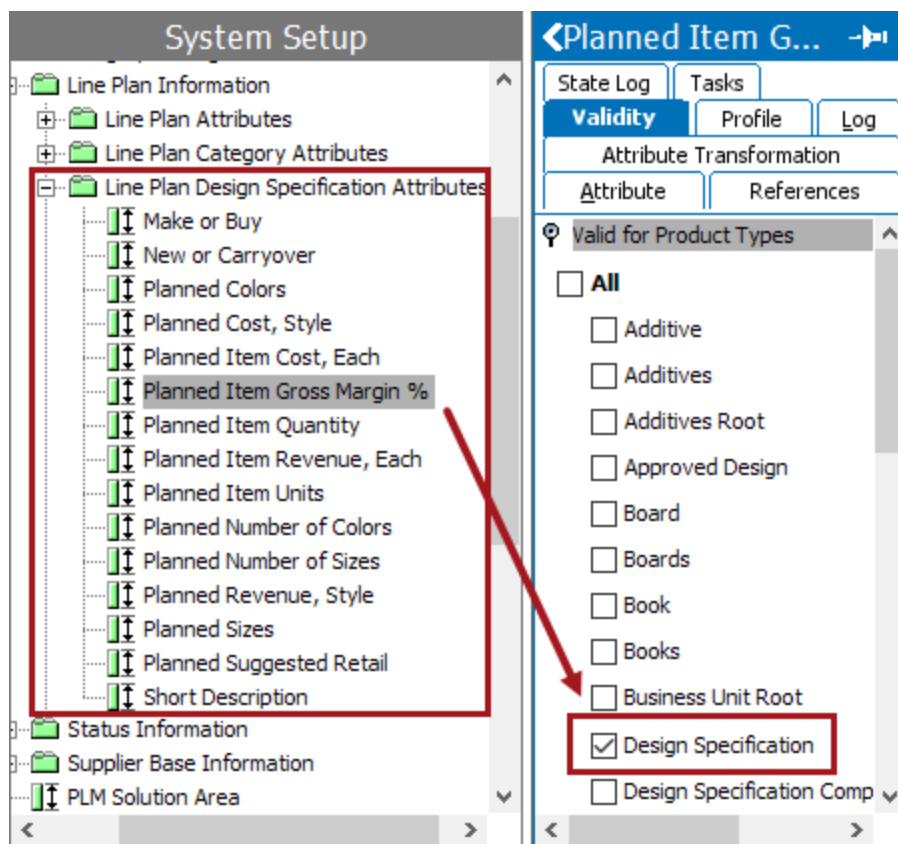
The values input into the design specification attributes and calculated attributes then generate information in the line plan column headers (see section above for more information).

To create design specification attributes, follow the directions below:

1. In workbench, navigate to System Setup and create an attribute group to hold design specification attributes.
2. Create any attributes or calculated attributes needed for design specifications.

Note: Some design specification attributes and calculated attributes may be uploaded via a PLM data package, and may need to be tweaked for company needs. If additional attributes need to be created, talk to your implementation team.

3. Make all of these attributes and calculated attributes valid for the design specification object type. In this example, it is called 'Design Specification.'



Once setup is finished in workbench for design specifications, setup in Web UI can take place. See the **Adding a Classification Screen** topic in this documentation.

For more on how to create attributes, calculated attributes, and set validity, see the **Attributes** topic in the **System Setup / Super User Guide** documentation.

Line Plan Setup in Web UI

Users interact with line plans only in the Web UI, thus it is important to have proper configuration so that the user has a seamless experience. There are a number of configurations and screens that need to be established for users to create, interact with, maintain, and delete line plans. They are the following:

- Adding Line Plans to a Links Widget
- Adding a Children Of Types Screen
- Adding a PLM Line Plan Screen
- Adding a Classification Screen

Adding Line Plans to a Links Widget

It is possible to add a link to a Links Widget for easy user access to line plans. The steps below outline how to add the link. For more information on Links Widgets, see the **Homepage Widgets** topic in the **Web User Interfaces / Web UI Setup and User Guide** documentation.

1. From the Web UI Designer, navigate to --[HOMEPAGE]-- > Widget Grid > Child Components. Under Child Components, add a Links Widget.

Properties

Configuration Web UI style

---[HOMEPAGE]---

Save Close New... Delete Rename Save

Widget Grid Properties

[go to parent](#)

Component Description A component that can contain Web UI widgets in a grid layout

StyleName

Css Class

Child Components

Widgets

Text Widget (Welcome)

Links Widget (Shortcuts)

Add...

Add Component

Analytics Widget

Asset Import Widget

File Loading Widget

Html Asset Widget

Impersonate User Widget

KPI Widget

Links Widget

Mass Creation Widget

Filter

☐ Show deprecated components

✓ Add ✗ Cancel

A widget that can contain a list of components and arranges them vertical when displayed.

2. On the Links Widget Properties under Child Components, add a Node Navigation component.

The screenshot shows the 'Links Widget Properties' dialog box in the StiboSystems interface. The 'Configuration' tab is active. At the top, there are tabs for 'Configuration' and 'Web UI style', and a dropdown menu showing '--[HOMEPAGE]--'. Below these are buttons for 'Save', 'Close', 'New...', 'Delete', 'Rename', and 'Save'. The main title is 'Links Widget Properties' with a 'go to parent' link. A 'Component Description' field contains the text: 'A widget that can contain a list of components and arranges them vertical when displayed.' Below this is a 'StyleName' field. A 'Title' field contains the text 'Shortcuts'. An 'Advanced' section is collapsed. Below the 'Advanced' section is a 'Child Components' section with a table. The table has a header 'Rows' and a column 'Node Navigation'. A red box highlights the 'Add..' button in the table. An 'Add Component' sub-dialog is open, showing a list of components: 'External Navigation', 'GDSN Create RFCIN Link Action', 'Navigation', 'Node Navigation' (highlighted), 'Static Html', and 'Static Text'. To the right of the list is a text field 'Add link to a node hierachy'. Below the list is a 'Filter' field and a checkbox 'Show deprecated components'. At the bottom are 'Add' and 'Cancel' buttons.

3. Double click on the newly added Node Navigation component to open the Node Navigation Properties. Enter in the desired line plan label, and select the node that all line plans will be stored under. In this case, PLMLinePlansRoot is selected to be the parent node when line plans are created.

Properties

Configuration
Web UI style

--[HOMEPAGE]--
Save
Close
New...
Delete
Rename
Save as...

Node Navigation Properties
[go to parent](#)

Component Description
Add link to a node hierachy

StyleName

Context Help

Css Class

Label
Line Plans

Node*
step://classification?id=PLMLinePlansRoot
...

- When finished configuring the link, click Save and close the designer. The link will appear in the Link Widget on the Web UI homepage.

SHORTCUTS

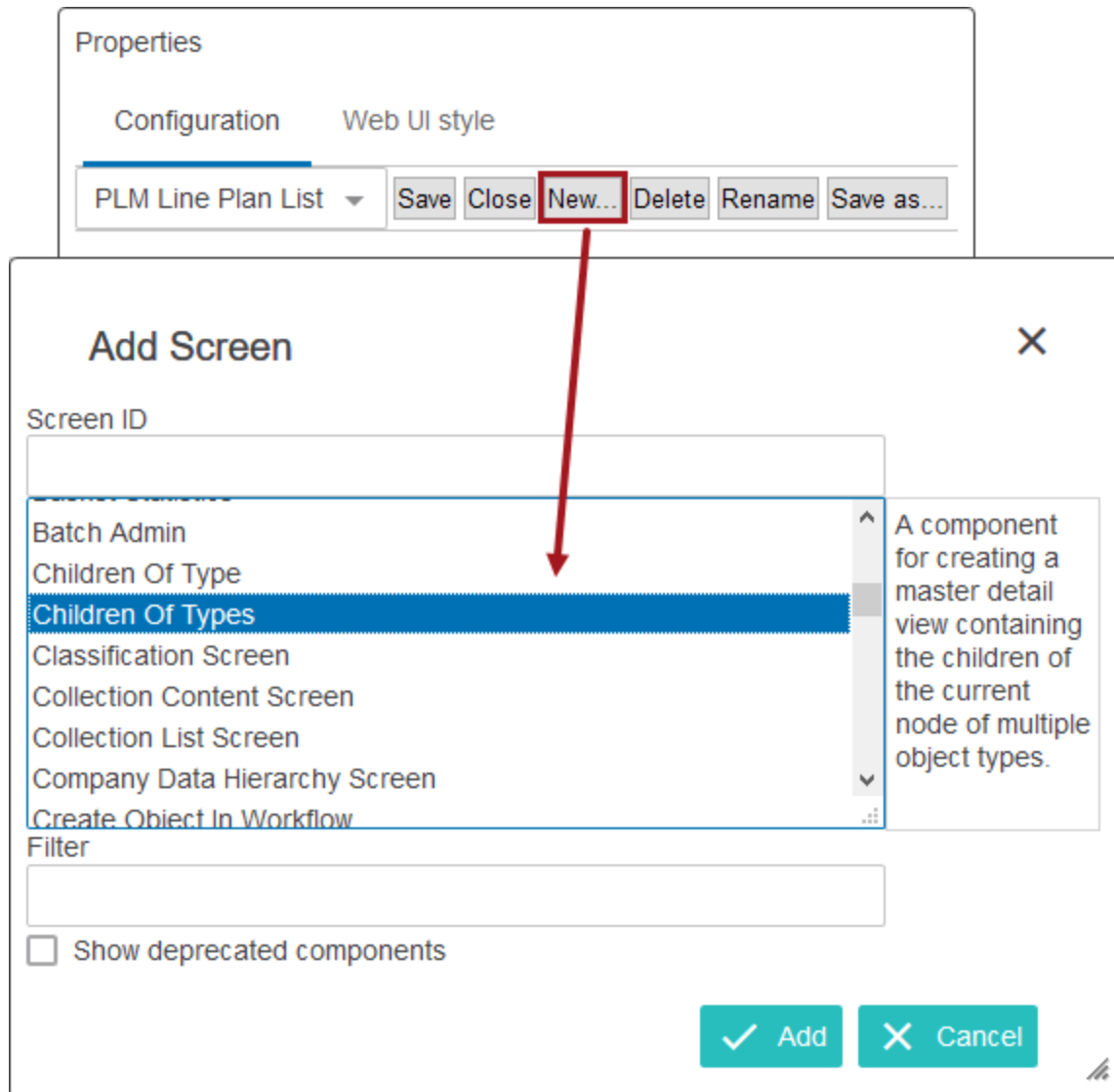
Boards
Schedules
Line Plans

After the line plans link has been created, the Children of Types properties screen, which is the landing page for all line plans in Web UI, needs to be created. See the **Adding a Children Of Types Screen** topic.

Adding a Children Of Types Screen

The Children of Types screen is the landing page for all line plans in Web UI. From this page, users are able to select which line plan they need to work on. To configure this screen:

1. From the designer, select New... > **Children of Types** screen.



2. Enter in a Screen ID, and click **Add**. Specify a screen ID that will be easily identifiable when setting up screen mappings later in this process.

Add Screen [X]

Screen ID
 1

Batch Admin
 Children Of Type
Children Of Types
 Classification Screen
 Collection Content Screen
 Collection List Screen
 Company Data Hierarchy Screen
 Create Object In Workflow

A component for creating a master detail view containing the children of the current node of multiple object types.

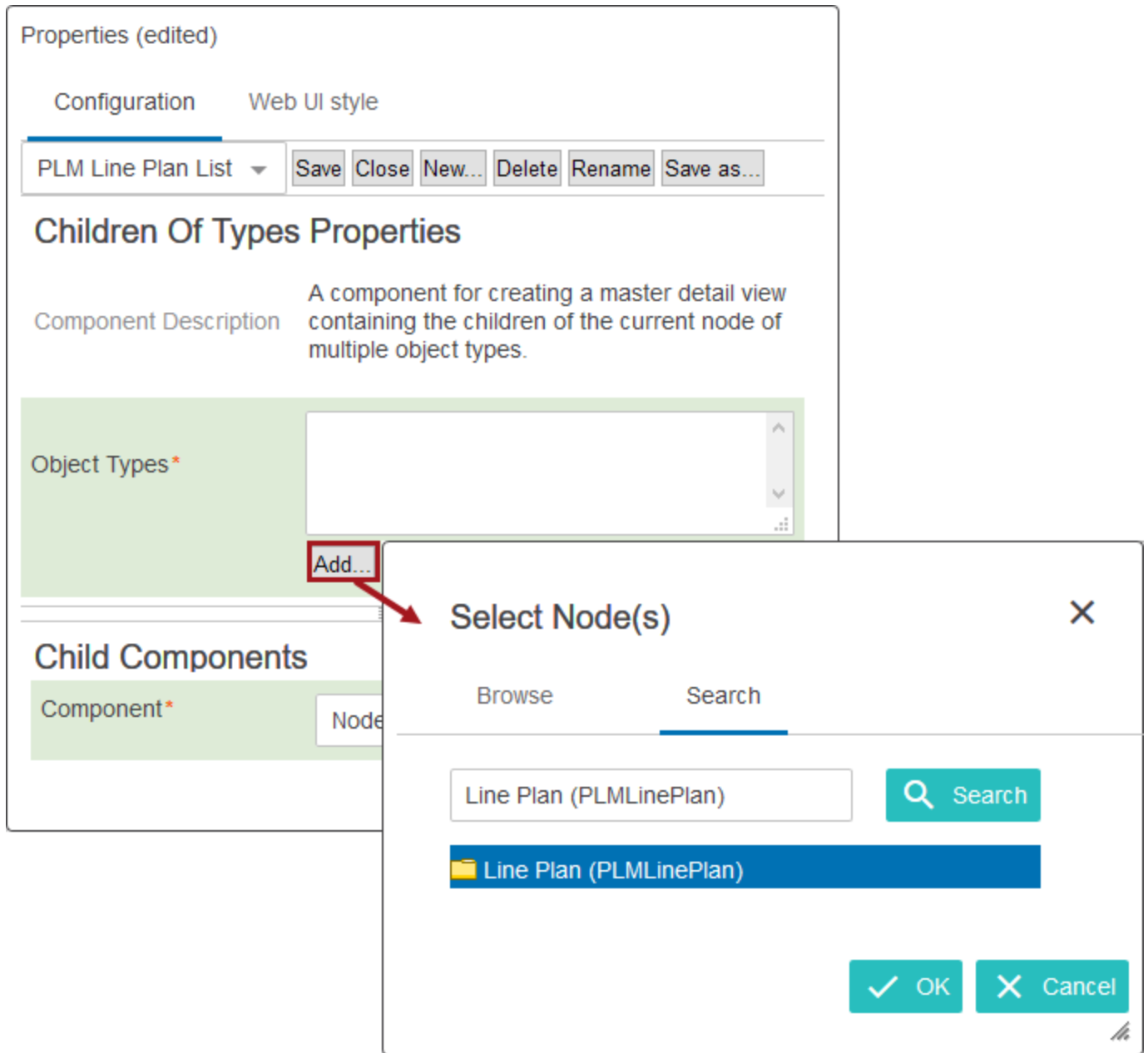
Filter

☐ Show deprecated components

2

- On Children of Types Properties, click on the **Add** button under the **Object Types** field to select the appropriate PLMLinePlan node.

Note: Depending on your configuration in workbench, you may have a different ID than the example. Check with your administrator or implementation team.



4. In the Components field under Child Components, add a Node List.

Properties (edited)

Configuration Web UI style

PLM Line Plan List ▼ Save Close New... Delete Rename Save as...

Children Of Types Properties

Component Description A component for creating a master detail view containing the children of the current node of multiple object types.

Object Types* Line Plan (PLMLinePlan) ▲ ▼

Add... Remove Up Down

Child Components

Component* Node List ▼ [go to component](#)

5. Click on the 'go to component' next to the Node List to open the Node List Properties.
 - For the Display Modes parameter, add a Table Display Mode.
 - In the Actions field, add the Create Classification Action and the Delete From Grid Action.

Note: The Create Classification Action will have the default name of 'Create Classification.' It is considered recommended practice to edit this label so that it is easier for users to understand that clicking this action will create a new line plan.

For more on how to configure action buttons, see the **Action Buttons** topic in the **Web User Interfaces / Web UI Setup and User Guide** documentation.

Properties

Configuration
Web UI style

PLM Line Plan List
Save
Close
New...
Delete
Rename
Save

Node List Properties
[go to parent](#)

StyleName

Child Components

Display Modes
Table Display Mode

Add..
Remove
Up
Down

Actions
Create Classification Action
Delete From Grid Action

Add..
Remove
Up
Down

- Double click on the Table Display Mode to navigate to the Table Display Mode Properties. On Table Display Mode Properties, add any needed Headers, such as Attribute Value Headers or Attribute Value Group Headers, for the line plan. In this example, a Title Header and a number of Attribute Value Headers have been added.

Properties

Configuration
Web UI style

PLM Line Plan List
Save
Close
New...
Delete
Rename
Save as...

Table Display Mode Properties

[go to parent](#)

Component Description
Shows the nodes from a Node List in a table.

StyleName

Context Help

Headers

Title Header (/ true)
Attribute Value Header (false / false / false / PLMLinePlanBusinessCategory / false /
Attribute Value Header (false / false / false / PLMLinePlanYear / false / / false / NotV
Attribute Value Header (false / false / false / PLMLinePlanSeason / false / / false / N
Attribute Value Header (false / false / false / PLMLinePlanStatus / false / / false / No

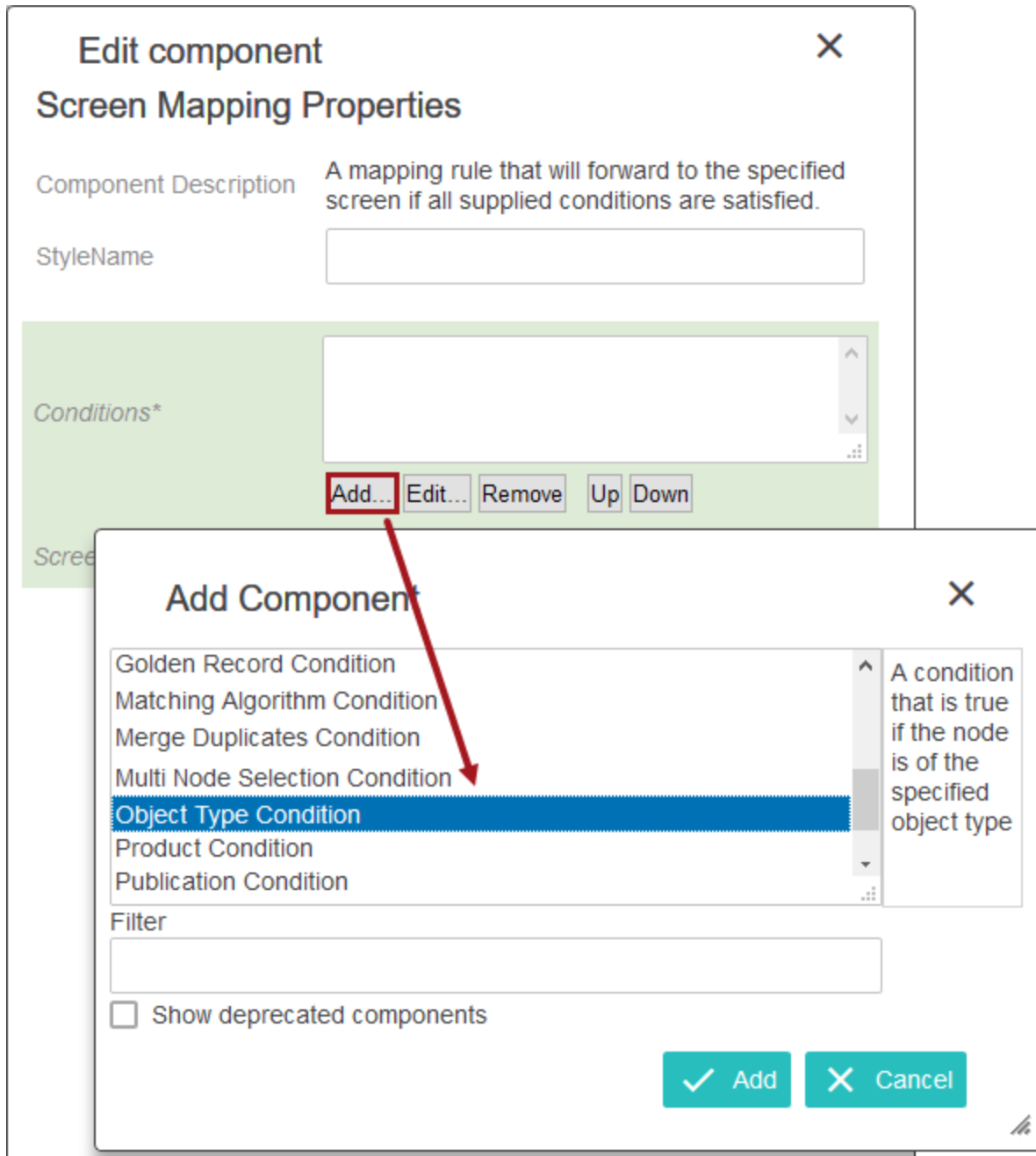
Add...
Edit...
Remove
Up
Down

Show Details
☐

Sizing and filtering

Child Components

7. When all needed headers are added and configured, click Save.
8. To use the screen, it needs to be mapped. Select --[MAIN]-- from the designer screen dropdown. Click Add in the Mappings field to bring up the Screen Mapping Properties.
9. Click Add under the Conditions field, and select the **Object Type Condition** component.



10. On the Object Type Condition Properties, click the ellipsis button (...) to specify the **Object Type**. In this example, it is PLMLinePlansRoot. Click **OK** when finished.

Edit component [X]

Object Type Condition Properties

Component Description: A condition that is true if the node is of the specified object type

StyleName:

Object Type*: PLMLinePlansRoot [Dropdown Arrow]

Select Node(s) [X]

Browse Search

[Search]

Line Plans Root (PLMLinePlansRoot)

[OK] [Cancel]

11. In the Screen dropdown, choose the Child of Types screen that you created in step 2 as the landing page to display the line plans. Click **Save**.

×

Edit component

Screen Mapping Properties

Component Description

A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

StyleName

Conditions*

ObjectType = PLMLinePlansRoot

Add...

Edit...

Remove

Up

Down

Screen*

PLM Line Plan List

homepage

login

Name and ID

nodeDetails

PLM Design Specs

PLM Line Plan List

PLM Line Plan Placeholder

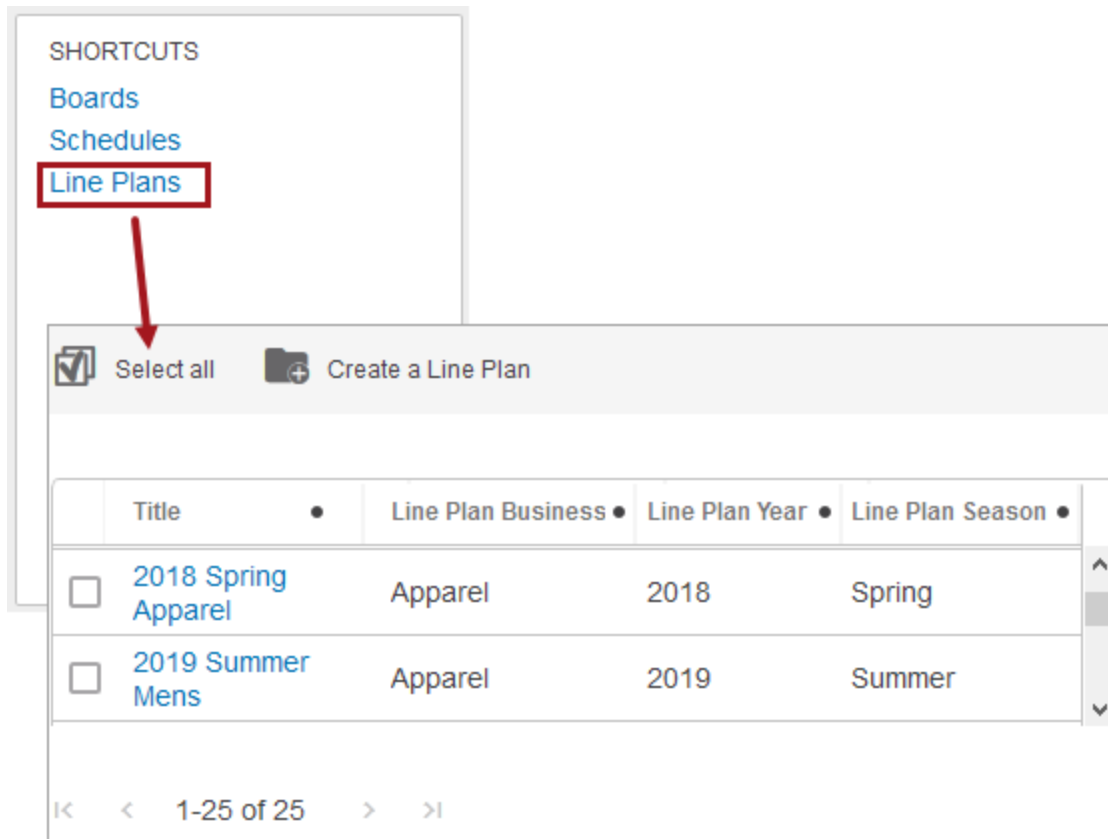
Add

✓ Save

✗ Cancel

- Click **Save** in the designer window, and **Close** when finished.

If the newly created screen is mapped correctly, clicking the Line Plan quick link will direct the user to the Children of Types screen. For more on mapping screens, see the **Mappings** topic in **Main Properties Overview** section in the **Web User Interfaces / Web UI Setup and User Guide** documentation.



Once this screen is configured, a PLM Line Plan screen needs to be configured, so that users are able to see and work with specific line plan data. For more on how to add this screen, see **Adding a PLM Line Plan Screen** in this documentation.

Adding a PLM Line Plan Screen

The PLM Line Plan screen is the screen that displays after an individual line plan has been selected.

← 2019 Summer Mens		Line Plan Year:	2019	
		Line Plan Start Date:	2018-02-05	
		Line Plan Status:	In Progress	
		Schedule Deadline:	2018-11-15	
Edit Line Plan		926037200	3222982000 / 1500000000	65.01 / 46
		Cost	Revenue	Gross Margin % Actions
^	Work From Home Collection	411384000	937278000	51.15 — + ✓
	Bottoms (72 styles)	158409000	349758000	47.01 —
	Shirts (32 styles)	166320000	384720000	51.91 —
	Sweaters (17 styles)	58695000	138000000	56.8 —
	Tees (23 styles)	27960000	64800000	58.93 —

From this detailed overview of the line plan, the user can add, edit, delete, and maintain line plans according to their user privileges. To add this screen:

1. Open the designer, click on **New** and add a **PLM Line Plan** screen. *Specify a screen ID that will be easily identifiable when setting up screen mappings later in this process.*

Add Screen

Screen ID

Multi Selection Screen

Multi Workspace Screen

Node Details

Node List Browser

PLM Line Plan

PLM Schedule Template

Packaging

Planned Spread Screen

^

A screen for maintaining line plans

v

Filter

☐ Show deprecated components

✓ Add

✗ Cancel

- In the PLM Line Plan Properties, click the ellipsis button (...) next to each field to fill in both the **Category Object Type** and **Sub Category Object type** parameters. Add the first and second levels of the line plan to these fields. In this example, it is PLMLinePlanCategoryLevel1 and PLMLinePlanCategoryLevel2.

Properties

Configuration

Web UI style

Line Plan Screen

Save

Close

New...

Delete

Rename

Save as...

PLM Line Plan Properties

Component Description

A screen for maintaining line plans

StyleName

Category Object Type

PLMLinePlanCategoryLevel1

...

Clear

Sub Category Object Type

PLMLinePlanCategoryLevel2

...

Clear

3. Click Save, and using the screen dropdown, navigate to --[MAIN]--.
4. In the Mappings field on --[MAIN]--, map the newly created screen by first clicking Add...
5. Click Add under the Conditions parameter, and then select an **Object Type Condition** component.

Edit component [X]

Screen Mapping Properties

Component Description: A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

StyleName:

Conditions*

[Add...] [Edit...] [Remove] [Up] [Down]

Screen

Add Component [X]

Golden Record Condition
Matching Algorithm Condition
Merge Duplicates Condition
Multi Node Selection Condition
Object Type Condition
Product Condition
Publication Condition

Filter:

☐ Show deprecated components

[✓] Add [X] Cancel

A condition that is true if the node is of the specified object type

6. In the Object Type Conditions Properties, click the ellipsis button (...) to specify the **Object Type**. In this example it is PLMLLinePlan. Click **OK** when finished.

Edit component

×

Object Type Condition Properties

Component Description

A condition that is true if the node is of the specified object type

StyleName

Object Type*

PLMLinePlan

...

Select Node(s)

×

Browse

Search

Line Plan (PLMLinePlan)

Q Search

Line Plan (PLMLinePlan)

✓ OK

✕ Cancel

- In the screen field, choose the PLM Line Plan screen that you created in step 1 as the landing page to display the line plan details. In this example, it is called Line Plan Screen. Click **Save**.

×

Edit component

Screen Mapping Properties

Component Description

A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

StyleName

Conditions*

ObjectType = PLMLinePlan

Add...

Edit...

Remove

Up

Down

Screen*

Line Plan Screen

homepage

assetdetails

Line Plan Screen

Name and ID

PLM Line Plan Placeholder

Add

✓ Save

✗ Cancel

8. Click **Save** in the designer window and then **Close** when finished.

If the newly created screen is mapped correctly, selecting one of the line plans will bring you to the PLM Line Plan screen. For more on mapping screens, see the **Mappings** topic in **Main Properties Overview** section in the **Web User Interfaces / Web UI Setup and User Guide** documentation.

	Select all		Create a Line Plan	
	Title •	Line Plan Business •	Line Plan Year •	Line Plan Status •
☐	2018 Spring Apparel	Apparel	2018	In Progress
☐	2019 Summer Mens	Apparel	2019	In Progress
☐	Christmas 2018	Apparel	2018	In Progress

1-25 of 25

← Christmas 2018		Line Plan Business Cat...	Apparel	
		Line Plan Year:	2018	
		Line Plan Season:	Holiday	
		Line Plan Start Date:	2018-03-06	
		Schedule Deadline:	2018-04-06	
		Line Plan Status:	In Progress	
Edit Line Plan		707 / 240000	623 / 120	
		Item Quantity	Item Unit	Actions
^	Men	707	623	— + ✓
	Jackets (9 styles)	202	22	—
	Jumpers and Cardigans (2 styles)	16	210	—
	Pants (2 styles)	42	19	—
	Shirts (1 styles)	56	23	—
	T-Shirts (7 styles)	327	285	—
	Underwear (1 styles)	64	64	—

The user will now be able to add, delete, maintain, and modify the selected line plan's data according to user group privileges. For more information on user group privileges, see the **Context, Action Sets, and User Groups** topic in the **PLM for Admins** documentation.

Adding a Classification Screen

After a line plan has been selected and a specific line plan item in that line plan has been selected, a user is brought to a Classification Screen, which displays further details about the particular item. On this page, a user is able to add a design specification and fill out or change any data needed for the selected item in the table. Follow the steps below to create and configure the Classification screen.

Classification Screen

☒ Select all
 ← Go Back
 Create Design Specification

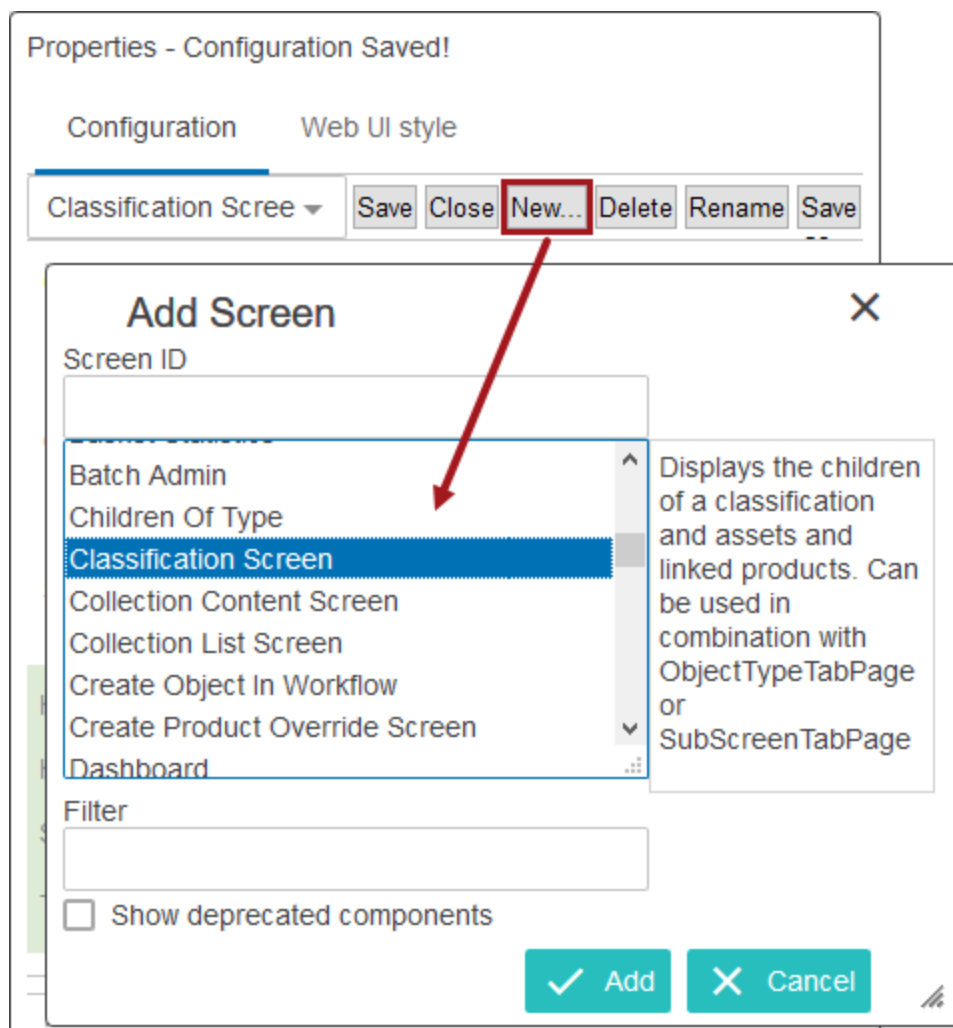
	Title •	Short Description •	Planned Colors •	Planned Number of Colors •	Planned Sizes •
☐	Style 116418	Accessory 1	Blue Black Red	3 <i>fx</i>	One-size
☐	Style 116421	Accessory 2	Blue Black Red	3 <i>fx</i>	One-size
☐	Style 116422	Accessory 3	Blue Black Red	3 <i>fx</i>	One-size
☐	Style 116423	Accessory 4	Blue Black	3 <i>fx</i>	One-size

←
→

↶
↷

Number of items : 16

1. Open the designer, click on **New...** and add a **Classification Screen**. *Specify a screen ID that will be easily identifiable when setting up screen mappings later in this process.*



2. In the Classification Screen Properties, under Child Components, select **Node List** from the Node List dropdown menu and click on **go to component**.

Properties - Configuration Saved!

Configuration Web UI style

Classification Scree ▼ Save Close New... Delete Rename Save

Classification Screen Properties

StyleName

Hide Assets ☐

Hide Linked Products ☐

Show Title ☒

Title

Child Components

Node List Node List ▼ [go to component](#)

3. On the Node List Properties, confirm that the following parameters are enabled:
 - Include Labels
 - Lookup Screen Type For Navigation
 - Enable Default Sorting

Properties

Configuration Web UI style

PLM Design Specs ▾ Save Close New... Delete Rename Save as...

Node List Properties [go to parent](#)

StyleName

Hide Standard Buttons ☐

ID*

Include Labels ☒

Lookup Screen Type For Navigation ☒

Page Size

Use Details Overlay ☐

Default Sorting Order

Enable Default Sorting ☒

4. Next, under Child Components, go to Display Modes and add a **Multi Edit Display Mode**. Additionally, under Actions, add:
 - Go Back To Line Plan Toolbar Action
 - Create And Link Design Specification Action
 - Delete From Grid Action

Properties

Configuration Web UI style

PLM Design Specs ▾ Save Close New... Delete Rename Save as...

Node List Properties [go to parent](#) ^

Child Components

Display Modes

Multi Edit Display Mode

Add.. Remove Up Down

Actions

Go Back To Line Plan Toolbar Action
Create And Link Design Specification Action
Delete From Grid Action

Add.. Remove Up Down

Create And Link Design Specification Action

1. Configure the Create And Link Design Specification Action Properties:
 - Enter a button label.
 - Choose the object type that represents the design specifications. In this example it is PLMDesignSpecification.
 - Choose the object type that is the parent to the design specifications. In this example it is PLMDesignSpecificationsRoot.
 - Configure the context help field that displays when a user hovers the mouse over the action button.

Add component - configure required properties

Required properties (*) must be set before the component can be added to the configuration.

Create And Link Design Specification Action Properties

Component Description Create a new Design Specification and link it into the current classification

Button Label	Create Design Spec
Object Type*	PLMDesignSpecification
Parent*	PLMDesignSpecificationsRoot
Context Help	Click this to create a new design specification

- Back on Node List Properties, double click on 'Multi Edit Display Mode' to configure the fields that will display in the table.
- In the Multi Edit Display Mode, click Add under the Headers field and repeat as necessary to add and configure as many headers as needed. In the example below, a Title Header and multiple Attribute Value Headers have been added.

Note: As part of the PLM configuration package, there are some attributes that need to be created and configured in workbench in order for design specifications to work properly. They then need to be added as attribute value headers (or as part of an attribute value group header) in order for design specifications to work in Web UI. See the **Setting Up Attributes for Line Plans** topic in this documentation for a list of attributes that need to be added for design specifications to work.

Properties

Configuration

Web UI style

PLM Design Specs ▾

Save

Close

New...

Delete

Rename

Save as...

Multi Edit Display Mode Properties [go to parent](#)

Component Description

Displays the objects list in a Node List presented in an editable table.

Context Help

i18n.stibo.portal.server.components.ma:

Title Header (true)

Attribute Value Group Header (false / fa

Headers

Add...

Edit...

Remove

Up

Down

Add Component

Approved Header

Attribute Value Group Header

Attribute Value Header

Classification-Specific Attribute Value Group Header

Classification-Specific Attribute Value Header

Data Container Attribute Value Group Header

Data Container Attribute Value Header

Deduplication Header

Select a component to see its description

Filter

☐ Show deprecated components

✓ Add

✗ Cancel

Delete From Grid Action

When configuring the Delete From Grid Action, if the 'Confirm Selection' checkbox is enabled, a confirmation dialog with selection details will appear when a user selects items from the table to be deleted.

The image shows two overlapping windows from the StiboSystems interface. The top window is titled 'Delete From Grid Action Properties' and is part of the 'PLM Line Plan List' configuration. It has tabs for 'Configuration' and 'Web UI style'. The 'Configuration' tab is active, showing various settings. A red rectangle highlights the 'Confirm Selection' checkbox, which is checked. A red arrow points from this checkbox to a 'Delete a Line Plan' dialog box in the foreground.

The 'Delete From Grid Action Properties' window includes the following fields:

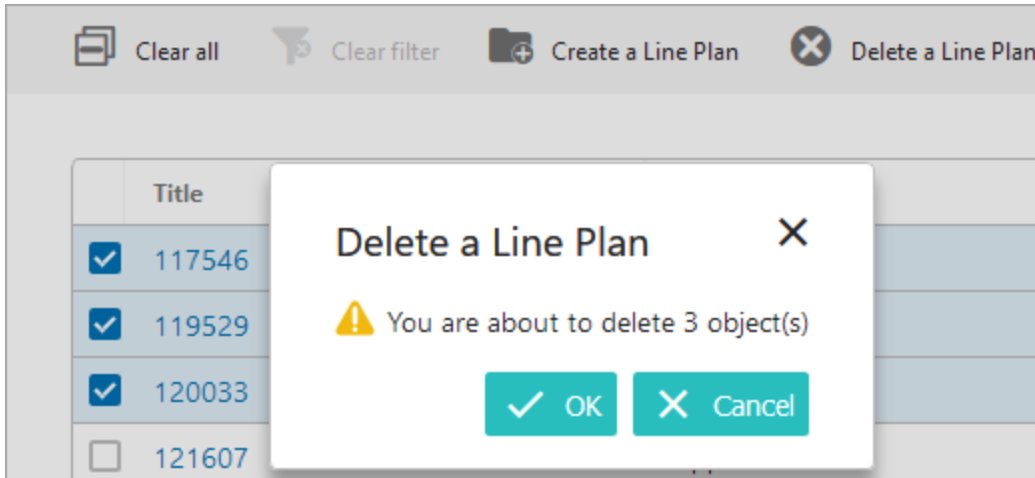
- Approve Deletion:** ☐
- Button Label:** Delete a Line Plan
- Cancel Button Label:** i18n.stibo.portal.server.component:
- Confirm Selection:** ☒ (highlighted with a red box)
- Custom Icon:** [Field with a 'Reset' button]

The foreground window is a 'Delete a Line Plan' dialog box. It contains a warning icon and the text: 'You are about to delete 3 object(s)'. Below this, there are radio buttons for 'All' (selected) and 'None'. A list of three items is shown, each with a checked checkbox:

- ☒ 117546
- ☒ 119529
- ☒ 120033

At the bottom of the dialog are two buttons: 'OK' (with a checkmark icon) and 'Cancel' (with an 'X' icon).

They can choose to delete only some of the initially selected items, All, or None. If the 'Confirm Selection' checkbox is not enabled, the user will just get a dialog asking them to confirm their initial selection.



If the newly created screen is mapped correctly, when a user clicks on a line plan item, they are able to see and interact with the item details according to user privileges. For more on mapping screens, see the **Mappings** topic in **Main Properties Overview** section in the **Web User Interfaces / Web UI Setup and User Guide** documentation.

← Christmas 2018

Line Plan Business Cat...

Apparel

Line Plan Year:

2018

Line Plan Season:

Holiday

Line Plan Start Date:

2018-03-06

Schedule Deadline:

2018-04-06

Line Plan Status:

In Progress

Edit Line Plan

707 / 240000

623 / 120

	Item Quantity	Item Unit	Actions
^ Men Jackets (9 styles) Jumpers and Cardigans (2 styles)	707	623	— + ✓ — —

Classification Screen

☒ Select all

← Go Back

Design spec creation

	Title	Planned Colors	Planned Number of Colors	Planned Sizes	Planned Quantity
☐	Long Quilted Jacket	Blue Black	2 <i>fx</i>	S M	200
☐	Quilted Coat		0 <i>fx</i>		

Number of items : 9

If a user needs to edit a reference on a design specification, a Node Details screen needs to be created, configured, and mapped.

In the Node Details Properties (Child Components > Main), add a Multi Reference component configured with a Node List that uses Multi Edit Display Mode. Configure the Multi Edit Display Mode as needed. For more on how to configure this, see the **Multi-Reference Editor** topic in the **Using a Web UI** section of the **Web User Interfaces / Web UI Setup and User Guide** documentation.

Once configured, this will need to be mapped correctly. For information on mapping screens, see the **Mappings** topic in **Main Properties Overview** section in the **Web User Interfaces / Web UI Setup and User Guide** documentation.

Setting Up Schedules in Workbench

Schedules assist managers in attaining deadlines and goals outlined by the line plan, helping to ensure that a product goes to market on time. For schedules to work correctly in Web UI though, proper setup first needs to happen in the workbench.

Prerequisites

- The PLM configuration file is uploaded to STEP. See **Configuration Setup for Line Plans and Schedules** in this documentation.
- Line plans must be set up on the system. This includes the ability to create design specifications, and the ability to approve line plans that are deemed complete. See the topics **Setting Up Line Plans in Workbench** and **Adding a Classification Screen** in this documentation for more information.
- If using workflows, create any workflows needed for schedules. For more on how to configure a workflow, see the **Workflows** topic in the **Workflows** documentation.

Additional setup needed

Read the following topics on how to set up schedules properly in the workbench:

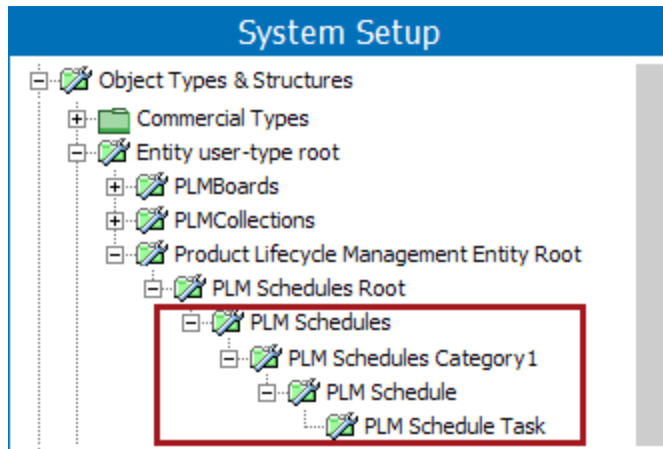
- Creating Schedule Object Types
- References Needed for Schedules
- Setting Up Attributes for Schedules

Creating Schedule Object Types

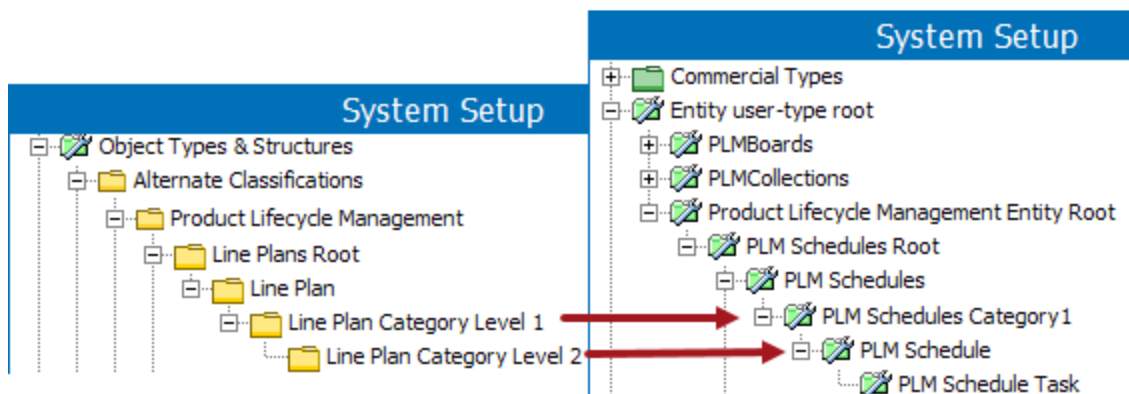
There are a number of entity object types necessary for schedules to function properly in Web UI. The following sections describe what entity object types need to be created.

Creating PLM Schedules

Schedules are entity object types that live in a hierarchy structure like the one shown below.



In the example structure, every time a user approves a Line Plan Category 1 object type, it creates a schedule in the hierarchy structure as a PLM Schedules Category 1 object type. Additionally, for each Line Plan Category 2 object type, a PLM Schedule object type is created.



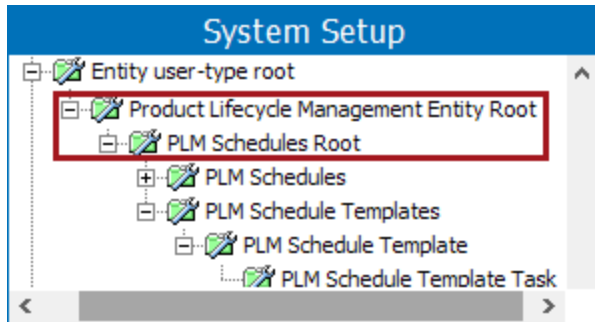
For more on schedule task templates and how to create them, see the next section of this topic.

Note: It is important that the object type IDs used in workbench for schedules are the same used in the uploaded PLM configuration file and in the Web UI designer, otherwise schedules will not work correctly. Talk to your implementation team to ensure they are in alignment.

For more on how to create entity object types in System Setup, see the **Object Maintenance in Tree** topic in the **Getting Started / User Guide** documentation.

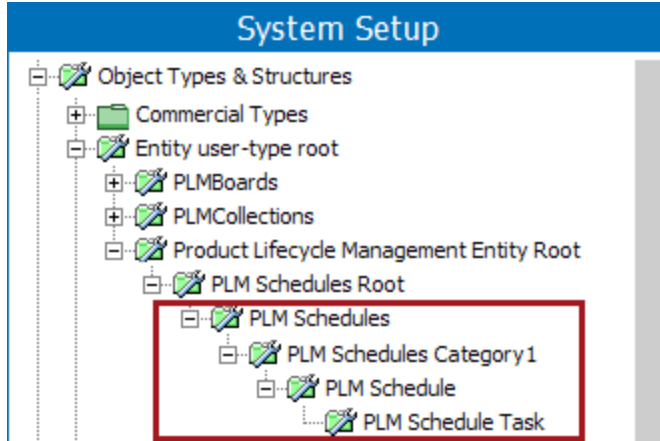
To create the PLM Schedules structure:

1. Create the Product Lifecycle Management Entity Root node and its child PLM Schedules Root if they do not already exist.

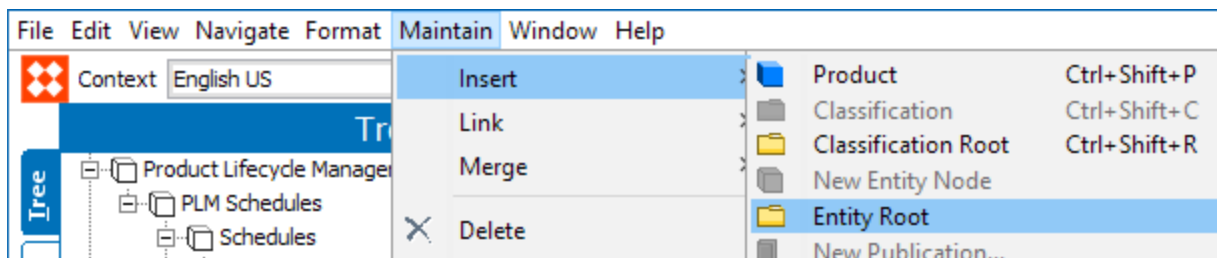


Note: This top-level parent Product Lifecycle Management Entity Root folder and its child PLM Schedules Root will not just hold PLM Schedules, but will hold PLM Schedule Templates as well. For more on schedule templates and how to create them, see the next section of this topic.

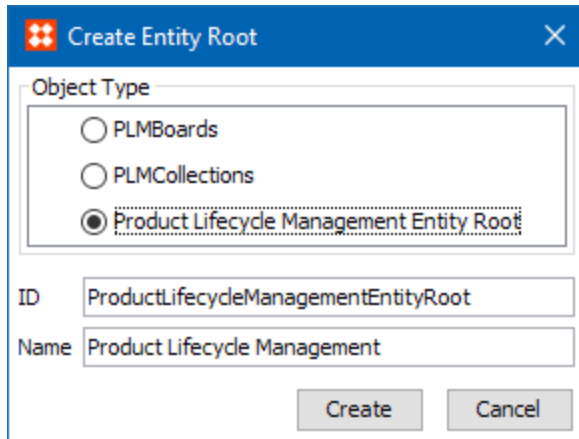
2. Under the Product Lifecycle Management Entity Root node, create the needed hierarchy structure for PLM Schedule object types.



3. To add the schedule structure to Tree, go to Maintain > Insert > click Entity Root.

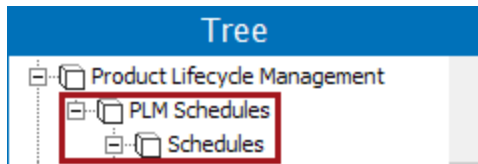


- The 'Create Entity Root' dialog will appear. Select the object type created to hold all entity schedule object types. In this case it is ProductLifecycleManagementEntityRoot.

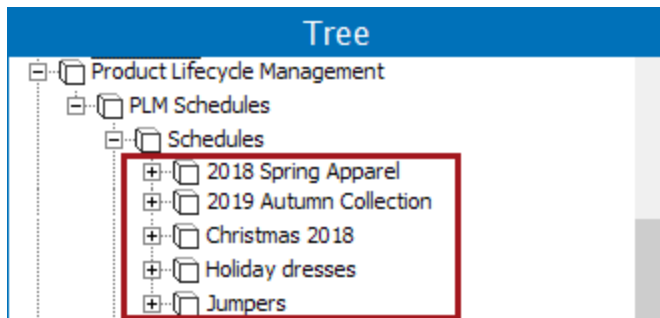


The 'Create Entity Root' dialog box is shown. It has a title bar with a close button. Inside, there is a section labeled 'Object Type' with three radio buttons: 'PLMBoards', 'PLMCollections', and 'Product Lifecycle Management Entity Root'. The third option is selected. Below this, there are two text input fields: 'ID' with the value 'ProductLifecycleManagementEntityRoot' and 'Name' with the value 'Product Lifecycle Management'. At the bottom right are 'Create' and 'Cancel' buttons.

- Once added, add the nodes PLM Schedules and its child Schedules.



All future schedules and their subcategories will be created automatically under the Schedules node as a result of approved line plans. For more on line plan approval, see **Setting up Attributes for Line Plans** in the **PLM for Admins** documentation, or the **Creating Line Plans** topic in **PLM for Users** documentation.



Creating Schedule Template Tasks

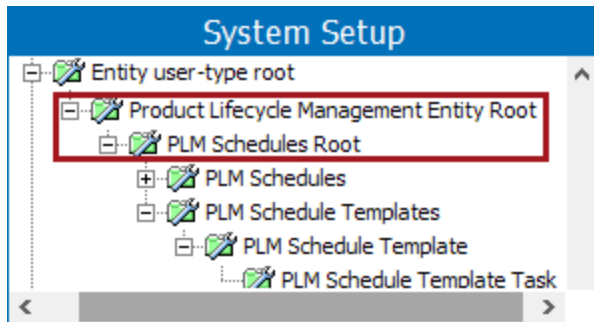
Schedule Template tasks are the templates used when creating schedules. Each template has a set of tasks that may have dependencies on one another, and help define the schedule. In order for schedules to work properly in Web UI, schedule templates need to be created using entity object types. Schedule templates can be created one of two ways:

- Import and mapping upon import
- Building the template in the workbench

For more information on how to import and properly map inbound data, see the **Import Manager** and **Data Mapping** topics in the **Data Exchange** documentation.

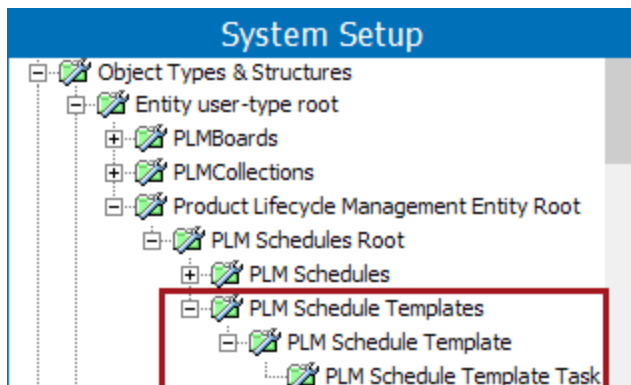
Note: It is important that the object type IDs used in workbench for schedule templates tasks are the same used in the uploaded PLM configuration file and in the Web UI designer, otherwise schedules will not work correctly. Talk to your implementation team to ensure they are in alignment.

1. If the Product Lifecycle Management Entity Root folder and its child PLM Schedules Root do not already exist, create these nodes.

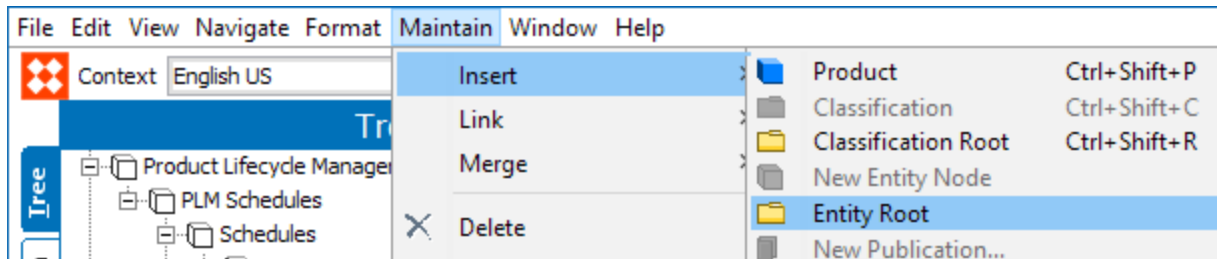


Note: This top level parent Product Lifecycle Management Entity Root folder and its child PLM Schedules Root will not just hold PLM Schedule Template, but will hold PLM Schedule as well. For more on schedules and how to create them, see the above section of this topic

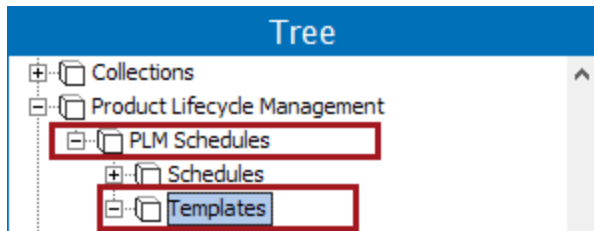
2. Add the needed hierarchy structure for PLM Schedule Templates according to business needs. Below is an example hierarchy structure for PLM Schedule Templates.



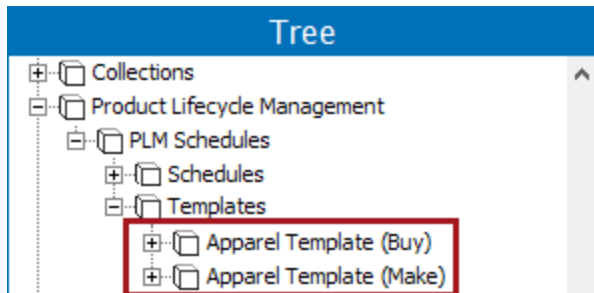
3. To add the schedule templates to Tree, go to Maintain > Insert > click Entity Root.



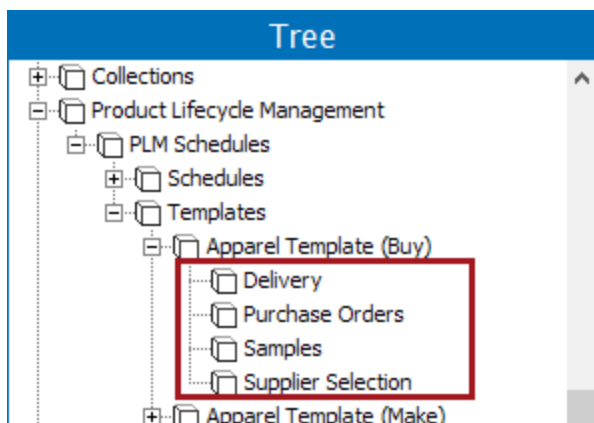
4. The 'Create Entity Root' dialog will appear. Select the object type created to hold all entity schedule object types, if it does not already exist in the system. In this case it is ProductLifecycleManagementEntityRoot.
5. Once added, add the nodes PLM Schedules and its child Templates if it does not already exist on the system.



6. Add as many templates as needed under the Templates node. These templates will become available to select from when creating schedules. See the **Creating Schedules** topic in the **PLM for Users** documentation.



7. Under each template, create the number of needed template tasks.



In Web UI, when a user is creating a schedule and chooses one of the templates, a copy of these schedule template tasks then gets created for each individual line plan.

Note: Schedule Template Tasks use the reference 'PLM Schedule Template Task Dependency,' to ensure that each template task has the appropriate task dependencies. If the reference dependencies are not configured properly, schedules will not work. See the topic **References Needed for Schedules** in this documentation for more information.

References Needed for Schedules

There are three entity references and one product reference needed for schedules to work properly in Web UI. The three entity references are:

- Schedule Template
- PLM Schedule Template Task Dependency
- Schedule Task Dependency

The Product reference needed is:

- Schedule Task

These references enable schedules to work with templates, tasks, and design specifications to allow for a seamless experience with schedules.

Entity Reference Types

Schedule Template

The Schedule Template entity reference is a reference between the schedule and the schedule templates. Each time a template is selected for the creation of a schedule, a copy of the templates is created for the schedule.

The screenshot displays the 'System Setup' interface. On the left, a tree view under 'Reference Types' shows 'Schedule Template' selected and highlighted with a red box. On the right, the 'Schedule Template - Validity' window is open, showing two sections: 'Valid Source Types' and 'Valid Target Types'. Both sections have a table with columns 'ID' and 'Name'.

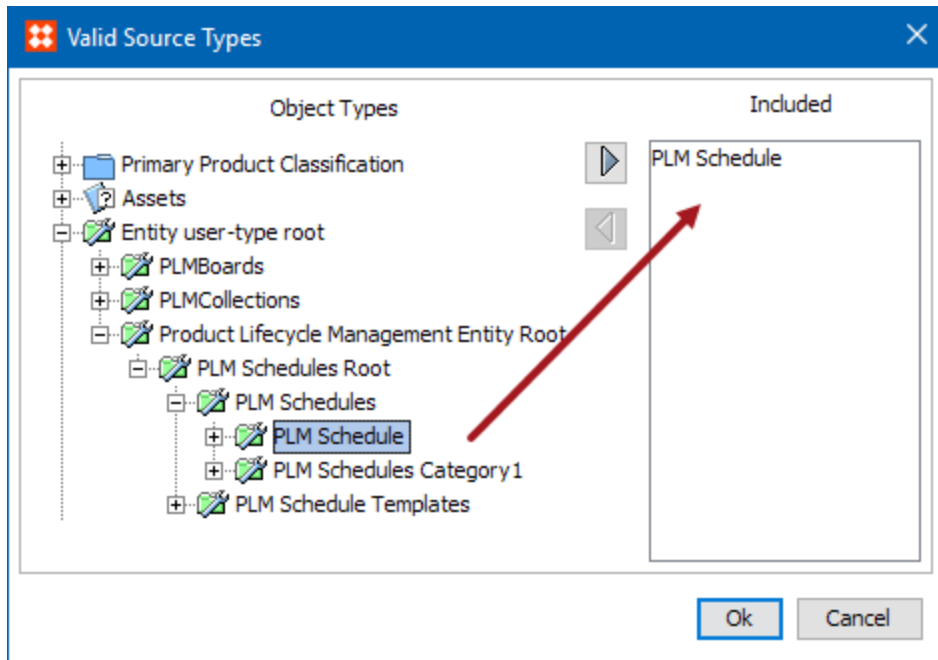
ID	Name
PLMSchedule	PLM Schedule

Below the 'Valid Source Types' table is a link '> Modify Source Types'. The 'Valid Target Types' section has a similar table:

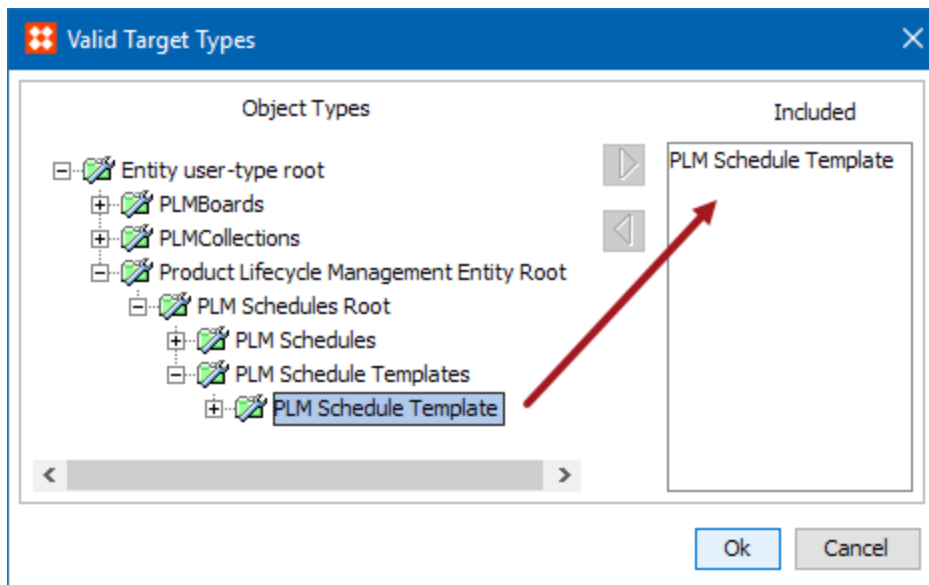
ID	Name
PLMScheduleTemplate	PLM Schedule Template

Below the 'Valid Target Types' table is a link '> Modify Target Types'.

The Valid Source Types is the object type that represents schedules, in this case it is PLM Schedule.



The Valid Target Type is the object type that represents schedule templates, in this case it is called PLM Schedule Template.



Additionally, this reference has two attributes on it that save the selected Scheduling Method and the Scheduling Date. For more on the Scheduling Method and Scheduling Date attributes, see the **Scheduling Method and Scheduling Date** topic in this documentation.

Tree

- Product Lifecycle Management
 - PLM Schedules
 - Schedules
 - Apparel
 - 2018 Spring Apparel
 - 2019
 - Children
 - T-shirts

T-shirts rev.0.2 - References

PLM Schedule | **References** | Referenced By | Status | State Log | Tasks

Schedule References

Reference Type	Target	Scheduling Method	Scheduling Date
Schedule Template	Apparel Template (Buy)	Need By	2018-05-05

PLM Schedule Template Task Dependency

The PLM Schedule Template Task Dependency reference determines the order, or dependencies, of the schedule template tasks for the schedules. For example, when looking at the schedule template 'Apparel Template (Make)' each task created in the template has a defined dependency on the other tasks in the template. Looking at the 'Sample' template task, you can see that once this task is complete, the Design template task will begin.

Tree

- Product Lifecycle Management
 - PLM Schedules
 - Schedules
 - Templates
 - Apparel Template (Buy)
 - Apparel Template (Make)
 - Delivery
 - Design
 - Design Brief
 - Margin Review and Selection
 - Production
 - Purchase Orders
 - Samples

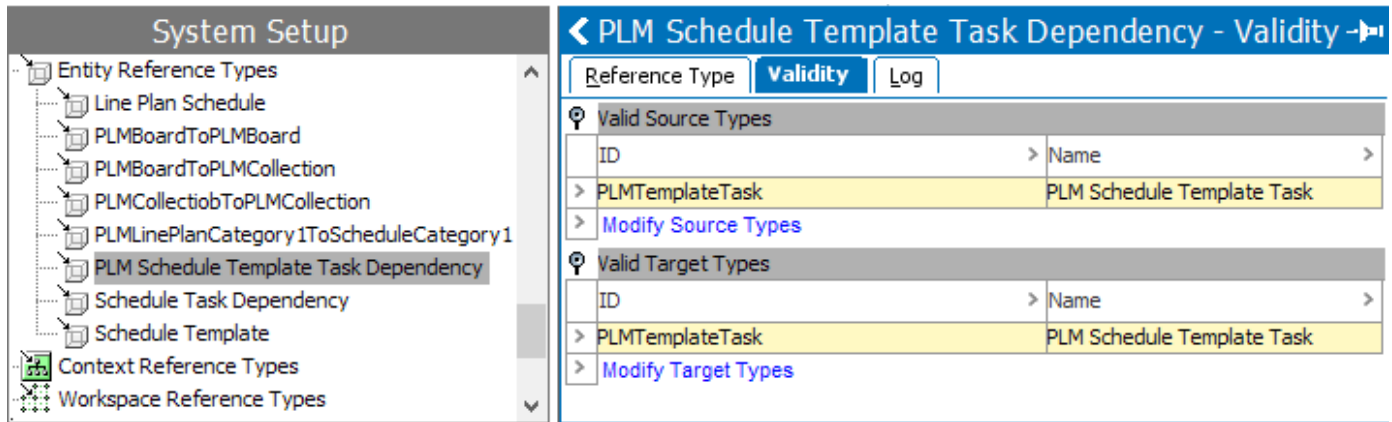
Samples rev.0.2 - References

Status | State Log | Tasks | **References** | Referenced By

Schedule References

Reference Type	Target
PLM Schedule Template Task Dependency	Design

This entity reference is used to establish the order of tasks so that scheduled dates can be calculated, based on dependent tasks and their durations. Notice that the validity for the Valid Source Types and Valid Target Types are both the Entity 'PLM Template Task' object type.

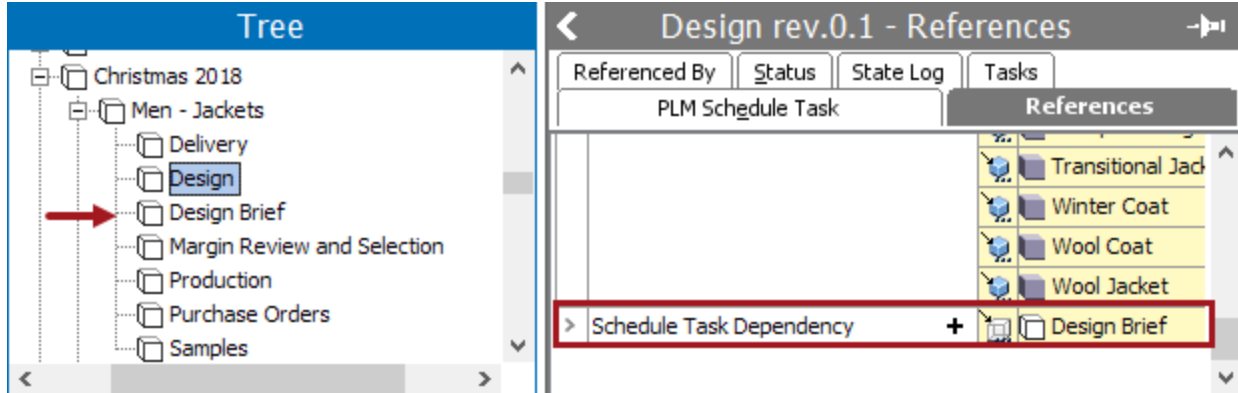


Note: When a user chooses which template to use when creating a schedule, a copy of the template tasks and reference are created for each individual schedule. The reference that mimics the PLM Schedule Template Task Dependency reference for the actual schedule is called Schedule Task Dependency.

Schedule Task Dependency

This reference determines the order, or dependencies, of the tasks in the schedules. In the example below, the name of the reference is Schedule Task Dependency.

In this example, when looking at the task assigned to the manufacturing of a men's jacket, we can see that after the 'Design' phase takes place, the next task is to create 'Design Brief.'



This entity reference is used to establish the order of tasks so that scheduled dates can be calculated, based on dependent tasks and their durations. Notice that the validity for the Valid Source Types and Valid Target Types are both the 'PLM Schedule Task' entity object type.

System Setup

- Entity Reference Types
 - Line Plan Schedule
 - PLMBoardToPLMBoard
 - PLMBoardToPLMCollection
 - PLMCollectionToPLMCollection
 - PLMLinePlanCategory1ToScheduleCategory1
 - PLM Schedule Template Task Dependency
 - Schedule Task Dependency**
 - Schedule Template
- Context Reference Types
- Workspace Reference Types

Schedule Task Dependency - Validity

Reference Type
Validity
Log

Valid Source Types

ID	Name
PLMScheduleTask	PLM Schedule Task

[Modify Source Types](#)

Valid Target Types

ID	Name
PLMScheduleTask	PLM Schedule Task

[Modify Target Types](#)

Product Reference Type

Schedule Task

This reference enables schedules and design specifications to work together seamlessly. In the example below, the name of the reference is 'Schedule Task.'

System Setup

- Reference Types
 - Product Reference Types
 - Idea Board
 - Line Plan Item
 - Package Label
 - Packaging Material
 - Parameter
 - PBoardToCollection
 - PLMBoardToProductContent
 - ProductContentHeroImage
 - Related Specification
 - Requirement
 - Schedule Task**
 - Image and Document Reference Types
 - Classification Reference Types
 - Product to Classification Link Types
 - Product Attribute Link Type
 - Classification Attribute Link Type
 - Entity Reference Types
 - Context Reference Types
 - Workspace Reference Types
 - Workspaces

Schedule Task - Reference Type

Reference Type
Validity
Log

Description

Name	Value
ID	PLMScheduleTask
Name	Schedule Task
Last edited by	2018-01-18 14:23:47.0 by STEPSYS
Externally Maintained	No
Allow multiple references	Yes
Mandatory	No
Inheritance	None

Aspects

In Attribute Groups

ID	Name
PLMScheduleActualsAttributeGr...	Schedule Actuals Attributes
PLMScheduleReferences	Schedule References
Add Attribute Group	

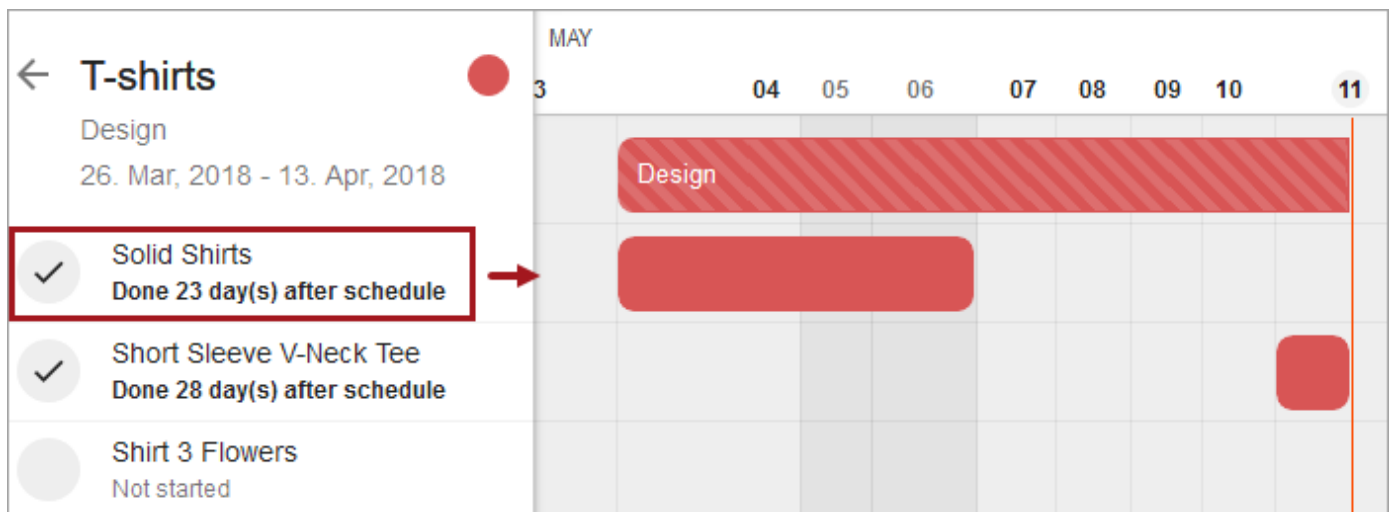
Valid Attributes

ID	Name
PLMProjectStartDateActual	Actual Start
PLMProjectEndDateActual	Actual End
PLMScheduleTaskDurationActual	Actual Duration
Add Attribute	

When a design specification is finished with a task on a schedule, through business rules, the actual start and end dates of that task can be recorded on the Schedule Task reference via the 'PLMProjectStartDateActual' and 'PLMProjectEndDateActual' attributes.

Tree		rift rev.0.1 - Referenced By			
Product Lifecycle Management		Images & Documents	Commercial	Tables	Status
Design Specifications		Product	Sub Products	References	State Log
Red Flower T-shirt		Tasks			
Yellow Dress		Referenced By			
Yellow Pants		Line Plan References			
Yellow T-shirt		Schedule Actuals Attributes			
Pants		Reference Type	Source	Actual Start	Actual End
Party Pants			Delivery		
Pink Rose Pants			Design	2018-05-04	2018-05-06
Pink Pants			Design Brief		
Red Pants			Margin Review and Selection	2018-05-07	
Santa Pajamas			Production		
Shirt 3 Flowers			Purchase Orders	2018-05-07	
Solid Shirts			Samples	2018-05-07	2018-05-07

This then generates in the Web UI to show a complete picture of where a design specification is in relation to the schedule.



For more on attributes needed for schedules to work, see the **Setting Up Attributes for Schedules** topic in this documentation.

When creating the Schedule Task product reference type, make the Valid Source Types valid for the 'PLM Schedule Task' entity type. Make the Valid Target Type valid for the design specification object type.

System Setup

- Reference Types
 - Product Reference Types
 - Idea Board
 - Line Plan Item
 - Package Label
 - Packaging Material
 - Parameter
 - PBoardToCollection
 - PLMBoardToProductContent
 - ProductContentHeroImage
 - Related Specification
 - Requirement
 - Schedule Task**

Schedule Task - Validity

Reference Type
Validity
Log

Valid Source Types

ID	Name
PLMScheduleTask	PLM Schedule Task
Modify Source Types	

Valid Target Types

ID	Name
PLMDesignSpecification	Design Specification
Modify Target Types	

Setting Up Attributes for Schedules

When using schedules, there are numerous attributes that need to be configured. These attributes help schedules:

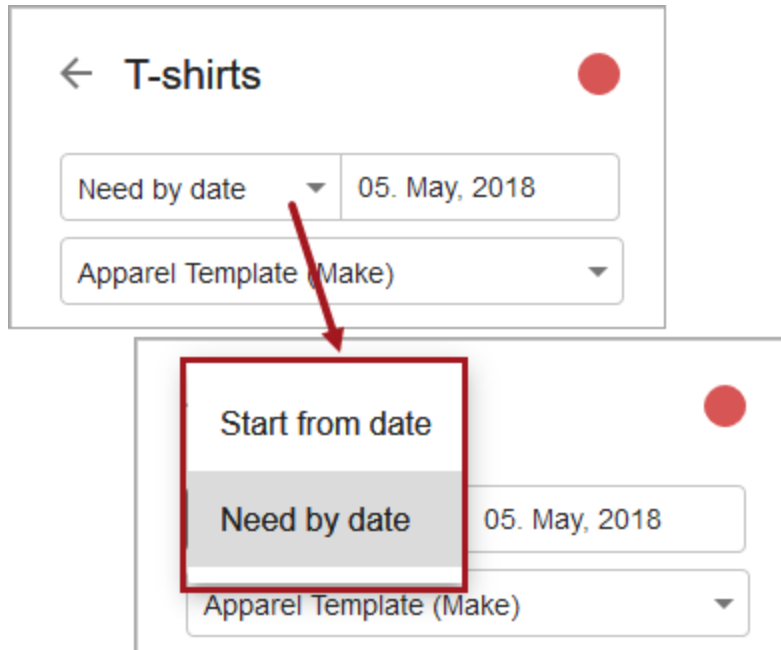
- Identify if the schedule being created is a 'Start from date' vs. 'Need by date'
- The planned duration a task should take
- After the schedule is submitted, what are the estimated planned start and planned end dates for each task
- The actual start and actual end dates a design specification takes to complete a task in a schedule
- Which workflow should be initiated for tasks in a schedule, if using workflows

The sections below detail what attribute setups are need. For more on how to create attributes, see the **Attributes** topic in the **System Setup / Super User** documentation.

- Scheduling Method and Scheduling Date
- Planned Start and Planned End
- Planned Duration
- Schedule Is Submitted
- Actual Start and Actual End
- Schedule Deadline
- Workflow and State ID

Scheduling Method and Scheduling Date

When creating a schedule in Web UI, a user is asked to select the Scheduling Method, indicating if the schedule being created is either a 'Start from date' or 'Need by date.'



← T-shirts

Need by date ▼ 05. May, 2018

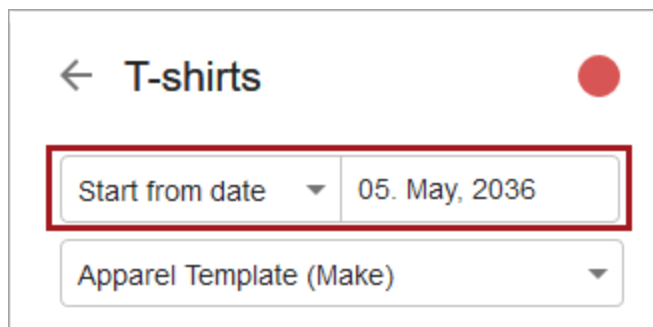
Apparel Template (Make) ▼

Start from date

Need by date 05. May, 2018

Apparel Template (Make) ▼

Depending on the chosen selection, the date (Scheduling Date) next to the selection indicates either the start from or need by date. In the example below, a 'Start from date' was chosen, indicating that May 5, 2036, will be the date that this schedule starts.



← T-shirts

Start from date ▼ 05. May, 2036

Apparel Template (Make) ▼

In workbench, the attributes are recorded on the Schedule Template reference.

To configure these two attributes, see the direction below:

Scheduling method

1. Create a text-based List Of Value with the values of FIXED_END and FIXED_START in the Values field.

Note: The name and ID for this List Of Value can vary, but the values for the list of value must be FIXED_END and FIXED_START.

- Under the desired parent folder, create a new description attribute to represent the scheduling method. This attribute is externally maintained with a 'List Of Values' validation base type. Make the List of Values created in the previous step the chosen List of Values for this attribute.

< Scheduling Method - Attribute **→**

Validity Profile Log State Log Tasks

Attribute References Attribute Transformation

Description

Name	Value
ID	PLMSchedulingMethod
Name	Scheduling Method
Last edited by	2018-04-25 17:16:34 by
Full Text Indexable	No
Externally Maintained	Yes
Hierarchical Filtering	None
Calculated	No
Type	Description
Mandatory	No

Attribute Validation

Name	Value
Validation Base Type	List Of Values
List Of Values	Scheduling Method
Multi Valued	No

[Edit Validation Rule](#)

- The attribute you just created should be valid for the reference type that represents the schedule template. In this example, the attribute is called Scheduling Method, and the reference type is Schedule Template.

System Setup

- Schedule Information
 - Schedule References
 - Schedule Deadline
 - Scheduling Date
 - Scheduling Method**
 - Status Information
 - Supplier Base Information
 - PLM Solution Area
- Basic Attributes
- Board Essential Attributes
- BookAttributes
- Metadata
- PLMBoardAttributes
- PLMCollectionAttributes
- PLMColorAttributes
- PLMImageAttributes
- PLMMaterialAttributes

Scheduling Method - Validity

Validity | Profile | Log | State Log | Tasks

Attribute | References | Attribute Transformation

Valid for Entity Reference Types

- ☐ All
 - ☐ Line Plan Schedule
 - ☐ PLMBoardToPLMBoard
 - ☐ PLMBoardToPLMCollection
 - ☐ PLMCollectionToPLMCollection
 - ☐ PLMLinePlanCategory1ToScheduleCategory1
 - ☐ PLM Schedule Template Task Dependency
 - ☐ Schedule Task Dependency
 - ☒ **Schedule Template**

Scheduling date

1. Create a description attribute with an ISO Date validation base type.

Scheduling Date - Attribute

Validity | Profile | Log | State Log | Tasks

Attribute | References | Attribute Transformation

Description

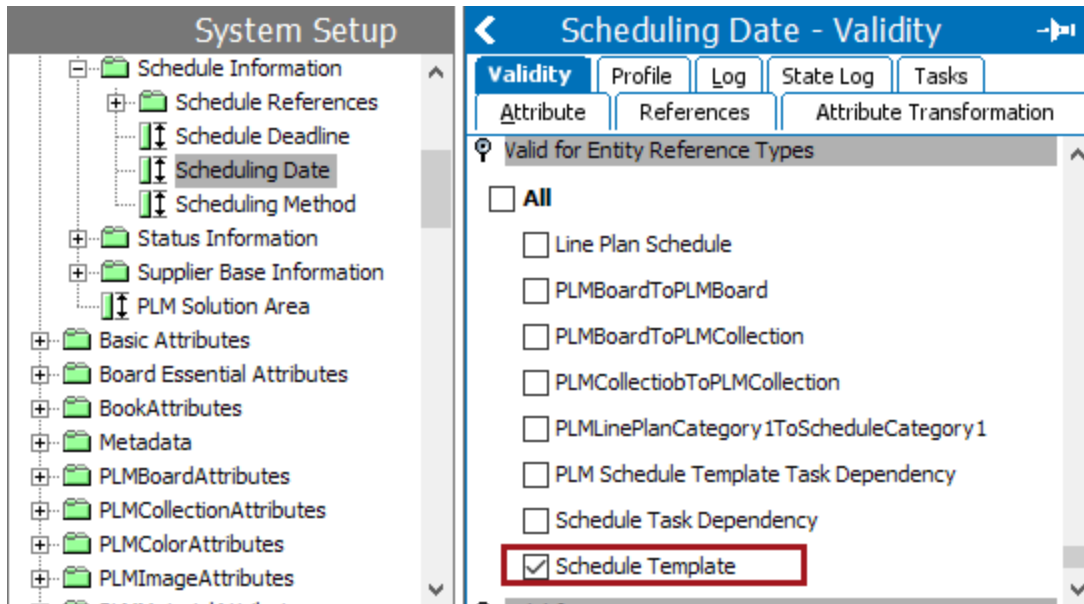
Name	Value
ID	PLMSchedulingDate
Name	Scheduling Date
Last edited by	2018-04-25 17:16:34 by
Full Text Indexable	No
Strict Validation	No
Externally Maintained	No
Hierarchical Filtering	None
Calculated	No
Type	Description
Mandatory	No

Attribute Validation

Name	Value
Validation Base Type	ISO Date

[Edit Validation Rule](#)

- The attribute to represent the scheduling date should be valid for the reference type that represents the schedule template. In this example, the attribute is Scheduling Date and the reference type is Schedule Template.



Planned Start and Planned End

When creating a schedule, a planned start date and a planned end date for each schedule task needs to be defined. This is done when a user creating the schedule selects the attributes that represent the scheduling method, scheduling date, and the template type.

← Men - Jackets

Start from date ▼ 06. May, 2018

Select template ▼

Design Brief 03/01/2018 - 12/01/2018	8
Design 15/01/2018 - 01/02/2018	14
Samples 02/02/2018 - 02/03/2018	21
Margin Review and Selection 05/03/2018 - 08/03/2018	4
Purchase Orders 05/03/2018 - 06/03/2018	2
Production 09/03/2018 - 09/03/2018	1
Delivery 12/03/2018 - 30/03/2018	15

Submit

Additionally, each template has a set number of tasks associated with the template. Each template task has a defined duration of days allotted for completing the task. If a user wishes to change the default number of days for a task, they are able to do so when creating the schedule. For more on how to create schedule templates, see the **Creating Schedule Object Types** topic in this documentation.

←

Men - Jackets

Start from date ▾

06. May, 2018

Select template ▾

Design Brief

03/01/2018 - 12/01/2018

8

Design

15/01/2018 - 01/02/2018

14

Samples

02/02/2018 - 02/03/2018

21

Margin Review and Selec...

05/03/2018 - 08/03/2018

4

Purchase Orders

05/03/2018 - 06/03/2018

2

Production

09/03/2018 - 09/03/2018

1

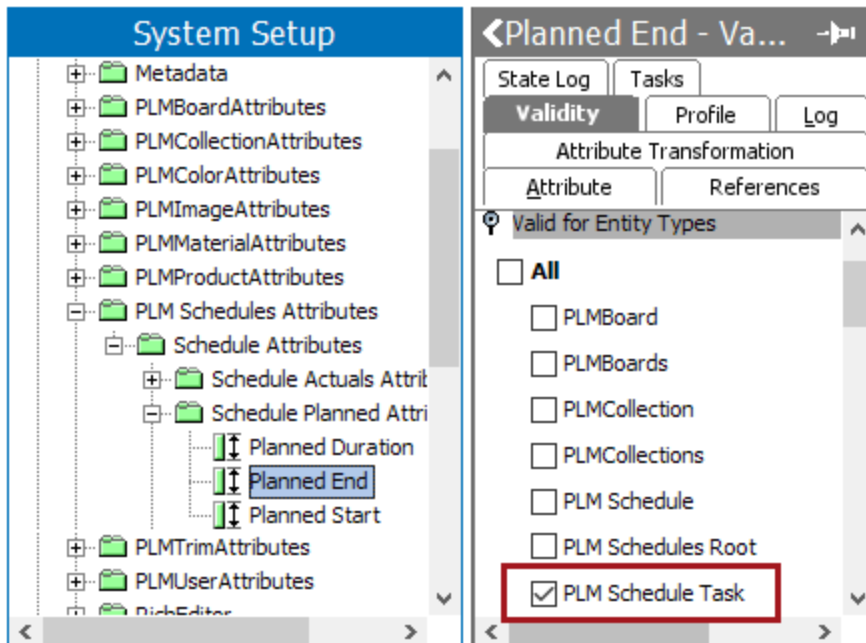
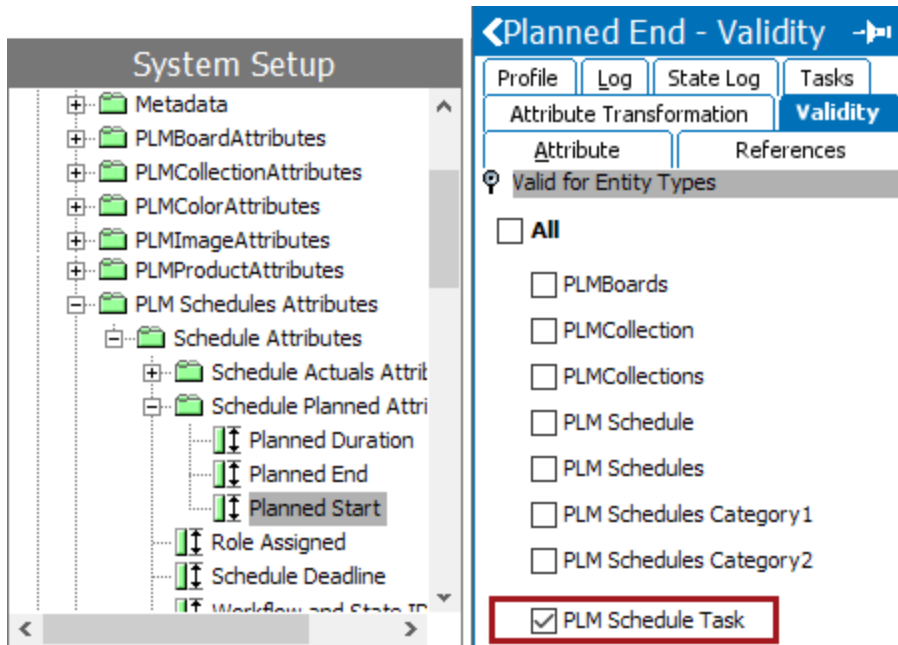
Delivery

12/03/2018 - 30/03/2018

15

Submit

Once the schedule is submitted, depending on the selections chosen for the attributes that represent the Scheduling Method, scheduling date, template type, the task dependencies, and task day durations, the planned start and planned end dates for the schedule are defined. In this example, these dates are recorded in the PLMProjectStartDateScheduled and PLMProjectEndDateScheduled attributes valid for entity type PLM Schedule Task.



In the example below, 'Men-Jackets' has been submitted as a schedule. 'Design Brief' is the first task, with nine days being determined as the duration of business working days needed to complete the task.

Tree

- Christmas 2018
 - Men - Jackets
 - Delivery
 - Design
 - Design Brief**
 - Margin Review and Selection
 - Production
 - Purchase Orders
 - Samples
 - Men - Jumpers and Cardigans
 - Men - Pants
 - Men - Shirts
 - Men - T-shirts
 - Men - Underwear
 - Women - Jackets
 - Women - Pants

< Design Brief rev.0.1 - PLM Schedul...

Referenced By	Status	State Log	Tasks
PLM Schedule Task		References	
Description			
Name	>	>	Value
ID	>		PLMScheduleTask123648
Name	>		Design Brief
Object Type	>		PLM Schedule Task
Revision	>		0.1 Last edited by USERL on
Path	>		Entity hierarchy root/Produc
Planned Start	>	31	2018-05-06
Planned End	>	31	2018-05-17
Role Assigned	>		
Workflow and State ID	>		PLMProductDevelopment_De

To create both the Planned Start and Planned End Date attributes:

1. Go to System Setup, and create these attributes in the desired attribute group.
2. Make them both description attributes with a ISO Date validation base type, and a validity for the entity type PLM Schedule Task.

Planned Duration

The attribute to represent the planned duration's value determines the baseline number of days that a particular template task should take. For example, in the Apparel Template (Buy), the template task 'Delivery' has a planned duration of 20 days as its baseline.

The screenshot displays the Stibo Systems Web UI. On the left, a 'Tree' view shows the hierarchy: Product Lifecycle Management > PLM Schedules > Templates > Apparel Template (Buy) > Delivery. On the right, the 'Delivery rev.0.2 - PLM Schedule Template Task' details are shown. The 'Planned Duration' attribute is highlighted with a red box, showing a value of 123 20 d.

Description	
Name	Value
ID	PLMScheduleTemplateTask112439
Name	Delivery
Object Type	PLM Schedule Template Task
Revision	0.2 Last edited by STEPSYS on Thu Nov 30
Path	Entity hierarchy root/Product Lifecycle Man
Planned Duration	123 20 d
Role Assigned	Suppliers
Workflow and State ID	PLMSupplierPurchasedItems_Delivery

When a user creates a schedule in Web UI, and picks the 'Apparel Template (Buy)' schedule template to use, the tasks associated with the template populate with their values pre-populated from the attribute that represents the Planned Duration attribute.

←

T-shirts

Start from date

▼

05. May, 2036

Apparel Template (Make)

▼

Design Brief

05/05/2036 - 15/05/2036

9

Design

16/05/2036 - 12/06/2036

20

Samples

13/06/2036 - 24/07/2036

30

Margin Review and Selec...

25/07/2036 - 31/07/2036

5

Purchase Orders

25/07/2036 - 31/07/2036

5

Production

01/08/2036 - 04/09/2036

25

Delivery

05/09/2036 - 02/10/2036

20

Submit

Should the user wish to alter these pre-populated values when they are creating the schedule, they are able to do so before clicking 'Submit.'

To create the attribute to represent the Planned Duration:

1. Go to System Setup and create the attribute in the desired attribute folder.
2. Make the attributes a description attribute, with a Number validation base type, that is valid for the PLM Schedule Template Task Entity type. In the example below, the attribute created is called 'Planned Duration.'

System Setup

- Attribute Groups
 - !Product Lifecycle Management Attributes
 - Basic Attributes
 - Board Essential Attributes
 - BookAttributes
 - Metadata
 - Schedule Attributes
 - Schedule Actuals Attributes
 - Actual Duration
 - Actual End
 - Actual Start
 - Schedule Task
 - Schedule Planned Attributes
 - Planned Duration**
 - Planned End
 - Planned Start
 - Role Assigned
 - Schedule Deadline
 - Schedule Is Submitted

<Planned Duration - Validity

Log
State Log
Tasks

Attribute Transformation
Validity
Profile

Attribute
References

Valid for Entity Types

☐ All

☐ PLM Schedules

☐ PLM Schedules Category1

☐ PLM Schedules Category2

☐ PLM Schedules Root

☐ PLM Schedule Task

☐ PLM Schedule Template

☐ PLM Schedule Templates

☒ **PLM Schedule Template Task**

☐ Product Lifecycle Management Entity Ro

Valid for Publication Types

Schedule Is Submitted

When a schedule is submitted in the Web UI, the schedule is no longer able to be edited and is ready for use.

Item	Value
Design Brief 08/05/2018 - 21/05/2018	10
Design 22/05/2018 - 18/06/2018	20
Samples 19/06/2018 - 30/07/2018	30
Margin Review and Selec... 31/07/2018 - 06/08/2018	5
Purchase Orders 31/07/2018 - 06/08/2018	5
Production 07/08/2018 - 10/09/2018	25
Delivery 11/09/2018 - 08/10/2018	20

Submit

The ability to submit a schedule is due to the Schedule Is Submitted attribute.

To create this attribute:

1. Go to System Setup and create the an attribute to represent the Schedule Is Submitted attribute in the desired attribute group.
2. Make the attribute a description attribute with an Integer as a validation base type.
3. The minimum value should be zero, and the maximum value should be one.
4. The validity for the Schedule Is Submitted attribute is for the PLM Schedule entity type.

Schedule Is Submitted - Attribute

Profile Log State Log Tasks

Attribute References Attribute Transformation Validity

Description

Name	Value
ID	PLMScheduleSubmitted
Name	Schedule Is Submitted
Last edited by	2017-12-08 09:05:47 by STEPSYS
Full Text Indexable	No
Externally Maintained	No
Hierarchical Filtering	None
Calculated	No
Type	Description
Mandatory	No

Attribute Validation

Name	Value
Validation Base Type	Integer
List Of Values	N/A
Multi Valued	No
Minimum Value	0
Maximum Value	1
Maximum Length	N/A

Schedule Is Submitted - Validity

Profile Log State Log Tasks

Attribute References Attribute Transformation Validity

Valid for Product Types

Valid for Classification Types

Valid for Asset Types

Valid for Entity Types

☐ **All**

- ☐ PLMBoard
- ☐ PLMBoards
- ☐ PLMCollection
- ☐ PLMCollections
- ☒ **PLM Schedule**
- ☐ PLM Schedules
- ☐ PLM Schedules Category1
- ☐ PLM Schedules Category2
- ☐ PLM Schedules Root
- ☐ PLM Schedule Task
- ☐ PLM Schedule Template
- ☐ PLM Schedule Templates
- ☐ PLM Schedule Template Task
- ☐ Product Lifecycle Management Entity Root

If a schedule has been submitted, the fields in Web UI are not able to be edited, and if looked at in the workbench, the value of '1' is in the Schedule Is Submitted field. If a change to the schedule needs to be made, you will need to delete the value of '1' in the Schedule Is Submitted attribute field. This will activate the schedule again, where the user is able to make any needed corrections.

←

Men - T-shirts

Start from date

▼

08. May, 2018

Design Brief

22/01/2018 - 31/01/2018

8

Design

01/02/2018 - 20/02/2018

14

Samples

21/02/2018 - 22/03/2018

22

Margin Review and Selec...

23/03/2018 - 27/03/2018

3

Purchase Orders

23/03/2018 - 25/03/2018

1

Production

28/03/2018 - 29/03/2018

2

Delivery

30/03/2018 - 18/04/2018

14

Submitted

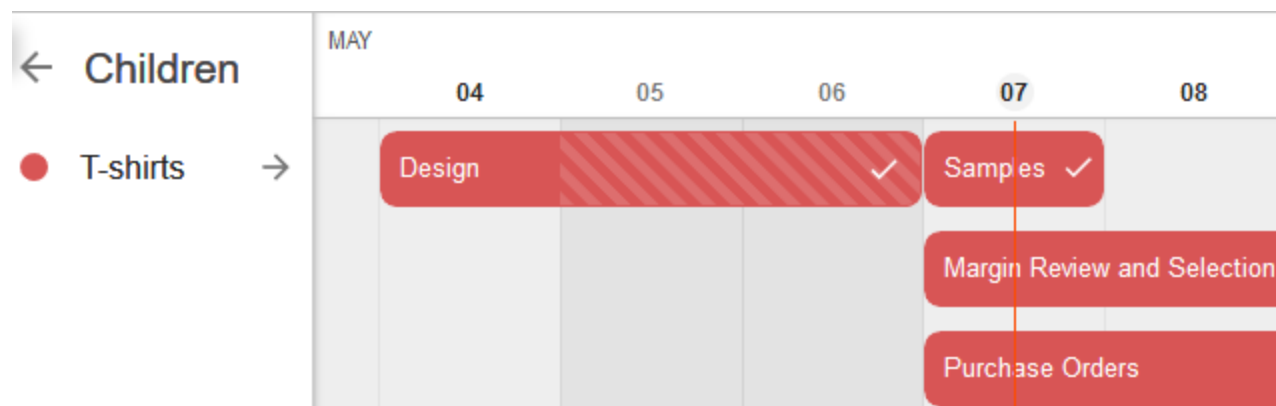
Men - T-shirts rev.0.2 - PLM Schedule			→
PLM Schedule	References	Referenced By	Status
Description			
Name	>	>	Value
> ID			PLMSchedule113984
> Name			Men - T-shirts
> Object Type			PLM Schedule
> Revision			0.2 Last edited by STEPSYS on Mon Jan 22 12:23:4
> Path			Entity hierarchy root/Product Lifecycle Managemen
> Schedule Deadline			31
> Schedule Is Submitted		123	1

Actual Start and Actual End

Through business rules, the Actual Start and Actual End attributes record when a design specification genuinely starts and finishes with a task in a schedule. The dates are recorded on the Schedule Task reference via the attributes that represent the Actual Start and Actual End dates.

rift rev.0.1 - Referenced By			
Product		Sub Products	References
Referenced By			
Line Plan References			
Schedule Actuals Attributes			
Reference Type	Source	Actual Start	Actual End
> Schedule Task +	Delivery		
	Design	2018-05-04	2018-05-06
	Design Brief		
	Margin Review and Selection	2018-05-07	
	Production		
	Purchase Orders	2018-05-07	
	Samples	2018-05-07	2018-05-07

When looking at the schedules in Web UI, you can then see a complete visual picture of what deadlines are being met or are running over.



For more on the Schedule Task reference type, see the **References Needed for Schedules** topic in this documentation.

To create the attributes to represent the Actual Start and Actual End:

1. Go to System Setup and create the attribute in the desired attribute group.
2. Create the Actual Start and Actual End attributes as description attribute, with an ISO Date validation base type, that is valid for the product reference type Schedule Task.

For more on business rules, see the **Business Rules** documentation.

Schedule Deadline

This attribute is used to represent the last date of the schedule.

In this example, it is called Schedule Deadline. It is a description attribute, with a Validation Base Type for an ISO Date. It is valid for the entity type that represents the PLM Schedule.

The screenshot displays the 'Schedule Deadline - Validity' configuration window. The 'Valid for Entity Types' section shows a list of entity types with checkboxes. 'PLM Schedule' is checked and highlighted with a red box. A red arrow points from this box to an inset window showing the 'System Setup' tree. In the tree, 'PLM Schedule' is highlighted under the 'PLM Schedules' folder, also with a red box.

Workflow and State ID

While it is not required to have workflows when using schedules, it is possible to use workflows in conjunction with schedules. When a design specification goes through the tasks in a schedule, the design specification is often also progressing through a workflow. As the design specification enters and exits each workflow state and transitions to the next state, business rules applied to states and transitions can capture the actual start and actual end dates of each task. These actual start and actual end dates can then be recorded on attributes on the Schedule Task reference and also display visually on the schedule. See the above topic in this documentation for more on actual start and actual end date attributes. For more on workflows see the **Workflows** topic in the **Workflows** documentation. For more on business rules see the **Business Rules** topic in the **Business Rules** documentation.

In order for schedule template tasks and workflows to work together seamlessly, the schedule task template needs to be told what workflow and state ID to look at for the entry and exit of tasks in workflow states, thus helping to provide the actual start and actual end dates. This is accomplished via the attribute that represents the Workflow and State ID.

Tree

- Product Lifecycle Management
 - PLM Schedules
 - Schedules
 - Templates
 - Apparel Template (Buy)
 - Delivery
 - Purchase Orders
 - Samples
 - Supplier Selection
 - Apparel Template (Make)
 - Delivery
 - Design
 - Design Brief**
 - Margin Review and
 - Production
 - Purchase Orders

Design Brief rev.0.2 - PLM Schedule Te...

Status | State Log | Tasks | References | Referenced By

PLM Schedule Template Task

Description

Name	Value
ID	PLMScheduleTemplateTask112429
Name	Design Brief
Object Type	PLM Schedule Template Task
Revision	0.2 Last edited by STEPSYS on Thu Nov
Path	Entity hierarchy root/Product Lifecycle M
Planned Duration	123 10 d
Role Assigned	Designers
Workflow and State ID	PLMProductDevelopment_DesignBrief

New Product Development (example) (PLMProductDevelopment)

```

graph TD
    DesignBrief[Design Brief] -- Submit --> Design[Design]
  
```

To configure the Workflow and State ID attribute:

1. Go to System Setup and create an attribute to represent the Workflow ID and State ID List of Values in the desired folder.

2. Make the Workflow ID and State ID List of Values a Text validation base type and add the values. The values need to be the workflow ID in accompaniment to every workflow state ID that needs to be captured. The ID pattern 'Workflow ID_State ID' is used for the values in this example.

Workflow ID and State ID - List of Values

List of Values
References
Log
State Log
Tasks

Description

Name	Value
ID	PLMWorkflowIDAndStateID
Name	Workflow ID and State ID
Edited by	2017-12-06 10:55:01 by STEPSYS
Path	Lists of Values / LOVs/!Product Lifecycle ...
Use Ids on values	No
Use Ids for sorting	No

In Attribute Groups

List of Values Validation

Name	Value
Validation Base Type	Text
Allow Users to Add Values	No
Mask	

Values

Values

- > PLMProductDevelopment_Delivery
- > PLMProductDevelopment_Design
- > PLMProductDevelopment_DesignBrief
- > PLMProductDevelopment_MarginReviewAndSelection
- > PLMProductDevelopment_Production
- > PLMProductDevelopment_PurchaseOrders
- > PLMProductDevelopment_Samples
- > PLMSupplierPurchasedItems_Delivery
- > PLMSupplierPurchasedItems_Purchase Orders
- > PLMSupplierPurchasedItems_Samples
- > PLMSupplierPurchasedItems_SupplierSelection

Add Value

3. Next, in the desired attribute group, create a description attribute with a 'List Of Values' validation base type.
4. Select the List of Values created to hold the workflow ID and state ID that you want the attribute to refer to.
5. Make the validity for the Workflow and State ID attribute valid for the PLM Schedule Task and PLM Schedule Template Task entity object types.

Workflow and State ID - Validity

Validity | Profile | Log | State Log | Tasks

Attribute | References | Attribute Transformation

Valid for Entity Types

- ☐ All
- ☐ PLM Schedule
- ☐ PLM Schedules
- ☐ PLM Schedules Category1
- ☐ PLM Schedules Category2
- ☐ PLM Schedules Root
- ☒ PLM Schedule Task
- ☐ PLM Schedule Template
- ☐ PLM Schedule Templates
- ☒ PLM Schedule Template Task
- ☐ Product Lifecycle Management Entity Root

Valid for Publication Types

6. To apply the Workflow and State ID after creating a schedule template, click on one of the tasks in the schedule template.
7. Navigate to the Workflow and State ID attribute, and select the desired workflow and state ID from the dropdown menu.

<
Supplier Selection rev.0.2 - PLM Schedule Template Task
→

PLM Schedule Template Task

References

Referenced By

Status

State Log

Tasks

Description

Name	Value
ID	PLMScheduleTemplateTask112438
Name	Supplier Selection
Object Type	PLM Schedule Template Task
Revision	0.2 Last edited by STEPSYS on Wed Dec 06 05:06:50 UTC 2017
Path	Entity hierarchy root/Product Lifecycle Management/PLM Schedules...
Planned Duration	123 5 d
Role Assigned	Buyer
Workflow and State ID	PLMSupplierPurchasedItems_SupplierSelection PLMProductDevelopment_MarginReviewAndSelection PLMProductDevelopment_Production PLMProductDevelopment_PurchaseOrders PLMProductDevelopment_Samples PLMSupplierPurchasedItems_Delivery PLMSupplierPurchasedItems_Purchase Orders PLMSupplierPurchasedItems_Samples PLMSupplierPurchasedItems_SupplierSelection

Schedule Setup in Web UI

Users interact with schedules only in the Web UI, thus it is important to have proper configuration so that the user has a seamless experience.

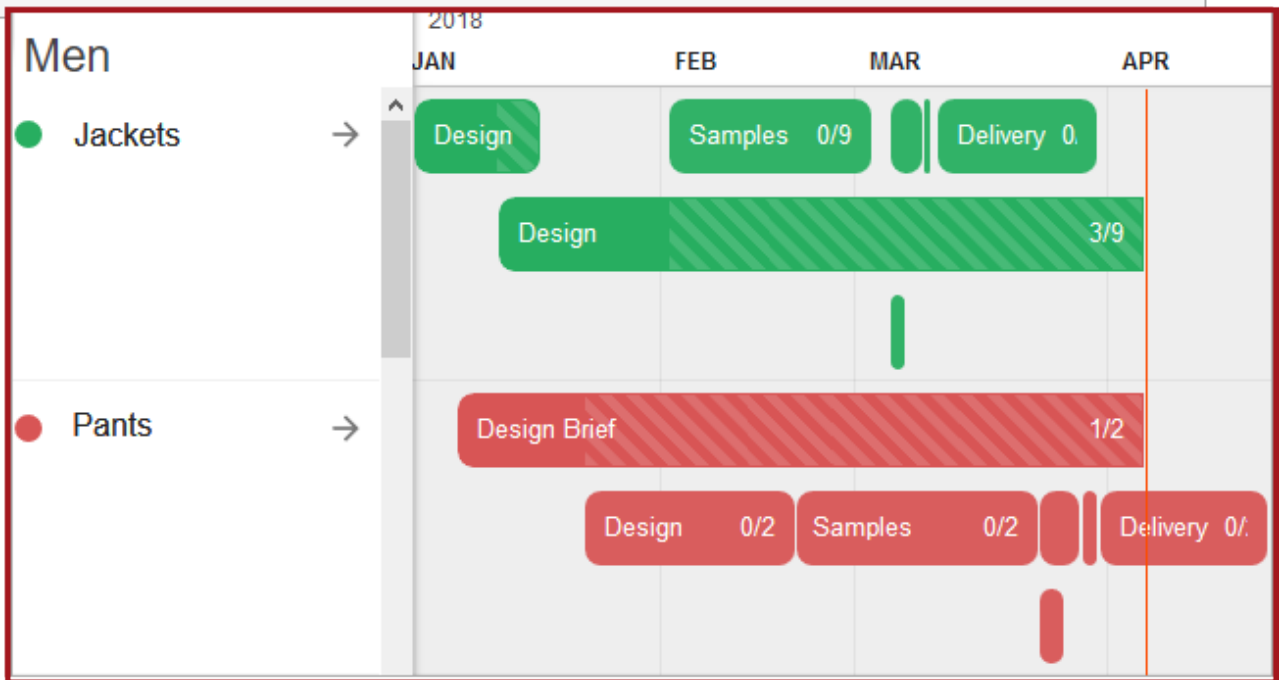
Prerequisite

Before any Web UI setup can occur, proper configuration in workbench must be in place. Be sure to read through all of the setup schedule topics for workbench in this documentation before attempting to configure schedules in Web UI.

PLM Schedule Template

The PLM Schedule Template screen is the landing page for schedules after a user clicks on the '**Schedule**' hyperlink in the approved line plan.

← Christmas 2018		Business Category:	Apparel	
		Line Plan Start Date:	2018-03-06	
		Schedule Deadline:	2018-06-06	
		Line Plan Status:	In Progress	
Edit Line Plan		682153.96	549542.74 / 2000000	3.57 / 54
		Cost	Revenue	Gross Margin % Actions
^	Men	649489.96	501524.74	5.36 Schedule
	Jackets (9 styles)	642574	493682	-3.87
	Pants (1 styles)	319.02	543.14	41.26



This screen displays the approved line plan in a schedule format, with all of its subcategories, and any task bar graphs for submitted subcategory schedules.

To create this PLM Schedule Template screen, follow the directions below:

1. In Web UI, open the designer and create a new PLM Schedule Template screen. Enter in a recognizable Screen ID, and click **Add**.

Add Screen ✕

Screen ID 2

PLM Schedule Template

Node List Browser

PLM Line Plan

PLM Schedule Template 1

Packaging

Planned Spread Screen

Power Search

Print On Demand

Recycle Bin Screen

Search Statistics

A screen that allows to define line schedule from template

Filter

☐ Show deprecated components

3

✓ Add

✕ Cancel

- In the PLM Schedule Template Properties, click the ellipsis button (...) to the right of each parameter field to select the appropriate attributes, object types, and references. A complete list of what is needed is labeled below.

Add component - configure required properties



Required properties (*) must be set before the component can be added to the configuration.

PLM Schedule Template Properties

Component Description

A screen that allows to define line schedule from template

Actual End Date *		...
Actual Start Date *		...
Line Plan Schedule Task *		...
Planned Duration *		...
Planned End Date *		...
Planned Start Date *		...
Schedule *		...
Schedule Deadline *		...
Schedule Is Submitted *		...
Schedule Task *		...
Schedule Task Dependency *		...
Schedule Task Design Specification *		...
Schedule Task Group *		...
Schedule Task Template *		...

- **Actual End Date:** Through business rules, this is the attribute that records when a design specification genuinely finishes with a task in a schedule.
- **Actual Start Date:** Through business rules, this is the attribute that records when a design specification genuinely starts with a task in a schedule.
- **Line Plan Schedule Task:** The reference that connects the classification object type for line plan category level 2 to entity type schedule level 2.
- **Planned Duration:** The attribute that represents the baseline number of days that a particular template task should take.
- **Planned End Date:** The attribute created to hold the value for the planned end date that a task in a schedule should have.
- **Planned Start Date:** The attribute created to hold the value for the planned start date that a task in a schedule should have.
- **Schedule:** The alternate classification object type for line plan category level 1.
- **Schedule Deadline:** The attribute used to represent the last date of the schedule.
- **Schedule Is Submitted:** The attribute created for when the schedule is submitted.
- **Schedule Task:** The object type of the tasks in the schedule.
- **Schedule Task Dependency:** The entity reference type created that determines the order, or dependencies, of the tasks in the schedules.
- **Schedule Task Design Specification:** The reference type created that enables schedules and design specifications to work together.
- **Schedule Task Group:** The entity type that represents the second level in the schedule.
- **Schedule Task Template:** The entity type created to represent the third level of the schedule task templates.

- **Schedule Task Template Dependency:** The entity reference type that determines the order, or dependencies, of the schedule template tasks for the schedules.
 - **Schedule Template:** The entity type created to represent the second level of the schedule task templates.
 - **Schedule Templates Root:** The entity type created to be the root node for templates.
 - **Schedule To Template:** The entity reference used between the schedule and the schedule templates.
 - **Scheduling Date:** The attribute created to indicate the actual date that is either the start from or need by date.
 - **Scheduling Method:** The attribute that indicates if the schedule being created is either a 'Start from date' or 'Need by date.'
 - **Workflow And State:** The attribute that is used to point to what workflow and state IDs should be used in conjunction with schedules.
3. Once all configurations are set, click Add.
 4. Now that the PLM Schedule Template Screen is configured, it needs to be saved and mapped. Click **Save** and go to --[MAIN]-- in the designer.
 5. Under Mappings, click **Add...** to add a new screen mapping.
 6. In Screen Mapping Properties, add an Object Type Condition.

Edit component

Screen Mapping Properties

Component Description A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

Conditions*

Add... Edit... Remove Up Down

Screen*

adjustschedule
Add

Add Component

Collection Group Condition
Compare Boms Condition
Entity Condition
Golden Record Condition
Matching Algorithm Condition
Merge Duplicates Condition
Multi Node Selection Condition
Object Type Condition

A condition that is true if the node is of the specified object type

Filter
☐ Show deprecated components

✓ Add

✗ Cancel

- In the Object Type Condition, add the first level entity type for schedules, and click **OK** then **Add**.

Add component - configure required properties

Required properties (*) must be set before the component can be added to the configuration.

Object Type Condition Properties

Component Description A condition that is true if the node is of the specified object type

Object Type*

Select Node(s)

Browse Search

- Entity user-type root (Entity user-type root)
 - PLMBoards (PLMBoards)
 - PLMCollections (PLMCollections)
- Product Lifecycle Management Entity Root (ProductLifecycle...)
 - PLM Schedules Root (PLMSchedulesRoot)
 - PLM Schedules (PLMScheduleRoot)
 - PLM Schedule (PLMSchedule)
 - PLM Schedules Category1 (PLMSchedulesCategory1)**
 - PLM Schedule Templates (PLMScheduleTemplateRoot)
 - Publication section types (Section root)

8. In the Screen dropdown field, select the screen for PLM Schedule Template, and click Save.

Edit component

Screen Mapping Properties

Component Description

A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

Conditions *

ObjectType = PLMSchedulesCategory1

Add... Edit... Remove Up Down

Screen *

Add

PLM Schedule Template

Save
Cancel

9. Adjust the screen up or down in the mappings as needed.

10. Click **Save** in the designer before closing it out.

For more information on attributes, object types, and references needed for schedule setup in the PLM Schedule Template screen, see the following topics in this documentation:

- Creating Schedule Object Types
- References Needed for Schedules
- Setting Up Attributes for Schedules
- References Needed for Line Plans

For more on how to set up and use schedules after configuring the PLM Schedule Template screen, see the **Creating Schedules** topic in the **PLM for Users** documentation.

© Stibo Systems - Confidential - Release 9.3-MP3 (April 2020)

112

Private Label Food Solution

The private label food solution enables companies to create products from the ground up. They can create a product idea, decide upon the flavors of the product, and create and agree upon a recipe with their suppliers for the regions they plan to sell the product.

There are a number of object types, attributes, and reference types that need to be created and properly set up in workbench, and a number of component tabs needed in Web UI for the private label food solution to function properly. For more information, see the following sections below on how to properly set up the private label food solution in workbench and Web UI.

- Private Label Food Solution Setup in Workbench
- Private Label Food Solution Setup in Web UI

Once proper set up of the private label food solution has been established, it is also possible to set up Change Reports. This allows users to see a report of changes to data between a specific event and the current data. For more information on how to set up Change Reports, see the **Change Reports** topic in this documentation.

Note: When a finalized PLM project is awarded, it is now possible to transform each label within the awarded project to a PMDM product through the use of business rules. For more information see the **PLM to PMDM Transformation Overview** topic in the **PLM Solution Enablement** section of the **Solution Enablement** documentation.

The following licenses and components are needed for the private label food solution to work:

Required License	Components needed
X.FlexTableHeader	N/A
N/A	spire-plm
N/A	private-label-food

Change Reports

Users can use Change Reports to see a report of changes to data between a specific event and the current data. Change Reports are populated via highly configurable 'snapshots'. These snapshots can take place when there is a change in a workflow state, a trigger by an inbound or outbound integration endpoint, or a change by derived events through an Event Processor. Change reports can be added on a Node Details screen via a Product Summary Card.

For more on Product Summary Cards, see the **Below Title Component** topic in the **Web User Interfaces / Web UI Setup and User Guide** documentation.

Change Reports highlight any changes that have been made during user-defined events so important changes are not overlooked accidentally. This is different than revision history in that administrators have full control of when Change Reports are created, and what they want to capture, such as data that was added, deleted, or changed on the object, the reference target of an object, or on reference metadata. Snapshots have the added benefits of not being impacted by changes to the data model. All of this helps product teams notice important changes to the data that may otherwise be overlooked while collaborating with other users.

Change Report Limitations

- Change Reports do not support snapshots of publications, assets, data containers, children, or data model (set up) changes.
- Only one version of the snapshot will be kept.
- Sort order of the target of the changes within the report is alphanumeric and cannot be changed.

Note: To access and use the full set of Change Report functionality, including snapshot configuration, a 'change-reports' add-on component must both be activated on your system. See your Stibo Systems representative for details.

Configuring Change Reports requires set up in both workbench and Web UI.

Set up performed in Workbench

The following topics explain the setup and configurations needed in workbench for Change Reports.

- Object, References, and Attributes for Change Reports
- Snapshot Configurations
- Snapshot Recorders
- Business Rules and Snapshots

Set up performed in Web UI

When implementing Change Reports in Web UI, Change Reports can be added on a Node Details screen via a Product Summary Card. The following topics explain the setup, configuration, and use of Change Reports.

- For more information on how to add and configure the Product Summary Card, see the **Below Title Component** topic in the **Web User Interfaces / Web UI Setup and User Guide** documentation.
- Using Change Reports

The following licenses and components are needed for change reports to work:

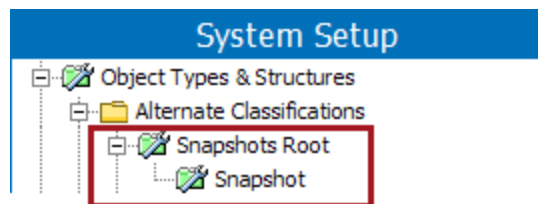
Required License	Components needed
X.ChangeReport	change-reports
N/A	spire-plm
N/A	private-label-food

Object Types, References, and Attributes for Change Reports

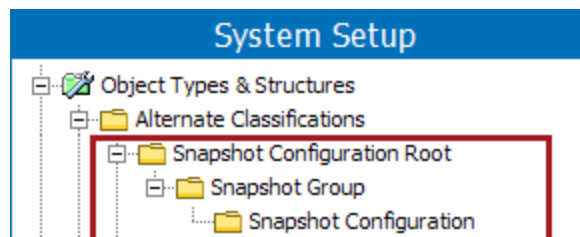
There are a number of object types, references, and attributes needed for Change Reports to work. An XML file is available for implementation purposes. Please talk to your Stibo Systems representative for more information on the XML file. After implementation, all IDs and ID patterns should be exactly like the ones provided bellow.

Note: While Change Reports are used in Web UI, set up for Change Reports in the creation of object types, references, and attributes can only be done in workbench.

Classifications

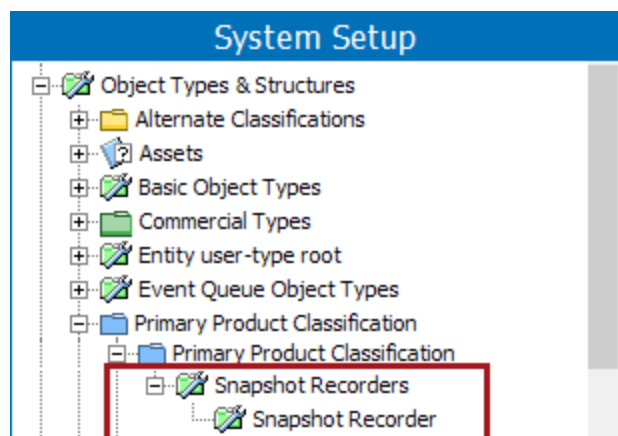


Object Type ID	ID Pattern	Purpose
SnapshotsRoot	SnapshotsRoot	Root level for storage of snapshots
Snapshot	ss-[id]	Stores a snapshot in JSON format



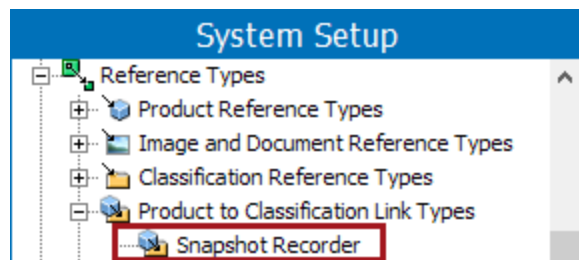
Object Type ID	ID Pattern	Purpose
SnapshotConfigurationRoot	SnapshotConfigurationRoot	Root level for snapshot configuration
SnapshotGroup	ssg-[id]	A self-referenced object type, allowing for organization of Snapshot Configurations
SnapshotConfiguration	sscnf-[id]	Defines the context and event ID

Products



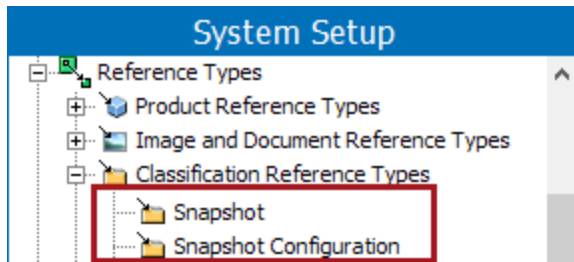
Object Type ID	ID Pattern	Purpose
SnapshotRecorderRoot	SnapshotRecorderRoot	Stores the recorder and tab name for the Change Report
SnapshotRecorder	ssrec-[id]	Stores the recorder and tab name for the Change Report

Reference Types



Object Type ID	ID Pattern	Valid Target	Purpose
SnapshotRecorder	SnapshotRecorder	SnapshotConfiguration	Link a Snapshot Recorder to one of many Snapshot Configuration types

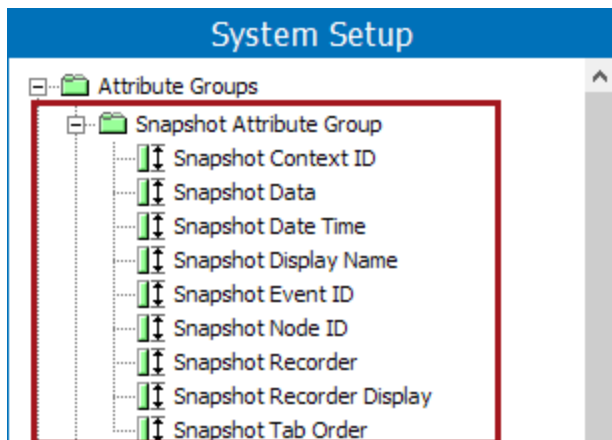
Classification Reference Types



Object Type ID	Valid Source	Valid Target	Purpose
Snapshot	Valid from any object that a snapshot is taken of	Snapshot	Relates the object that has a snapshot to its snapshot
SnapshotConfiguration	Snapshot	SnapshotConfiguration	Links the Snapshot Configuration with the snapshot

Attributes

All attributes are description attributes.



Attribute ID	Type	Valid For	External	Purpose
SnapshotContextID	LOV with ID Multi Valued	SnapshotConfiguration	Yes	Which context(s) the snapshot should be taken in

Attribute ID	Type	Valid For	External	Purpose
SnapshotData	Text	Snapshot	Yes	Store the JSON snapshot
SnapshotEventID	LOV with ID	Snapshot, SnapshotConfiguration	Yes	Event ID for the Snapshot
SnapshotNodeID	Text	Snapshot	Yes	Node ID the snapshot was done for
SnapshotRecorder	Text	SnapshotRecorder	Yes	Specifies which values to store for the snapshot
SnapshotDataTime	ISO Date and Time	Snapshot	Yes	When the snapshot was taken
SnapshotDisplayName	Text	Link Type: SnapshotRecorder	No	Name of the tab in the change report, to be translated
SnapshotRecorderDisplay	LOV (values Yes or No)	Link Type: SnapshotRecorder	Yes	Indicates if the recorder tab should be included in the change report
SnapshotTabOrder	Number	Link Type: SnapshotRecorder	Yes	Order of the tab in the change report

Snapshot Configurations

Snapshot Configurations are used in the Change Reports data model to help capture changes to data that were made at specific events. Change Reports are populated via highly configurable 'snapshots'.

Snapshot Configurations are set up as classifications in workbench and used for screens in Web UI. They determine which contexts to capture data from (take a snapshot), and any customer-defined change in events. This could be a change in a workflow state, a trigger by an inbound or outbound integration endpoint, or a change by derived events through an Event Processor. When the event takes place, the Snapshot Recorder captures the data at that point in time if configured by the Snapshot Configuration.

For more on Snapshot Recorders, see the **Snapshot Recorders** topic in this documentation.

For more information on Change Reports, see the **Change Reports** topic in this documentation.

Important: While Change Reports are used in Web UI, set up for Snapshot Configurations can only be done in workbench.

Recipe Specification rev.0.2 - Classification		
Referenced By	Images & Documents	Tables
Status	State Log	Tasks
Classification		Sub Products
		References
Description		
Name	>	Value
ID		sscnf-149070
Name		Recipe Specification
Object Type		Snapshot Configuration
Revision		0.2 Last edited by USERL on Thu Jun 20 17:28:17 EDT 2019
Approved		✖ Never Been Approved
Translation		Not Translated
Path		Classification 1 root/Snapshot Configuration/Snapshot Group/Recipe Specification
Visibility		
Snapshot Context ID		Context1 Context2
Snapshot Event ID		Supplier Response

Snapshot Configurations store context IDs (SnapshotContextID), a mandatory attribute that indicates which contexts the snapshots will be taken in.

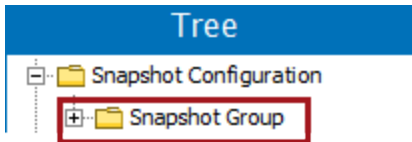
Additionally, an event ID (SnapshotEventID) is stored on the Snapshot Configuration, also a mandatory attribute. This attribute determines what event will trigger a snapshot. Snapshots compare the data at the time of the event with the current data. For example, as seen pictured above, the event ID could be the 'Supplier Response'.

For a list of all data scenarios that can be in a snapshot, see the **Elements in a Snapshot** topic in this documentation.

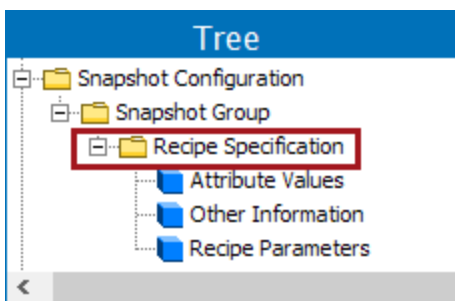
Create Snapshot Configurations

Below are the steps to create a Snapshot Configuration.

1. In Tree, in Classifications, under the parent 'SnapshotConfigurationRoot', add a Snapshot Group to hold all Snapshot Configurations for a needed project.



2. Under your Snapshot Group, create as many child Snapshot Configurations as needed. A Snapshot Configuration will hold all needed Snapshot Recorders.



For more on Snapshot Recorders, see the **Snapshot Recorders** topic in this documentation.

Snapshot Recorders

Snapshot Recorders are used in the Change Reports data model to help capture data at a specific event. Change Reports are populated via highly configurable 'snapshots,' where the Snapshot Recorders aid in capturing the change for the Change Report.

Snapshot Recorders are Product object types that store a recorder text in the format of a JSON string that defines which data should be captured in the snapshot.

For more information on Change Reports, see the **Change Reports** topic in this documentation.

Note: While Change Reports are used in Web UI, setup for Snapshot Recorders can only be done in workbench.

The screenshot displays the Stibo Systems Workbench interface. On the left, a 'Tree' pane shows a hierarchical structure of product lifecycle management objects. The 'Food Supplier Recipe Parameters' object is selected. The right pane, titled 'Food Supplier Recipe Parameters rev.0.1 - Product', shows the configuration details for this object type. The configuration is organized into tabs: Product, References, Referenced By, Images & Documents, Commercial, Tables, Proof View, Status, State Log, and Tasks. The 'Product' tab is active, showing a table with the following data:

Name	Value
ID	PLMFoodSupplierRecipeParameters
Name	Food Supplier Recipe Parameters
Object Type	Snapshot Recorder
Revision	0.1 Last edited by JJSO on Wed Jun 19 14:53:07 EDT 2019
Approved	✖ Never Been Approved
Translation	Not Translated
Path	Primary Product Hierarchy/Product Lifecycle Management/Snapshot Recorders/Food Supplier Recipe Parameters
SnapshotRecorder	abc

The 'SnapshotRecorder' field contains a JSON string defining the snapshot configuration:

```
{
  "references": {
    "PLMParameter": {
      "attributes": [
        "PLMMeetsRequirement",
        "PLMMethod",
        "PLMAdditionalSupplierComments"
      ],
      "flexAttributes": [
        "PLMDefiningAttribute"
      ]
    }
  }
}
```

They can be re-used for multiple Web UI tabs or multiple Web UIs where the same information should be captured by linking them into one or more Snapshot Configurations via the classification link type 'SnapshotRecorder'. See below in this topic on how to configure Snapshot Recorders and link to Snapshot Configurations.

For more information on Snapshot Configurations, see the **Snapshot Configurations** topic in this documentation.

Name	Value
ID	PLMFoodSupplierRecipeParameters
Name	Food Supplier Recipe Parameters
Object Type	Snapshot Recorder
Revision	0.1 Last edited by JJSO on Wed Jun 19 14:53:07 EDT 2019
Approved	✖ Never Been Approved
Translation	Not Translated
Path	Primary Product Hierarchy/Product Lifecycle Management/Snapshot Recorders/Food Supplier Recipe Parameters
SnapshotRecorder	<pre>{ "references": { "PLMParameter": { "attributes": ["PLMMeetsRequirement", "PLMMethod", "PLMAdditionalSupplierComments"], "flexAttributes": { "PLMDefiningAttribute" } } } }</pre>

Snapshot Recorders can be created for each Web UI tab or for processing logic, such as in a workflow, and are configured by each customer as customer needs differ.

On each Snapshot Recorder, on the Product to Classification Link Type 'SnapshotRecorder,' there are metadata attributes maintained. These determine factors about the Change Report so that each time the snapshot recorder is re-used, the information can be different.

The necessary attributes used as metadata are:

- **Snapshot Display Name** (SnapshotDisplayName): Contains the value that will be displayed in the change comparison dialog. It can be translated into multiple languages using the standard processes for translating products if needed.
- **Snapshot Recorder Display** (SnapshotRecorderDisplay): Is a 'Yes / No' LOV which determines if the Snapshot Recorder should be displayed in the Web UI.
- **Snapshot Tab Order** (SnapshotTabOrder): Determines the display order that the tab should appear in Web UI.

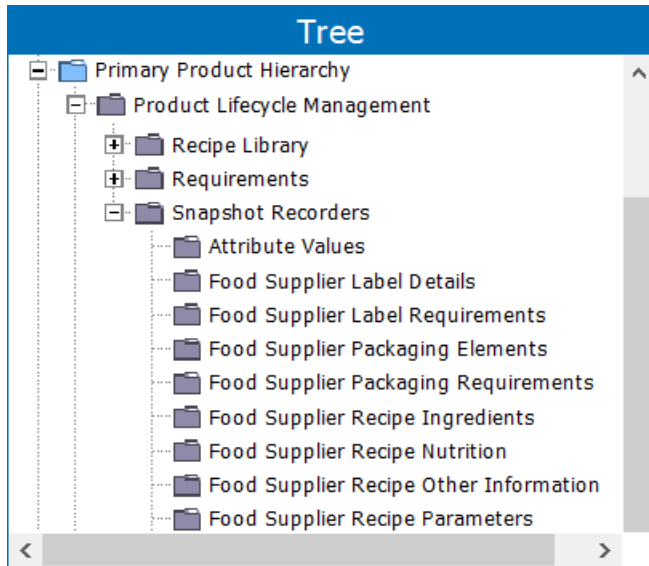
Food Supplier Recipe Parameters rev.0.1 - References						
Product	References	Referenced By	Images & Documents	Commercial	Tables	Proof View
Ungrouped Classification Links						
Reference Type	Target	Snapshot Display Name	Snapshot Recorder Display	Snapshot Tab Order		
Snapshot Recorder	Food Supplier Recipe Configurations	Recipe Parameters	Yes	20		

For more information on attributes needed for Snapshot Recorders and for Change Reports see the **Objects Types, References, and Attributes for Change Reports** topic in this documentation.

Create Snapshot Recorders

Below are the steps to create a Snapshot Recorder.

1. In Tree, in the Primary Product Hierarchy, under the parent 'SnapshotRecordersRoot,' create as many children Snapshot Recorders as needed.



2. Create the needed recorder text, in the format of a JSON string, for the Snapshot Recorders attribute.

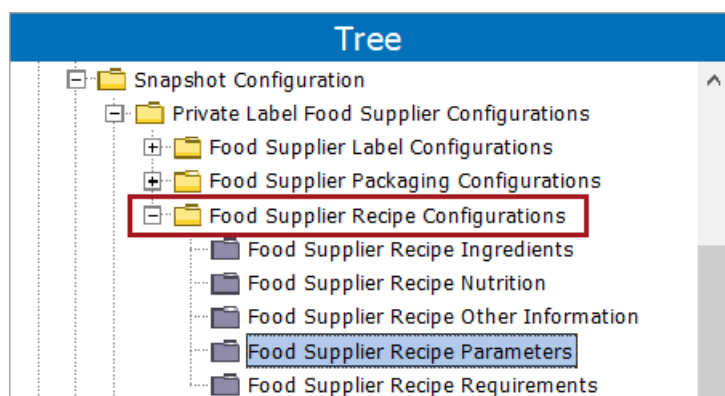
Food Supplier Recipe Parameters rev.0.1 - Product		
Product	References	Referenced By
Images & Documents	Commercial	Tables
Proof View	Status	State Log
Tasks		
Description		
Name	>	Value
ID	>	PLMFoodSupplierRecipeParameters
Name	>	Food Supplier Recipe Parameters
Object Type	>	Snapshot Recorder
Revision	>	0.1 Last edited by JJSO on Wed Jun 19 14:53:07 EDT 2019
Approved	>	Never Been Approved
Translation	>	Not Translated
Path	>	Primary Product Hierarchy/Product Lifecycle Management/Snapshot Recorders/Food Supplier Recipe Parameters
Snapshot Recorder	abc	<pre>{ "references": { "PLMParameter": { "attributes": ["PLMMeetsRequirement", "PLMMethod", "PLMAdditionalSupplierComments"], "flexAttributes": ["PLMDefiningAttribute"] } } }</pre>

- On the Reference tab, select the target Snapshot Configuration, and fill out the metadata attributes: Snapshot Display Name, Snapshot Recorder Display, and Snapshot Tab Order.

Food Supplier Recipe Parameters rev.0.1 - References					
Product	References	Referenced By	Images & Documents	Commercial	Tables
Ungrouped Classification Links					
Reference Type	Target	Snapshot Display Name	Snapshot Recorder Display	Snapshot Tab Order	
Snapshot Recorder	Food Supplier Recipe Configurations	Recipe Parameters	Yes	20	

For more information on attributes needed for Change Reports, see the **Object Types, References, and Attributes for Change Reports** topic in this documentation.

When configured correctly, the Snapshot Recorder will display under the specified Snapshot Configuration.



For more information on Snapshot Configurations, see the **Snapshot Configurations** topic in this documentation.

Business Rules and Snapshots

Change Reports are used to see a report of changes to data that were made between a specific event and the current data. Change Reports use snapshots and business rules to determine these changes.

Prerequisites

All object types, references, and attributes for Change Reports need to be created first before business rules for snapshots can be used. Additionally, all configurations need to be in place before business rules can work. See the **Change Reports** topic in this documentation for more details.

Snapshot Bind

Business rules for snapshots are created by customers and will differ depending on customer needs. They determine when snapshots for Change Reports should take place and can be applied on workflow transitions, or any other event that supports executing business rules.

For more on business rules in general, see the **Business Rules** documentation.

All business rules used for setup need to include the 'snapshot' bind so that it is possible to obtain a bind to the instance of a snapshot. This bind is applicable for Business Actions, Business Conditions, and Business Functions. With this 'snapshot' bind, the configured business rule has access to the public methods for creating a snapshot and to flag changes.

This bind has two public methods that can be used in a business rules. These methods are:

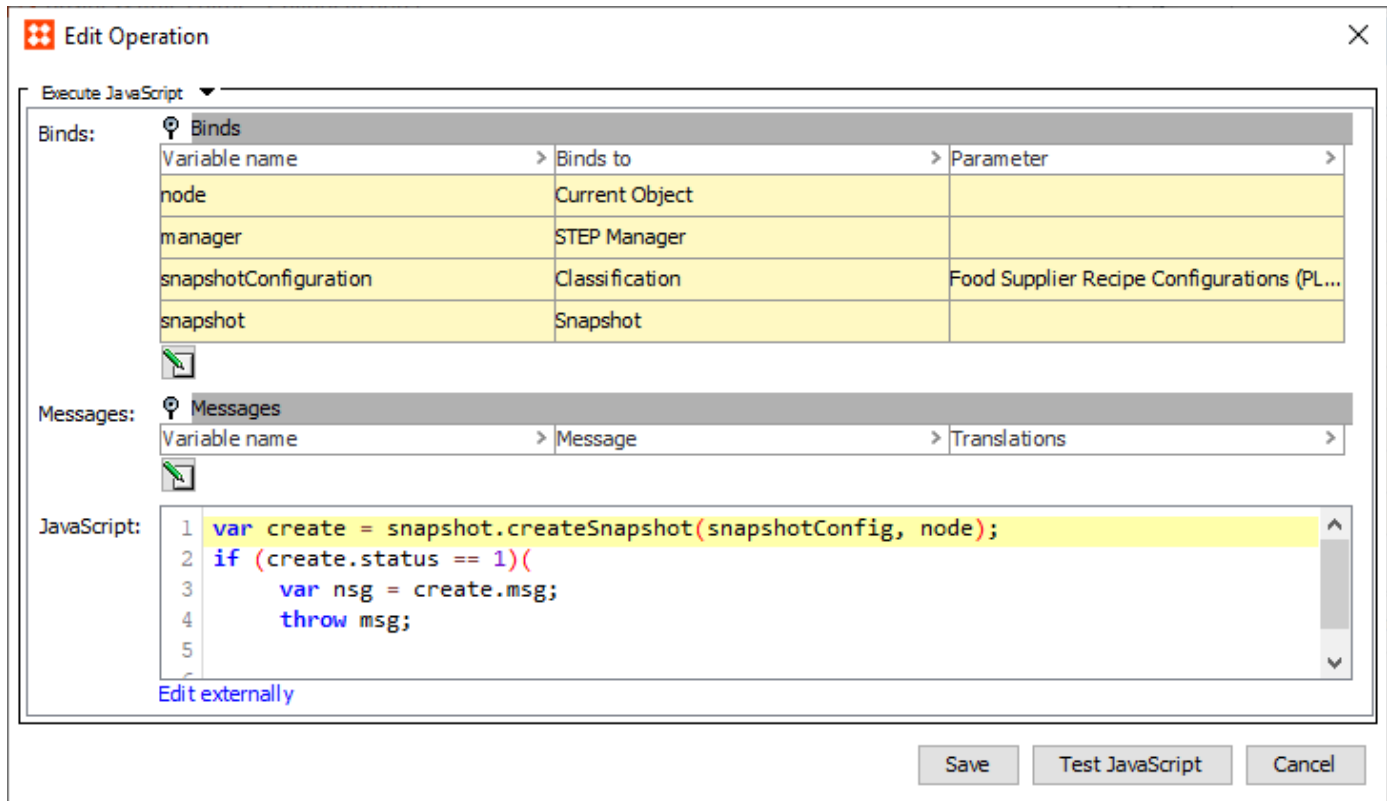
- Create Snapshot (createSnapshot)
- Flag Changes (hasChanges)

Important: While Change Reports are used in Web UI, the configuration outlined in this topic can only be done in workbench.

Creating Snapshots

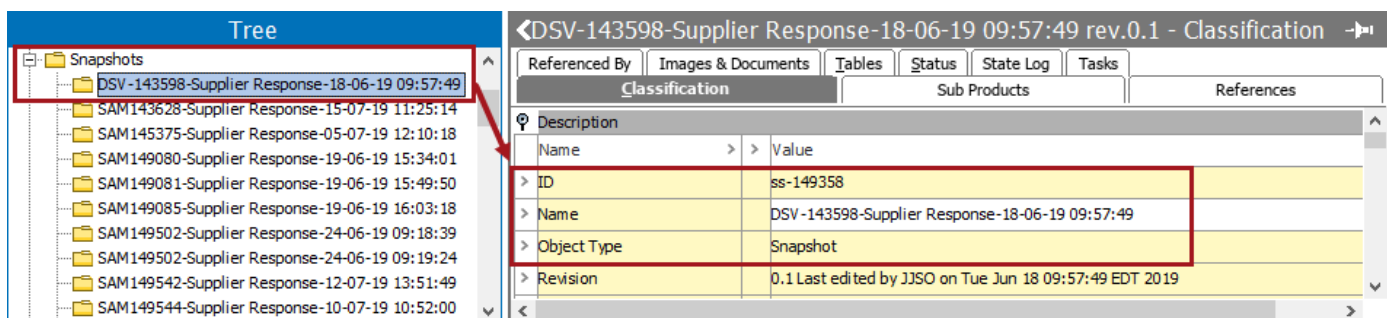
The public method for creating a snapshot, 'createSnapshot', has two arguments:

- A classification bind to the desired Snapshot Configuration.
- An instance of the node the snapshot should be taken of the object of interest that will have changes.



When using this method, not only is a snapshot created, but also the following will occur:

1. In Tree, under the 'Snapshots Root' classification folder, a new child with the object type 'Snapshot' is created. This child classification is named using the standard naming conventions of 'node ID + Event ID + Date / Time,' where the 'node ID' is the ID of the object that is being monitored for a snapshot. See image below.



2. The attribute Event ID 'SnapshotEventID' is both valid for the 'Snapshot' and 'SnapshotConfigurin'. During the snapshot, the Event ID 'SnapshotEventID' is populated by copying the value from the desired Snapshot Configuration's Event ID, 'SnapshotEventID'.
3. The attribute Node ID 'SnapshotNodeID' is populated with the ID of the node that is being monitored for changes.
4. The Date Time attribute, 'SnapshotDateTime', is populated with the current date and time in the ISO Date and Time format.

DSV-143598-Supplier Response-18-06-19 09:57:49 rev.0.1 - Classification		
Referenced By	Images & Documents	Tables
Status	State Log	Tasks
Classification		Sub Products
		References
Description		
Name	>	Value
ID		ss-149358
Name		DSV-143598-Supplier Response-18-06-19 09:57:49
Object Type		Snapshot
Revision		0.1 Last edited on Tue Jun 18 09:57:49 EDT 2019
Approved		✗ Never Been Approved
Translation		Not Translated
Path		Classification 1 root/Snapshots/DSV-143598-Supplier Response-18-06-19 09:57:49
Visibility		
Snapshot Data	abc	
Snapshot Date Time	2019-06-18 09:57:49	
Snapshot Event ID	Supplier Response	
Snapshot Node ID	abc	DSV-143598

- A 'SnapshotConfiguration' reference is created from the snapshot classification to the desired Snapshot Configuration.

Tree	
Snapshot Group	
Recipe Specification	
Attribute Values	
Other Information	
Recipe Parameters	
Snapshots	
DSV-143598-Supplier Response-18-06-19 09:57:49	

DSV-143598-Supplier Response-18-06-19 09:57:49		
Referenced By	Images & Documents	Tables
Status	State Log	Tasks
Classification		Sub Products
		References
Ungrouped Classification References		
Reference Type	>	Target
Snapshot Configuration	+	Recipe Specification

- Existing snapshot with the same Event ID and for the current Node ID will be deleted. Only one snapshot is stored for an Event ID / Node ID combination.
- Snapshot data will be captured in the 'SnapshotData' attribute on the snapshot.

DSV-143598-Supplier Response-18-06-19 09:57:49 rev.0.1 - Classification		
Referenced By	Images & Documents	Tables
Status	State Log	Tasks
Classification	Sub Products	References
Description		
Name	>	Value
ID		ss-149358
Name		DSV-143598-Supplier Response-18-06-19 09:57:49
Object Type		Snapshot
Revision		0.1 Last edited Tue Jun 18 09:57:49 EDT 2019
Approved		✖ Never Been Approved
Translation		Not Translated
Path		Classification 1 root/Snapshots/DSV-143598-Supplier Response-18-06-19 09:57:49
Visibility		
Snapshot Data	abc	{ "contexts": { "Context2": { "recorders": { "ssrec-149312": { "nodeID": "DSV-143598", "nodeName": "Dark Chocolate Chip", "objectTypeID": "PLMDesignSpecificationVariant", "objectTypeName": "Design" } } } } }
Snapshot Date Time	2019-06-18 09:57:49	
Snapshot Event ID	Supplier Response	
Snapshot Node ID	abc	DSV-143598

8. The 'createSnapshot' method will return a JSON status message of either:

- **Successful Snapshot:** {"status":0;"msg": "Snapshot created successfully"}

A status field marked with a 0 (zero) indicates that the snapshot was created successfully, and the 'msg' field will have the success message.

- **Snapshot with Errors:** {"status": 1; "msg": "An error message"}

A status field marked with a 1 (one) signals that an error occurred while the snapshot was being created. The 'msg' field will contain the error message. No snapshot was created.

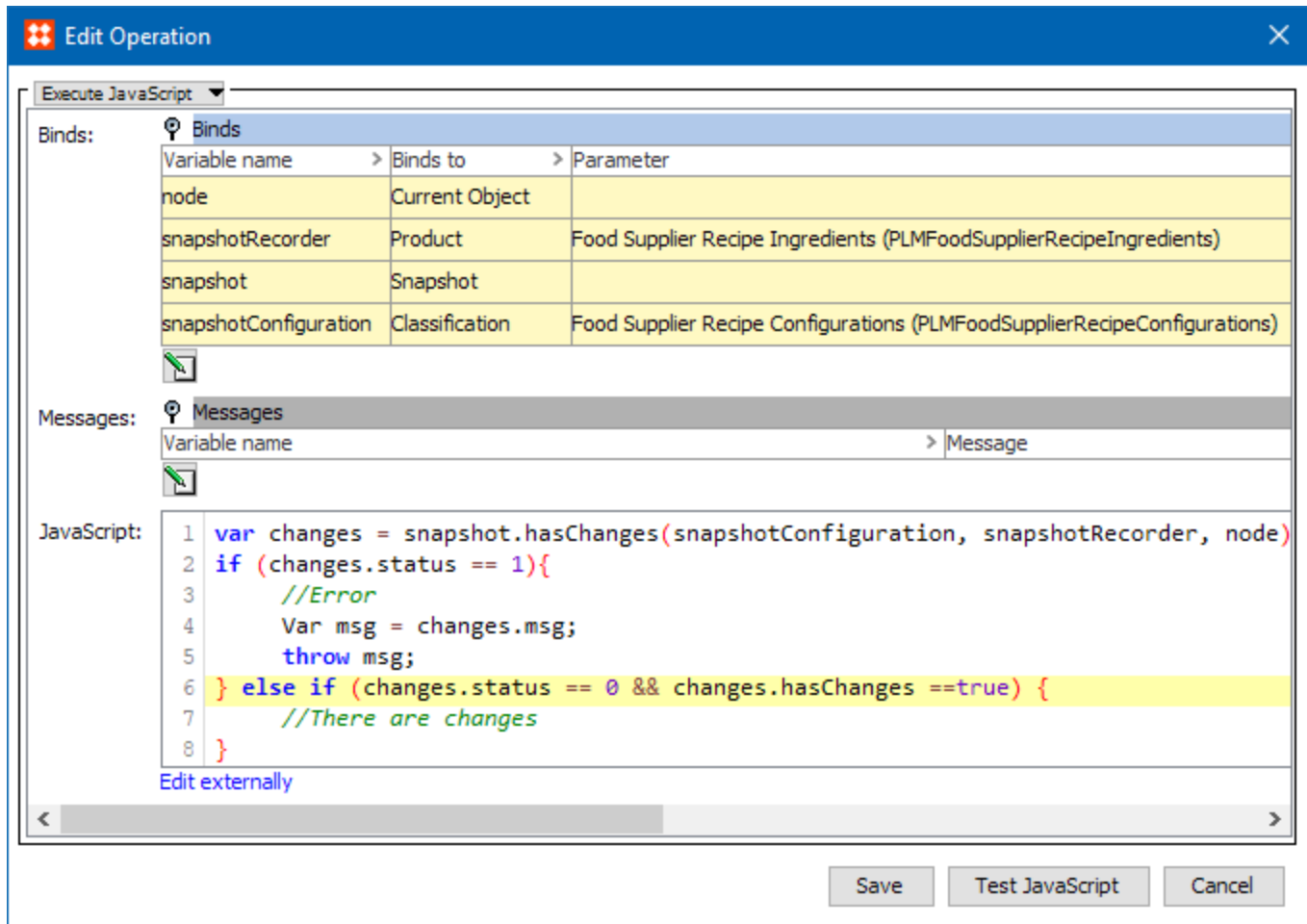
Flagging Changes

This public method flags if there are any changes based on what the Snapshot Recorder has defined. Using the defined Event ID for the specified node, it compares the current values with the values of the last snapshot.

Note: If there are no previous snapshots, then there are no changes.

The name of the method is 'hasChanges,' and it has three arguments:

- A classification bind to the Snapshot Configuration
- A product bind to a Snapshot Recorder node
- An instance of the node where flag changes are executed



Note: The Event ID is defined in the Snapshot Configuration instance and from the specified node in which there are references to snapshots.

The following actions take place when this is being executed to see if there are any Flag Changes:

1. A check is made to the specified Snapshot Recorder to see if it is linked to the selected Snapshot Configuration. If it is not, an error occurs.
2. The Event ID is retrieved from the Snapshot Configuration instance.
3. From the node, all the snapshots being referenced are:
 - filtered by Event ID
 - retrieved for comparison
4. The snapshot is then compared with the current values on the node, with only the values specified in the Snapshot Recorder being compared. The following outcomes could occur and return the corresponding JSON strings:
 - **No Changes:** {"status": 0, "msg": "No Changes", "hasChanges": false}
The 0 (zero) indicates that the method was executed successfully, the msg has a 'No changes' message, and the 'hasChanges' result is false, indicating that there were no changes.
 - **Has Changes:** {"status": 1, "msg": "Has changes", "hasChanges": true}

The 1 (one) indicates that the method was executed successfully, the msg has a 'Has changes' message, and the 'hasChanges' result is true, indicating that there were changes.

- **Node does not have any snapshots:** {"status": 0, "msg": "No snapshots exists", "hasChanges: false}

This JSON string returns when the node being monitored does not have a snapshot for the Event ID.

- **Errors:** {"status": -1, "msg": "Description of the error"}

A status field marked with a -1 (minus one) signals that an error occurred while executing the hasChanges method. The 'msg' field will contain the error message.

Elements in a Snapshot

Below is a list of all data scenarios that can be monitored via a snapshot:

STEP Element	Example of Recorder
STEP Object ID	The STEP object ID is stored but not shown in report. It is not part of the recorder.
STEP Object Name	"name": "true"
STEP Object Type ID	The Object Type ID is stored but not shown in report. It is not part of the recorder.
STEP Object Type Name	The Object Type Name is stored but not shown in report. It is not part of the recorder.
STEP Reference Type ID	The Reference Type Name is stored but not shown in report. It is not part of the recorder.
STEP Reference Type Name	"referenceTypeName": "true"
STEP Name of the attribute	The Attribute Name is stored but not shown in report. It is not part of the recorder.
Attribute value on a node, plus its language Dimension Dependency	{ "attributes": ["PLMDesignSpecificationDescription", "PLMSpecificationName"] }
Attribute value on a Reference, plus its language dimension dependency	{ "references": { "PLMParameter": { "attributes": ["PLMMethod", "PLMSuppliersExplanation"] } } }
Attribute value on a Referenced By plus its language dimension dependency	{ "referencedBy": { "PLMRelatedRecipe": { "objectTypeName": "true", "source": { "name": "true", "attributes": ["PLMSpecificationName"] } } } }
Attribute value on a parent plus its language dimension dependency	{ "parent": { "attributes": ["PLMProductName"] } }
Attribute value on a Classification Link	{ "classificationLinks": { "SnapshotRecorder": { "attributes": ["SnapshotDisplayName", "SnapshotRecorderDisplay", "SnapshotTabOrder"] } } }

STEP Element	Example of Recorder
Attribute value on a referenced target node plus its language dimension dependency	{ "classificationLinks": { "PLMProductCategory": { "target": { "attributes": ["PLMCategoryName", "PLMGlobalProductClassificationCode"] } } } }
Attribute value with units of measure plus its language dimension dependency	{ "attributes": [" PLMPlannedPackagingSizes"] }
Multi Valued Attributes plus its language dimension dependency	{ "attributes": [" PLMPlannedFlavorVariations"] }
Attributes with LOV ID's plus its language dimension dependency	LOV IDs are not captured. Only the values are captured.
Be able to follow a reference from a target node, for example: Capture data points on a node any number of nested levels away from the current node by following a reference (or reverse reference).	{ "referencedBy": { "PLMRelatedRecipe": { "objectTypeName": "true", "source": { "name": "true", "attributes": ["PLMSpecificationName"], "referencedBy": { "PLMLabelVariantSample": { "source": { "attributes": ["PLMSpecificationName"] } } } } } } }

Private Label Food Solution Setup in Workbench

There are a number of object types, attributes, and reference types that need to be created and properly set up in workbench in order for the private label food solution to function properly in Web UI. For more information, see the following sections below on how to properly set up the private label food solution object types, reference types, and attributes in workbench.

- Creating Private Label Food Solution Object Types
- References Needed for Compare Tabs
- Attributes Needed for Compare Tabs

Note: Suppliers can get a record of their specification response that they submit to the customer when they respond to a bid for a private label product. If chosen for production, the supplier's response is the basis for contractual agreements. This record outlines the bill of materials used to fulfill the private label product, and provides explanations on how the supplier will meet the customer's requirements. For more information see the Private Label Food Solution segment of the Product Lifecycle Management section in the Solution Enablement documentation.

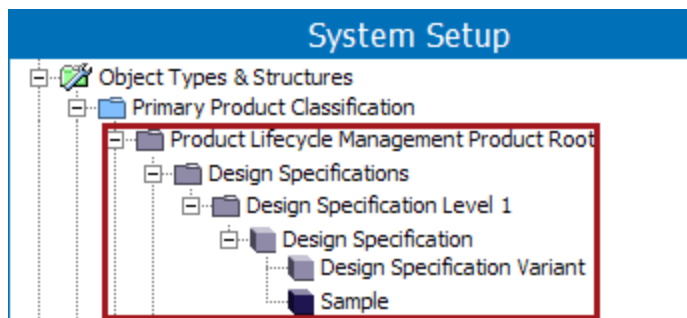
Creating Private Label Food Solution Object Types

The following sections describe what Primary Product Classifications and Alternative Classifications object types are needed for the private label food solution to function properly in Web UI.

Creating Primary Product Classification Object Types

Design Specification Variants and Samples are in the Primary Product Classification object type structures and have the same Design Specification parent object type.

In System Setup, there should be a structure created like the one pictured below. The created object types should use the following IDs and names in the table below.

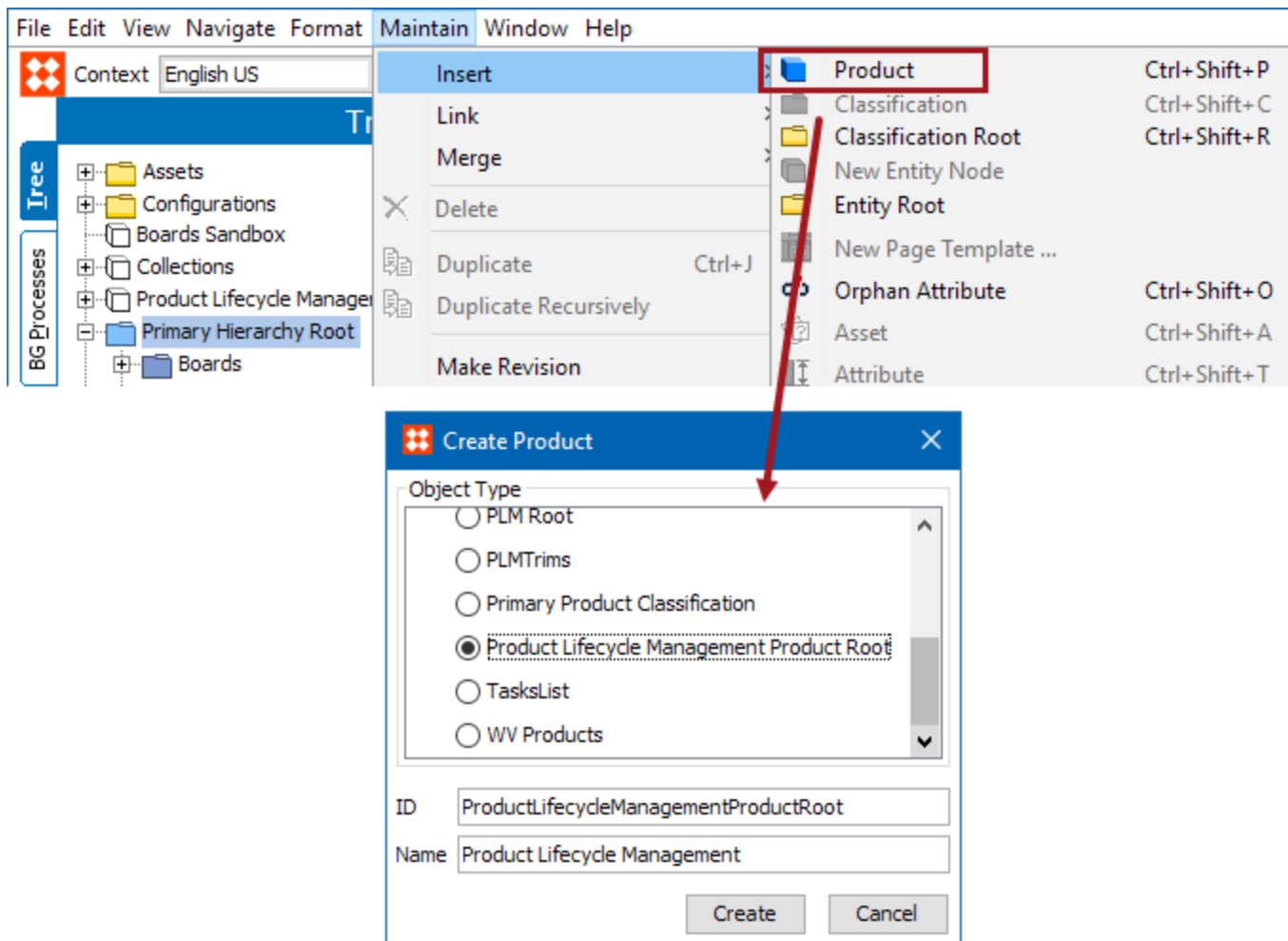


- **Design Specification:** Represents the entire project and is a high-level place to gather input for new products.
- **Design Specification Packaging Variant:** Represents a package size variation for one or more new products. All description and requirements are collected on this object.
- **Design Specification Variant:** Represents a customer's recipe variations for one or more products.
- **Sample:** The supplier's response that is provide back to the customer.

ID	Name
ProductLifecycleManagementProductRoot	Product Lifecycle Management Product Root
PLMDesignSpecificationsRoot	Design Specifications
PLMDesignSpecificationLevel1	Design Specification Level 1
PLMDesignSpecification	Design Specification
PLMDesignSpecificationPackagingVariant	Design Specification Packaging Variant

ID	Name
PLMDesignSpecificationVariant	Design Specification Variant
PLMSample	Sample

After the structure has been created in System Setup, it needs to be instantiated in Tree. In Tree, click on a Primary Product Classification folder and, go to Maintain > Insert > Product > select the object type to hold the **Design Specification Variants** and **Samples**.

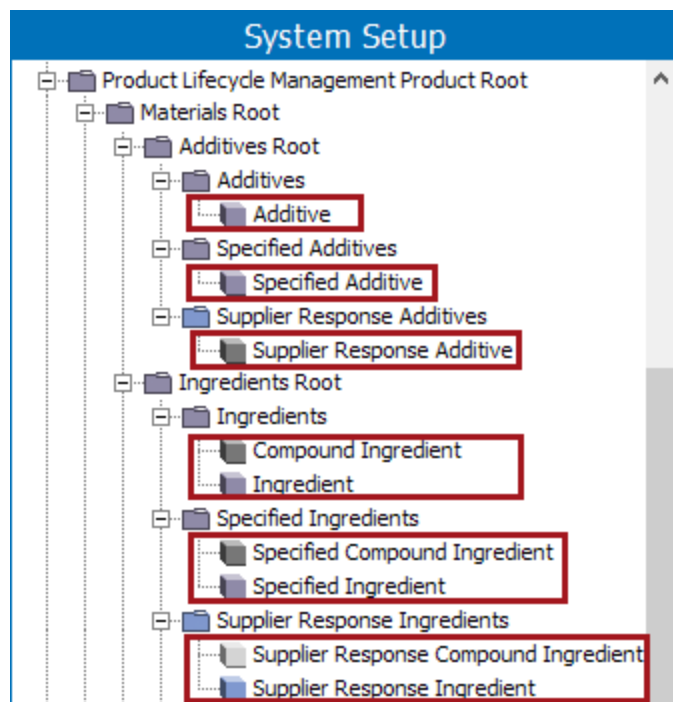


Once the hierarchy structure for the PLMDesignSpecificationVariant and PLMSample has been created and added to Tree, references need to be created between design specification variants and sample object types. See the **References Needed for Private Label Food Solution** topic in this documentation.

In Tree, when a recipe specification is created, it will be a **Design Specification Variant** object type. The Design Specification Variant object type holds the requirements for the recipe specification being created. Each supplier recipe option is created as a **Sample** object type.

Additional Primary Product Hierarchy Object Types

When working with design specification variants, there are a number of additional object types needed when creating recipes. They allow for companies to communicate with their suppliers about ingredients, compound ingredients, additives, and packaging, and provides a way for suppliers to communicate back with the companies.



- **Additive:** Represents the library of substances that are added to recipes to improve or preserve.
- **Specified Additive:** Copy of an additive that is used in a specification that provides instruction about the additive.
- **Supplier Response Additive:** Represents the supplier's response to the substances that are added to improve or preserve the specification recipe.
- **Compound Ingredient:** An ingredient used in the recipe that has sub-ingredients that is part of the ingredient library. The developed component will copy this as specified ingredient or supplier response ingredient.
- **Ingredient:** Foods or substances used in a recipe that is part of the library. The developed component will copy this as specified ingredient or supplier response ingredient.
- **Specified Compound Ingredient:** A copy of a compound ingredient that has sub-ingredients used in a specification that provides instruction about the compound ingredient.
- **Specified Ingredient:** A copy of an ingredient, such as a foods or substance, that is used in a specification recipe that provides instruction about the ingredient.
- **Supplier Response Compound Ingredient:** An ingredient that has sub-ingredients used in a recipe provided by the supplier.
- **Supplier Response Ingredient:** Foods or substances used in a recipe provided by the supplier.

The following IDs and names need to be used when creating these object types:

Important: The following IDs need auto IDs entered into the ID Pattern field:

- PLMSpecifiedAdditive
- PLMSpecifiedIngredient
- PLMSpecifiedCompoundIngredient
- PLMSupplierResponseAdditive
- PLMSupplierResponseIngredient
- PLMSupplierResponseCompoundIngredient

Auto IDs are needed when adding ingredients to the Design Specification Variant and Sample object types. For more on how to add auto IDs see the **Autogenerate Using Name Pattern and ID Pattern** topic in the **Object Types and Structures** section of the **System Setup / Super User Guide** documentation.

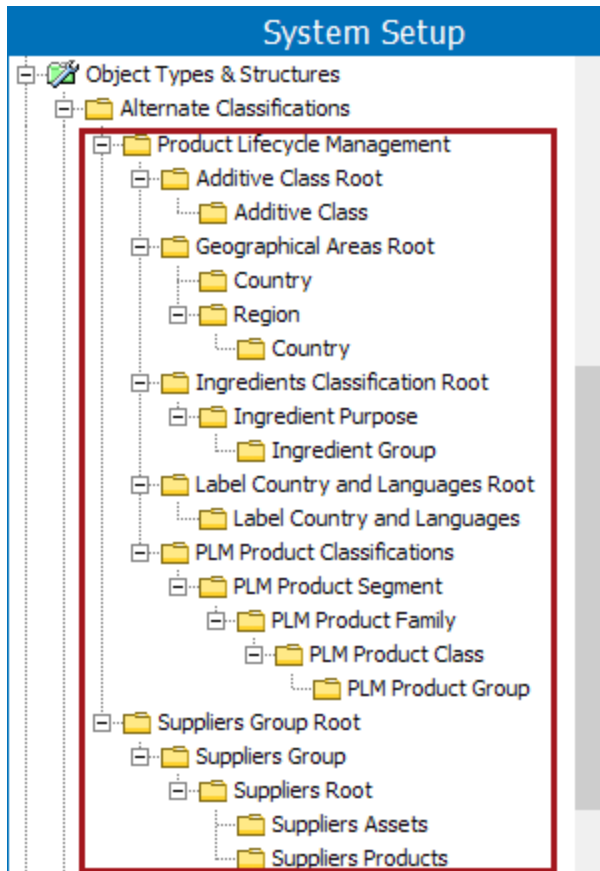
ID	Name
PLMMaterialsRoot	Materials Root
PLMAdditivesRoot	Additives Root
PLMAdditives	Additives
PLMAdditive	Additive
PLMSpecifiedAdditives	Specified Additives
PLMSpecifiedAdditive	Specified Additive
PLMSupplierResponseAdditives	Supplier Response Additives
PLMSupplierResponseAdditive	Supplier Response Additive
PLMIngredientsRoot	Ingredients Root
PLMIngredients	Ingredients
PLMCompoundIngredient	Compound Ingredient
PLMIngredient	Ingredient

ID	Name
PLMSpecifiedIngredients	Specified Ingredients
PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
PLMSpecifiedIngredient	Specified Ingredient
PLMSupplierResponseIngredientRoot	Supplier Response Ingredients
PLMSupplierResponseCompoundIngredient	Supplier Response Compound Ingredient
PLMSupplierResponseIngredient	Supplier Response Ingredient

Creating Alternative Classification Object Types

Alternative Classifications are where the private label food solution object types are categorized. These categorizations enable recipes to be created for multiple regions or product categories, and through business rules, automatically acquire any region, category, parameter, or requirement specifications needed. The Product Lifecycle Management Classification root folder will need to be created for all private label food solution root object types to live under.

When finished, the private label food solution classification structure will look similar to the one pictured below.



- **Additive Class:** A way of indicating what the additives are used for in a food product. It is used by developed components to capture the purpose for using the additive.
- **Region:** A broad geographic area distinguished by similar features.
- **Country:** Indicates which countries a product is planned for sale. Also used to classify requirements and parameters that should be applied to products through business rules that will be sold in that country.
- **Ingredient Purpose:** A number of ingredients that are all related to one another in some way. It also controls where ingredients are searched from.
- **Ingredient Group:** A number of ingredients that are all related to one another in some way.
- **Ingredient Specification Group:** A grouping of ingredients and compound ingredients that are linked into one or more classifications for organizational purposes.
- **Label Country and Languages:** Different labels with the various languages the labels support.
- **PLM Product Group:** A product classification that holds default requirements and parameters that can be applied via business rules when ingredients are added to recipes.
- **Suppliers Products:** All recipes samples, responses, ingredients, additives, compound ingredients, etc. that are created by a particular supplier.

ID	Name
ProductLifecycleManagementClassifRoot	Product Lifecycle Management
PLMAdditiveClassRoot	Additive Class Root
PLMAdditiveClass	Additive Class
PLMGeographicalAreasRoot	Geographical Areas Root
PLMCountry	Country
PLMRegion	Region
PLMIngredientsClassificationRoot	Ingredients Classification Root
PLMIngredientPurpose	Ingredient Purpose
PLMIngredientGroup	Ingredient Group
PLMIngredientSpecificationGroups	Ingredient Specification Groups
PLMIngredientSpecificationGroup	Ingredient Specification Group
PLMLabelCountryAndLanguagesRoot	Label Country and Languages Root
PLMLabelCountryAndLanguages	Label Country and Languages
PLMProductClassificationsRoot	PLM Product Classifications
PLMProductSegment	PLM Product Segment
PLMProductFamily	PLM Product Family
PLMProductClass	PLM Product Class
PLMProductGroup	PLM Product Group
SupplierGroupRoot	Suppliers Group Root

ID	Name
SuppliersGroup	Suppliers Group
SuppliersRoot	Suppliers Root
SuppliersAssets	Suppliers Assets
SuppliersProducts	Suppliers Products

References Needed for Private Label Food Solution

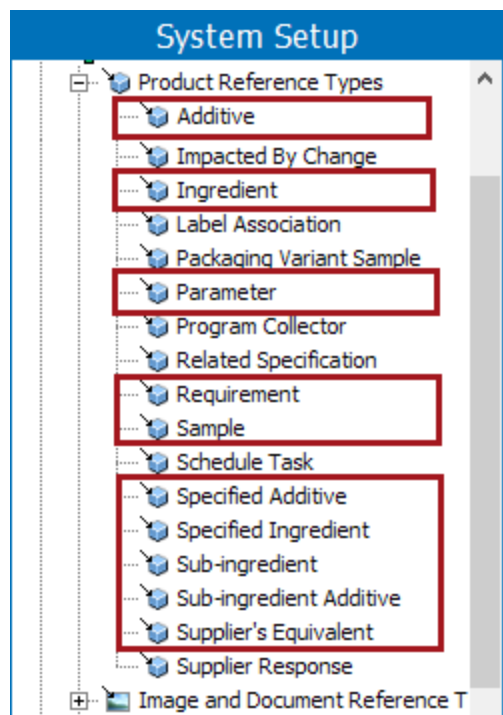
The following reference types are required for the private label food solution to work properly:

- Product Reference Types
- Classification Reference Types
- Product to Classification Link Types

See the above mentioned topics for more information on how to create the necessary reference types.

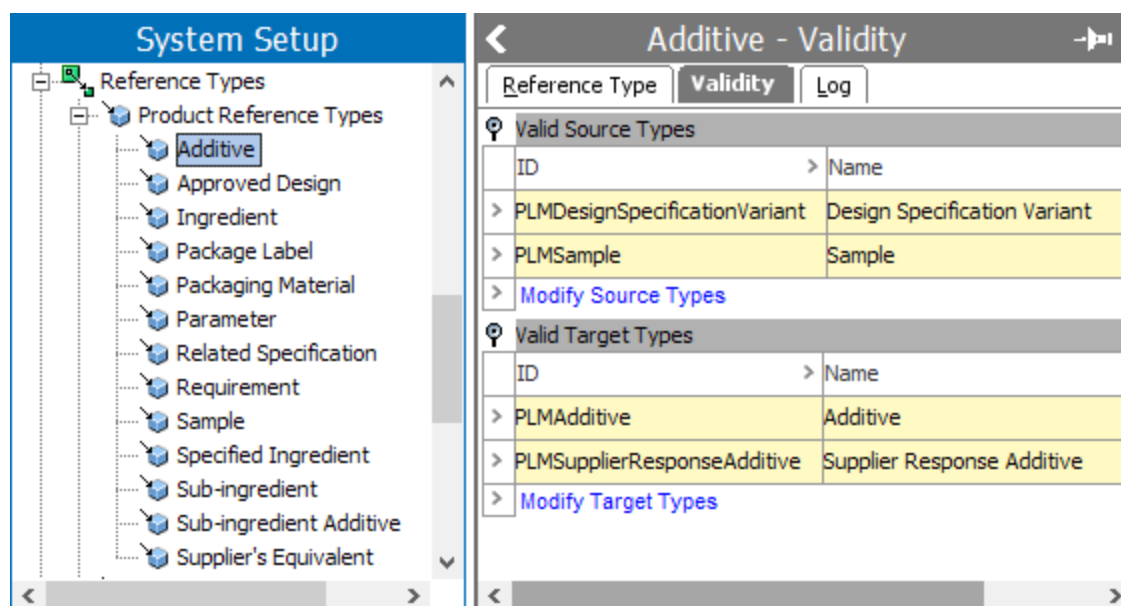
Product Reference Types

There are a number of Product Reference Types needed for private label tabs to work. The references needed are pictured below and described in further detail in this section.



Additive

PLMAdditive is used as a reference to connect a PLMAdditive to a PLMDesignSpecificationVariant object type or a PLMSupplierResponseAdditive to a PLMSample object type. The PLMAdditive reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:



Note: While PLMAdditive is mandatory, it is possible to have more than one additive reference type. However, each additional additive reference type still must have the same attributes as PLMAdditive.

Ingredient

PLMIngredient is used as a reference to connect a PLMDesignSpecificationVariant to a PLMSpecifiedCompoundIngredient or a PLMSpecifiedIngredient. Additionally, it is used as a reference between a PLMSample and a PLMSupplierResponseCompoundIngredient or a PLMSupplierResponseIngredient.

The PLMIngredient reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

System Setup

- Reference Types
 - Product Reference Types
 - Additive
 - Approved Design
 - Ingredient**
 - Package Label
 - Packaging Material
 - Parameter
 - Related Specification
 - Requirement
 - Sample
 - Specified Ingredient
 - Sub-ingredient
 - Sub-ingredient Additive
 - Supplier's Equivalent
 - Image and Document Reference
 - Classification Reference Types
 - Product to Classification Link Type
 - Product Attribute Link Type

Ingredient - Validity

Reference Type | Validity | Log

Valid Source Types

ID	Name
> PLMDesignSpecificationVariant	Design Specification Variant
> PLMSample	Sample
Modify Source Types	

Valid Target Types

ID	Name
> PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
> PLMSpecifiedIngredient	Specified Ingredient
> PLMSupplierResponseCompoundIngredient	Supplier Response Compound Ingredient
> PLMSupplierResponseIngredient	Supplier Response Ingredient
Modify Target Types	

Note: While PLMIngredient is mandatory, it is possible to have more than one ingredient reference type. However, each additional ingredient reference type still must have the same attributes as PLMIngredient.

Parameter

PLMParameter is used as a reference to connect the following object types to a PLMParameter object type:

- PLMCompoundIngredient
- PLMDesignSpecificationVariant
- PLMIngredient
- PLMSample
- PLMSpecifiedCompoundIngredient
- PLMSpecifiedIngredient
- PLMSupplierResponseCompoundIngredient
- PLMSupplierResponseIngredient

The PLMParameter reference should have the following Valid Source Types and Valid Target Types as seen in the picture below.

System Setup

- Reference Types
 - Product Reference Types
 - Additive
 - Approved Design
 - Ingredient
 - Package Label
 - Packaging Material
 - Parameter
 - Related Specification
 - Requirement
 - Sample
 - Specified Ingredient
 - Sub-ingredient
 - Sub-ingredient Additive
 - Supplier's Equivalent
 - Image and Document Reference
 - Classification Reference Types
 - Product to Classification Link Type
 - Product Attribute Link Type
 - Classification Attribute Link Type

Parameter - Validity

Reference Type
Validity
Log

Valid Source Types

ID	Name
> PLMCompoundIngredient	Compound Ingredient
> PLMDesignSpecificationVariant	Design Specification Variant
> PLMIngredient	Ingredient
> PLMSample	Sample
> PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
> PLMSpecifiedIngredient	Specified Ingredient
> PLMSupplierResponseCompoundIngredient	Supplier Response Compound Ingredient
> PLMSupplierResponseIngredient	Supplier Response Ingredient
> Modify Source Types	

Valid Target Types

ID	Name
> PLMParameter	Parameter
> Modify Target Types	

Note: It is possible to have more than one parameter reference type. However, each additional parameter reference type still must have the same source, target, and attributes as the original PLMParameter, though it could have additional source, target, and attribute types as well.

Requirement

PLMRequirement is used as a reference to connect the following object types to a PLMRequirement object type:

- PLMDesignSpecificationVariant
- PLMSample

The PLMRequirement reference should have the following Valid Source Types and Valid Target Types as seen in the picture below.

System Setup

- Reference Types
 - Product Reference Types
 - Additive
 - Approved Design
 - Ingredient
 - Package Label
 - Packaging Material
 - Parameter
 - Related Specification
 - Requirement**
 - Sample
 - Specified Ingredient
 - Sub-ingredient
 - Sub-ingredient Additive
 - Supplier's Equivalent
 - Image and Document Reference Types
 - Classification Reference Types
 - Product to Classification Link Types
 - Product Attribute Link Type

Requirement - Validity

Reference Type
Validity
Log

Valid Source Types

ID	Name
> PLMApprovedDesign	Approved Design
> PLMDesignSpecificationLabelVariant	Design Specification Label Variant
> PLMDesignSpecificationVariant	Design Specification Variant
> PLMPackageLabel	Package Label
> PLMPackagingElement	Packaging Element
> PLMSample	Sample
> PLMSupplierResponsePackagingElement	Supplier Response Packaging Element

[Modify Source Types](#)

Valid Target Types

ID	Name
> PLMRequirement	Requirement

[Modify Target Types](#)

Note: It is possible to have more than one requirement reference type. However, each additional requirement reference type still must have the same source, target, and attributes as the original PLMRequirement, though it could have additional source, target, and attribute types as well.

Sample

PLMSample is used as a reference from the PLMDesignSpecificationVariant object types to the PLMSample object types.

Tree

- Product Hierarchy
- Product Lifecycle Management
- Design Specifications
 - Food
 - Chocolate Chip Cookies
 - Chocolate Chip Cookies (Recipe)**
 - Chocolate Chip Cookies (Recipe) Sample (supplier A)
 - Chocolate Chip Cookies (Recipe) Sample (supplier B)
 - Chocolate Chip Cookies (Recipe) Sample (supplier C)
 - Chocolate Chip Cookies (Recipe) Sample (supplier D)
 - Chocolate Chip Cookies (Recipe) Sample (supplier E)

Chocolate Chip Cookies (Recipe) rev.0.3...

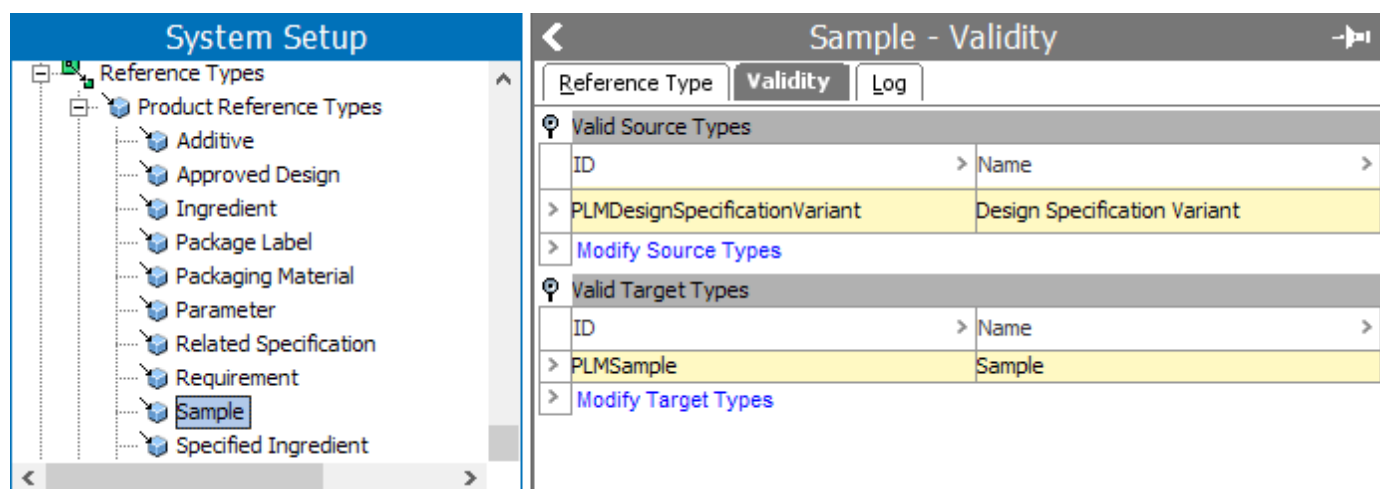
Commercial
Tables
Proof View
Status
State Log
Tasks

Product
References
Referenced By
Images & Documents

Sample Information

Reference Type	Target
> Sample	Chocolate Chip Cookies (Recipe)
> Sample	Chocolate Chip Cookies (Recipe)
> Sample	Chocolate Chip Cookies (Recipe)
> Sample	Chocolate Chip Cookies (Recipe)
> Sample	Chocolate Chip Cookies (Recipe)
> Sample	Chocolate Chip Cookies (Recipe)

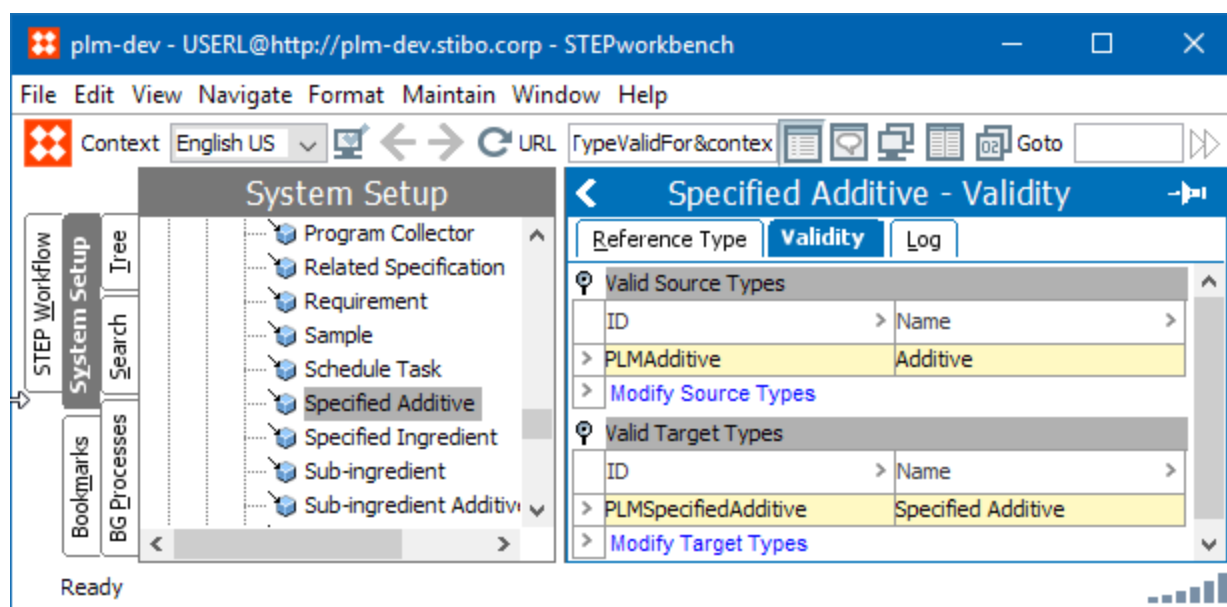
The PLMSample reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:



Specified Additive

PLMSpecifiedAdditive is used to connect a PLMAdditive to a PLMSpecifiedAdditive.

The PLMSpecifiedAdditive reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:



Specified Ingredient

PLMSpecifiedIngredient is used to connect a PLMIngredient to a PLMSpecifiedIngredient, and to connect a PLMCompoundIngredient to a PLMSpecifiedCompoundIngredient.

The PLMSpecifiedIngredient reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

The screenshot shows the 'System Setup' window on the left with a tree view of 'Product Reference Types'. 'Specified Ingredient' is selected. The main window on the right is titled 'Specified Ingredient - Validity' and contains two tables: 'Valid Source Types' and 'Valid Target Types'.

ID	Name
PLMCompoundIngredient	Compound Ingredient
PLMIngredient	Ingredient

ID	Name
PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
PLMSpecifiedIngredient	Specified Ingredient

Sub-Ingredient

PLMSubIngredient reference is used to connect PLMSpecifiedCompoundIngredient to PLMSpecifiedIngredient and PLMSupplierResponseCompoundIngredient to PLMSupplierResponseIngredient.

The PLMSubIngredient reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

The screenshot shows the 'System Setup' window on the left with 'Sub-ingredient' selected. The main window on the right is titled 'Sub-ingredient - Validity' and contains two tables: 'Valid Source Types' and 'Valid Target Types'.

ID	Name
PLMCompoundIngredient	Compound Ingredient
PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
PLMSupplierResponseCompoundIngre...	Supplier Response Compound Ingredient

ID	Name
PLMCompoundIngredient	Compound Ingredient
PLMIngredient	Ingredient
PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
PLMSpecifiedIngredient	Specified Ingredient
PLMSupplierResponseCompoundIngre...	Supplier Response Compound Ingredient
PLMSupplierResponseIngredient	Supplier Response Ingredient

Sub-Ingredient Additive

PLMSubIngredientAdditive reference is used to connect PLMSpecifiedCompoundIngredient to PLMSpecifiedAdditive and PLMSupplierResponseCompoundIngredient to PLMSupplierResponseAdditive.

The PLMSubIngredientAdditive reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

The screenshot shows two side-by-side windows from the Stibo Systems application. The left window, titled 'System Setup', displays a hierarchical tree of system components. The 'Sub-ingredient Additive' item is selected and highlighted. The right window, titled 'Sub-ingredient Additive - Validity', shows the configuration for this reference type. It has tabs for 'Reference Type', 'Validity', and 'Log', with 'Validity' currently active. The 'Validity' tab contains two sections: 'Valid Source Types' and 'Valid Target Types'. Each section has a table with columns for 'ID' and 'Name'. In the 'Valid Source Types' table, two entries are listed: 'PLMSpecifiedCompoundIngredient' (Specified Compound Ingredient) and 'PLMSupplierResponseCompoundIngredient' (Supplier Response Compound Ingredient). In the 'Valid Target Types' table, two entries are listed: 'PLMSpecifiedAdditive' (Specified Additive) and 'PLMSupplierResponseAdditive' (Supplier Response Additive). Both tables have a 'Modify' link below them.

ID	Name
PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
PLMSupplierResponseCompoundIngredient	Supplier Response Compound Ingredient

ID	Name
PLMSpecifiedAdditive	Specified Additive
PLMSupplierResponseAdditive	Supplier Response Additive

Supplier's Equivalent

The PLMSuppliersEquivalent references is used to connect PLMCompoundIngredient to PLMSupplierResponseCompoundIngredient, PLMIngredient to PLMSupplierResponseIngredient, and PLMAdditive to PLMSupplierResponseAdditive.

The PLMSuppliersEquivalent reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

System Setup

- Product Reference Types
 - Additive
 - Approved Design
 - Ingredient
 - Package Label
 - Packaging Material
 - Parameter
 - Related Specification
 - Requirement
 - Sample
 - Specified Ingredient
 - Sub-ingredient
 - Sub-ingredient Additive
 - Supplier's Equivalent**
- Image and Document Reference
- Classification Reference Types
- Product to Classification Link Type
- Product Attribute Link Type

Supplier's Equivalent - Validity

Reference Type
Validity
Log

Valid Source Types

ID	Name
PLMAdditive	Additive
PLMCompoundIngredient	Compound Ingredient
PLMIngredient	Ingredient
Modify Source Types	

Valid Target Types

ID	Name
PLMSupplierResponseAdditive	Supplier Response Additive
PLMSupplierResponseCompoundIngredient	Supplier Response Compound Ingredient
PLMSupplierResponseIngredient	Supplier Response Ingredient
Modify Target Types	

Classification Reference Types

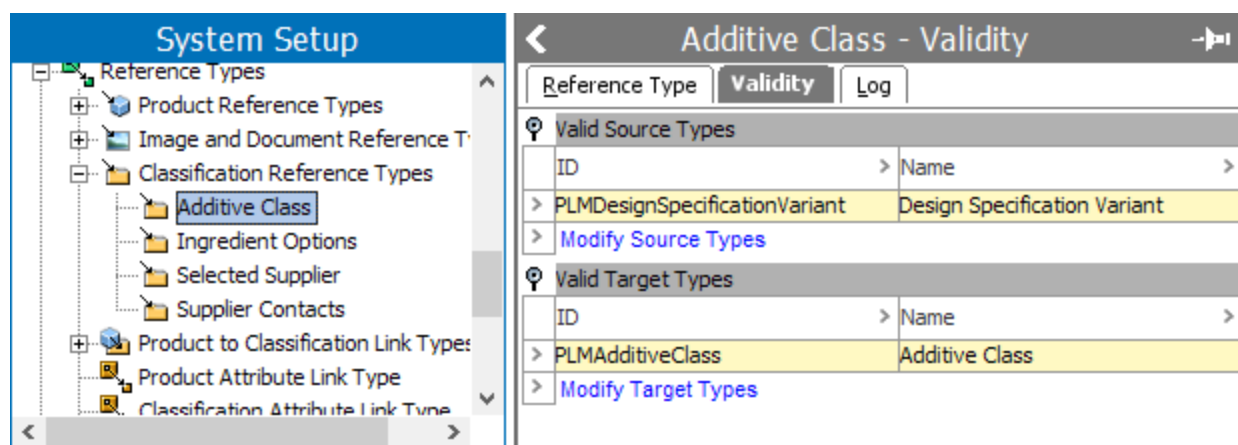
The Product to Classification Link Type is used to indicate which suppliers are bidding on the new product.

Additive Class

The classification reference can be used by customers to specify criteria about an entire group of additives rather than individually specify all of them. This allows the supplier to see what is in the group so that they can meet the specification.

The PLMAdditiveClass reference is used to connect the product PLMDesignSpecificationVariant object type to the classification PLMAdditiveClass object type.

The reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

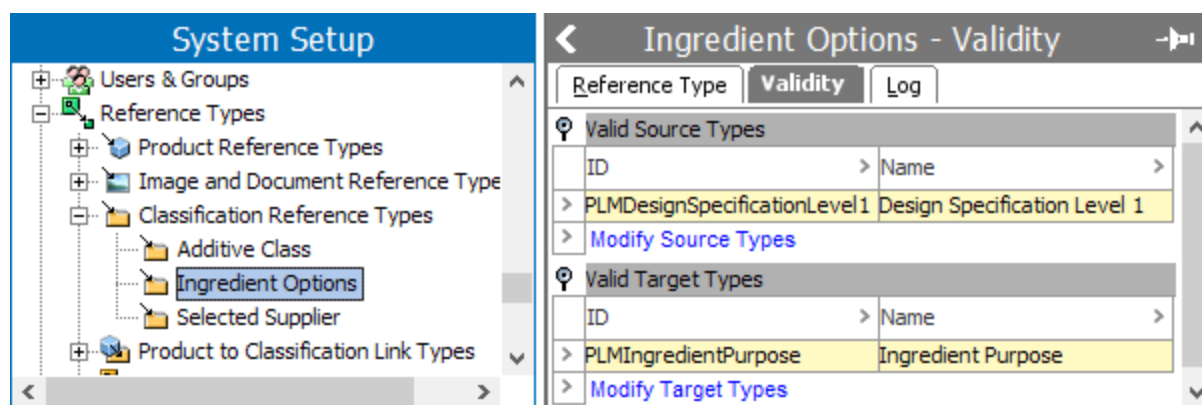


Ingredient Options

The ingredient Options reference allows customers to classify ingredients for different purposes, such as food vs pet food, and then limits where a user can choose ingredients to specify, or limits what supplier can choose to build recipes. Developed component looks for the parent of the design specification and connects all design specs in that category to the right ingredient classification.

The PLMIngredientOptions reference is used to connect the product PLMDesignSpecificationLevel1 object types to the classification PLMIngredientPurpose object type.

The reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

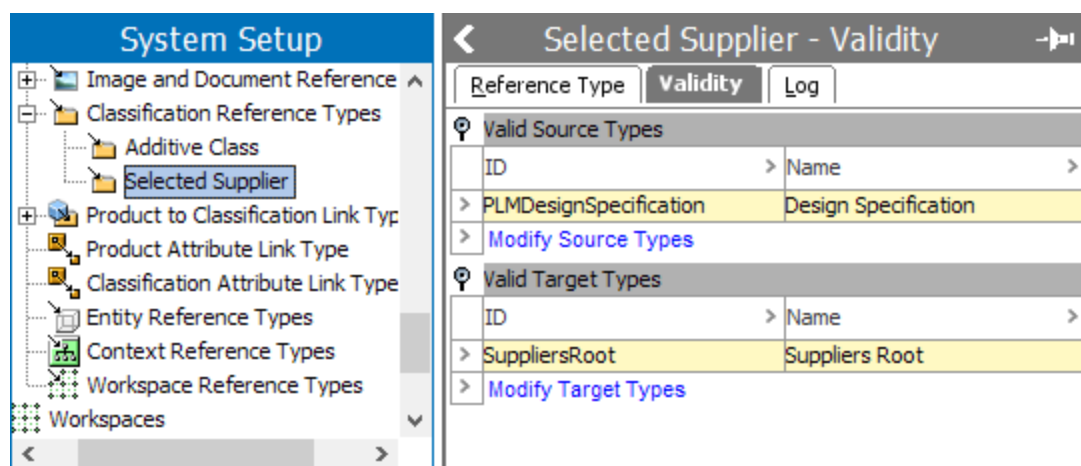


Selected Supplier

Note: The following reference type is not used in the developed component, but is used in the private label food solution.

The PLMSelectedSupplier reference is used to connect the product PLMDesignSpecification object types to the classification SupplierRoot object type.

The reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:



Ingredient Specification Group

The classification reference can be used by customers to specify criteria about an entire group of ingredients rather than individually specify all of them. This allows the supplier to see what is in the group so that they can meet the specification.

The PLMIngredientSpecificationGroup reference is used to connect the product PLMDesignSpecificationVariant object type to the classification PLMIngredientSpecificationGroup.

The reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:

System Setup

- Classification Reference Types
 - (PLMAdditiveClass)
 - (PLMIngredientOptions)
 - (PLMIngredientSpecificationGroup)
 - (PLMRecipeSupplier)
 - (PLMSelectedSupplier)
 - (PLMSupplierContacts)
 - (Snapshot)
 - (SnapshotConfiguration)
- Product to Classification Link Types
- Product Attribute Link Type
- Classification Attribute Link Type
- Entity Reference Types

< (PLMIngredientSpecificationGroup) - Validity

Reference Type
Validity
Log

Valid Source Types

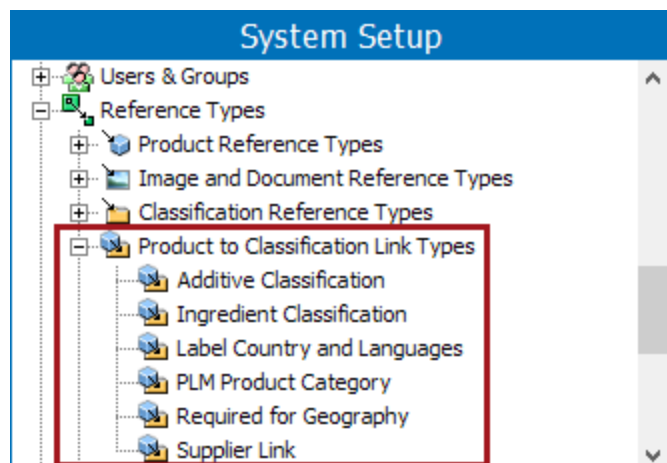
ID	Name
> PLMDesignSpecificationVariant	Design Specification Variant
> PLMSpecifiedCompoundIngredient	Specified Compound Ingredient
> Modify Source Types	

Valid Target Types

ID	Name
> PLMIngredientSpecificationGroup	Ingredient Specification Group
> Modify Target Types	

Product to Classification Link Types

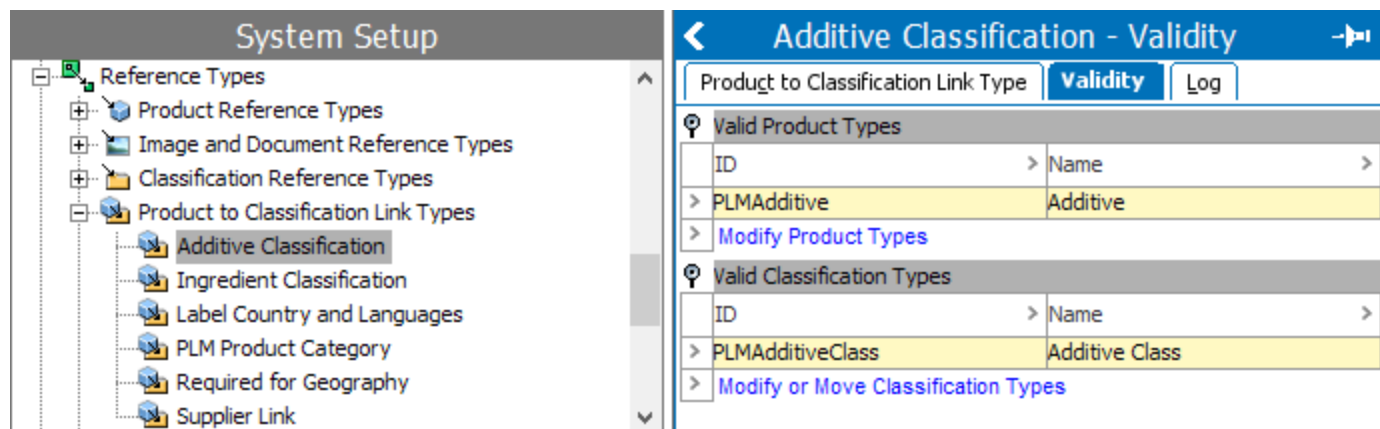
Product to classification link types are used to help classify which product object types are to be sorted into which classification group. It is possible for an object to be linked to more than one classification group.



Additive Classification

PLMAdditiveClassification is used to sort additive product types into various additive class folders. It is also possible for a PLMAdditiveClassification to be part of more than one additive class.

The PLMAdditiveClassification reference should have the following Valid Source Types and Valid Target Types as seen in the picture below:



Ingredient Classification

PLMIngredientClassification is used to control which ingredient and compound ingredient product types can be added into various ingredient classification folders, for example, food versus pet food. It is possible for PLMIngredient and PLMCompoundIngredient product types to be part of more than one ingredient class.

The PLMIngredientClassification should have the following Valid Source Types and Valid Target Types as seen in the picture below:

The screenshot shows the 'System Setup' interface on the left and the 'Ingredient Classification - Validity' configuration window on the right. The 'System Setup' tree on the left has 'Ingredient Classification' selected under 'Product to Classification Link Types'. The 'Validity' tab in the configuration window shows two sections: 'Valid Product Types' and 'Valid Classification Types'. Both sections have a table with 'ID' and 'Name' columns.

ID	Name
PLMCompoundIngredient	Compound Ingredient
PLMIngredient	Ingredient

ID	Name
PLMIngredientGroup	Ingredient Group

Label Country and Languages

Note: This reference is not part of the developed delivered components, but is used in the private label food solution.

The PLMLabelCountryAndLanguages reference is used to indicate which language bundles will be produced on a product label. It sorts PLMDesignSpecification and PLMDesignSpecificationLabelVariant product types into various PLMLabelCountryAndLanguages classes. It is possible for PLMDesignSpecification and PLMDesignSpecificationLabelVariant product types to be part of more than one label country and languages class bundle.

The PLMLabelCountryAndLanguages should have the following Valid Source Types and Valid Target Types as seen in the picture below:

The screenshot shows the 'System Setup' interface on the left and the 'Label Country and Languages - Validity' configuration window on the right. The 'System Setup' tree on the left has 'Label Country and Languages' selected under 'Product to Classification Link Types'. The 'Validity' tab in the configuration window shows two sections: 'Valid Product Types' and 'Valid Classification Types'. Both sections have a table with 'ID' and 'Name' columns.

ID	Name
PLMDesignSpecification	Design Specification
PLMDesignSpecificationLabelVariant	Design Specification Label Variant

ID	Name
PLMLabelCountryAndLanguages	Label Country and Languages

PLM Product Category

The PLMProductCategory is used to sort PLMDesignSpecification, PLMDesignSpecificationLabelVariant, PLMParameter, and PLMRequirement object types into various PLMProductCategory classes. In other words, it enables customers to associate a design specification with a product category, and then link any necessary requirements and parameters. It is possible for PLMDesignSpecification, PLMDesignSpecificationLabelVariant, PLMParameter, and PLMRequirement product types to be part of more than one label country and languages class.

The PLM Product Category product to classification link type should have the following Valid Source Types and Valid Target Types as seen in the picture below:

System Setup

- Reference Types
 - Product Reference Types
 - Image and Document Reference Types
 - Classification Reference Types
 - Product to Classification Link Types
 - Additive Classification
 - Ingredient Classification
 - Label Country and Languages
 - PLM Product Category**
 - Required for Geography
 - Supplier Link
 - Product Attribute Link Type
 - Classification Attribute Link Type
 - Entity Reference Types

PLM Product Category - Validity

Product to Classification Link Type | **Validity** | Log

Valid Product Types

ID	Name
> PLMDesignSpecification	Design Specification
> PLMParameter	Parameter
> PLMRequirement	Requirement
Modify Product Types	

Valid Classification Types

ID	Name
> PLMProductGroup	PLM Product Group
Modify or Move Classification Types	

Required for Geography

PLMRequiredForGeography is used to sort PLMDesignSpecification, PLMParameter, and PLMRequirement object types into various required for geography classes, thus indicating which countries the new product is intended to be sold. Additionally, through business rules, it is used to link in requirements and parameters that should be applied to all new products for sale in that country. It is possible for PLMDesignSpecification, PLMParameter, and PLMRequirement product types to be part of more than one required for geography class.

The PLMRequiredForGeography should have the following Valid Source Types and Valid Target Types as seen in the picture below:

The screenshot shows the 'System Setup' interface with a tree view on the left and a configuration panel on the right. The tree view has 'Required for Geography' selected. The configuration panel is titled 'Required for Geography - Validity' and contains two sections: 'Valid Product Types' and 'Valid Classification Types'.

Valid Product Types	
ID	Name
PLMDesignSpecification	Design Specification
PLMParameter	Parameter
PLMRequirement	Requirement
Modify Product Types	

Valid Classification Types	
ID	Name
PLMCountry	Country
Modify or Move Classification Types	

Supplier Link

Note: In workbench, in Users & Groups under the Web UI Setting flipper, the name of the 'Link type for vendor classification to product link' may be different than this example which uses 'SupplierLink.' Be sure to check your setting in workbench for the correct classification to product link, as this will need to be used for all components and business rules. Talk to your enablement team if changes need to be made.

SupplierLink is used to sort PLMSamples, PLMSupplierResponseAdditive, PLMSupplierResponseCompoundIngredient, PLMSupplierResponseIngredient, and PLMSupplierResponsePackagingElements into various supplier link classes.

The SupplierLink should have the following Valid Source Types and Valid Target Types as seen in the picture below:

The screenshot shows the 'System Setup' interface with a tree view on the left and a configuration panel on the right. The tree view has 'Supplier Link' selected. The configuration panel is titled 'Supplier Link - Validity' and contains two sections: 'Valid Product Types' and 'Valid Classification Types'.

Valid Product Types	
ID	Name
PLMSample	Sample
PLMSupplierResponseAdditive	Supplier Response Additive
PLMSupplierResponseCompoundIngredient	Supplier Response Compound Ingredient
PLMSupplierResponseIngredient	Supplier Response Ingredient
PLMSupplierResponsePackagingElement	Supplier Response Packaging Element
Modify Product Types	

Valid Classification Types	
ID	Name
SuppliersProducts	Suppliers Products
Modify or Move Classification Types	

Attributes Needed for Private Label Food Solution

When comparing recipes from suppliers, the table below details the required attribute groups and attributes.

All attributes listed in the table are description attributes, unless otherwise noted, with many being valid for multiple Reference Types. While the names for these attributes can vary, the IDs for the attributes needed for the private label food solution must be entered in exactly as displays in the table below:

Note: Even though an attribute may be valid for more than one reference type, this does not mean that the attribute needs to hold a value. Many references are used for multiple object types, and not all object types need all attributes to hold a value. This is because the same attributes are used on both suppliers and samples, and not all attributes need values for both of these object types at the same time.

Note: There are some attribute groups and attributes included in the table below that are used by Stibo Systems' private label food solution and workflows, but are not needed for the developed components. Attributes or attribute groups not needed for the developed components are identified by an asterisk.

Attribute Groups

Attribute Group IDs	Definition	Name	Attributes in Group
PLMCompareDetailsParameter	The attributes that appear in the show details dialog in the Compare Parameters Tab in Web UI.	Compare Details, Parameter	- PLMMeetsRequirement - PLMSuppliersExplanation - PLMDefiningAttribute - PLMMethod
PLMCompareDetailsRequirement	The attributes that appear in the show details dialog in the Compare requirements Tab in Web UI	Compare Details, Requirement	- PLMMeetsRequirement - PLMDefiningAttribute - PLMSuppliersExplanation
PLMCopiedAdditiveValues	The PLMSpecifiedAdditive or the PLMSupplierResponseAdditive are created and the attributes values from the attributes in this attribute group are copied form the generic PLMAdditive to the created objects (PLMSpecifiedAdditive/PLMSupplierResponseAdditive)	Copied Additive Values	Open for various attribute types

Attribute Group IDs	Definition	Name	Attributes in Group
PLMSpecifiedIngredientAdditionalInfo	These attributes display as additional info for the Specified Ingredients Side Panel. Additionally, they display in the Specified Ingredients Tab after the PLMSpecifiedIngredientStaticInformation attributes in the dialog for adding and editing ingredients.	Specified Ingredient Additional Information	- PLMSpecificationDetails - Open to other various attribute types
PLMSupplierIngredientAdditionalInfo	In the Compare Ingredients Tab, these attributes display as additional information for the sample ingredients. Additionally, in the Supplier Ingredients Tab, the attributes display after the PLMSampleIngredientStaticInformation attributes in the dialog for adding and editing ingredients.	Supplier Ingredient Additional Information	- PLMGuaranteedCountryOfOrigin - PLMSuppliersExplanation - Open to other various attribute types
PLMSpecifiedIngredientStaticInformation	In the Specified Ingredients Tab these attributes are the static information in the dialog for adding and editing ingredients. All the attributes are needed.	Specified Ingredient Static Information	- PLMSpecifiedIngredientAllowance - PLMSpecifiedIngredientPrecision - PLMSpecifiedIngredientQuantity
PLMSupplierIngredientStaticInformation	In the Supplier Ingredients Tab these attributes are the static information in the dialog for adding and editing ingredients. All the attributes are needed.	Supplier Ingredient Static Information	PLMSuppliersIngredientQuantity

Attributes

Attribute IDs	Definition	Validati on Base Type	Validity	Dependen cies
PLMAdditiveClassID	Used by PLM developed Web UI components for specifying a recipe and for suppliers to enter their recipe additives. After selecting an additive and class for the additive, the STEP ID is written to this attribute because that ensures uniqueness of the class. The additive class name is written to a different attribute (PLMAdditiveClass).	Text	Product Reference Types PLMAdditive	None
PLMAlternateName	An alternate name used in place of an ingredient or compound ingredient for searching, e.g., searching for an 'eggplant' will also show	Text	Primary Product Types - PLMCompoundIngredient - PLMIngredient	Language

Attribute IDs	Definition	Validation on Base Type	Validity	Dependencies
	results with 'aubergine,' the alternate name for eggplant.			
PLMEuropeanNumberingSystemNumber Note: This is a specification attribute	Number codes for substances that are permitted to be used as food additives if used within the European Union and European Free Trade Association. The recipe creation components use this as an alternate search term.	Text	Primary Product Types - PLMAdditive - PLMSupplierResponseAdditive	None
PLMGuaranteedCountryOfOrigin Note: Part of PLMSupplierIngredientAdditionalInfo attribute group	Indicates the country where the ingredient, additive, or compound ingredient is from.	List of Values	Product Reference Types - PLMAdditive - PLMIngredient - PLMSubIngredient - PLMSubIngredientAdditive	None
PLMHelpText	Directions that assist the user to answer requirements and parameters.	Text	Product Types - PLMParameter - PLMRequirement Product to Classification Link Types	Language

Attribute IDs	Definition	Validation on Base Type	Validity	Dependencies
			- PLMProductCategory - PLMRequiredForGeography Product Reference Types - PLMAdditive - PLMIngredient - PLMParameter - PLMRequirement - PLMSubIngredient - PLMSubIngredientAdditive	
PLMInternationalNumberingSystem Number* Note: This is a specification attribute	An international numbering system for food additives which provide a short designation of what may be a lengthy name. You can search using this number.	Text	Primary Product Types - PLMAdditive - PLMSupplierResponseAdditive	None
PLMMeetsRequirement Note: Part of the PLMCompareDetailsRequirement and PLMCompareDetailsParameter attribute groups.	The supplier's answer on the specified parameter / requirement.	List of Values Note: List of Values use IDs for compare logic	Product Reference Types - PLMParameter - PLMRequirement	None
PLMMethod	An explanation	Text	Product Reference	Language

Attribute IDs	Definition	Validati on Base Type	Validity	Dependen cies
Note: Part of the PLMCompareDetailsParameter attribute group.	of how the measurement was done in order to meet the request.		Types PLMParameter	
PLMParameterDescription	Describes the parameter being used, or gives further information on the viewed parameter.	Text	Primary Product Types - PLMParameter Product to Classification Link Types - PLMProductCategory - PLMRequiredForGeography Product Reference Types - PLMParameter	Language
PLMParameterType	Used to group different types of parameters.	List of Values	Primary Product Types - PLMParameter Classification Reference Types - PLMProductCategory - PLMRequiredForGeography Product Reference Type - PLMParamater	None
PLMRecipeNumber	A unique identifier for a	Text	Primary Product Types	None

Attribute IDs	Definition	Validati on Base Type	Validity	Dependen cies
Note: This is a specification attribute	supplier's recipe.		PLMSample	
PLMRequirementDescription	The explanation of a requirement.	Text	Primary Product Types - PLMRequirement Product to Classification Link Types - PLMProductCategory - PLMRequiredForGeography Product Reference Type - PLMRequirement	Language
PLMRequirementType	A grouping mechanism for different types of requirements.	List of Values	Primary Product Types - PLMRequirement Product to Classification Link Type - PLMProductCategory - Required for Geography Product Reference Type - PLMRequirement	None
PLMSortOrder	Used to specify the order that parameters and requirements should be listed.	Number	Primary Product Types - PLMParameter - PLMRequirement	None

Attribute IDs	Definition	Validation on Base Type	Validity	Dependencies
	For recipes, the supplier adds a sort order to indicate the descending order of ingredients and additives.		Product to Classification Link Type - PLMProductCategory - PLMRequiredForGeography Product Reference Types - PLMAdditive - PLMIngredient - PLMParameter - PLMRequirement - PLMSubIngredient - PLMSubIngredientAdditive	
PLMSpecificationDetails Note: Part of the PLMSpecifiedIngredientAdditionalInfo attribute group.	Provides further information, such as instructions, places of origin to exclude, brand to use, etc. on specified additives, ingredients, and sub-ingredient.	Text	Product Reference Type - PLMAdditive - PLMIngredient - PLMSubIngredient	None
PLMSpecifiedIngredientAllowance	Defines whether a recipe 'May Contain,' 'Must Contain,' or 'Must Not Contain' an ingredient,	List of Values	Product Reference Type - PLMAdditive - PLMIngredient - PLMSubIngredient - PLMSubIngredientAdditive	None

Attribute IDs	Definition	Validati on Base Type	Validity	Dependen cies
	compound ingredient, or additive. It is used together with 'Specified Ingredient Precision' and 'Specified Ingredient Quantity (%)' to generate statements like 'Must Contain Exactly 8%.'			
PLMSpecifiedIngredientPrecision	A specification of the amount of the ingredient, compound ingredient or additive you want in you recipe. It can be 'Approximately', 'Exactly', 'Maximum' or 'Minimum'. Used together with 'Specified Ingredient Allowance' and 'Specified Ingredient Quantity (%)' to generate statements like 'Must Contain Exactly 8%.'	List of Values	Product Reference Type • PLMAdditive • PLMIngredient • PLMSubIngredient • PLMSubIngredientAdditive	None

Attribute IDs	Definition	Validation on Base Type	Validity	Dependencies
PLMSpecifiedIngredientQuantity	Used together with 'Specified Ingredient Allowance' and 'Specified Ingredient Precision' to generate statements like 'Must Contain Exactly 8%.'	Number	Product Reference Type • PLMAdditive • PLMIngredient • PLMSubIngredient • PLMSubIngredientAdditive	None
PLMDefiningAttribute Note: Part of the PLMCompareDetailsRequirement and PLMCompareDetailsParameter attribute groups.	Provides an area that the supplier is able to respond if the supplier must add more detail to their answer.	LOV	Product Reference Type • PLMParameter • PLMRequirement	Language
PLMSuppliersExplanation Note: Part of the PLMSupplierIngredientAdditionalInfo, PLMCompareDetailsRequirement, and the PLMCompareDetailsParameter, attribute groups	Primarily used if the supplier wants to explain why they added another ingredient, or amount of the ingredient, other than the one specified. Additionally, it is used if the supplier wants to add something to the answer of a requirement or	Text	Product Reference Type • Additive • Ingredient • Parameter • Requirement • Sub-ingredient • Sub-ingredient Additive	Language

Attribute IDs	Definition	Validation on Base Type	Validity	Dependencies
	parameter.			
PLMSuppliersIdentificationNumber*	A unique number that a supplier itself is given.	Text	Primary Product Type Sample	None
PLMSuppliersIngredient Quantity	Gives a quantity of additives, ingredients, sub-ingredients, or sub-ingredient additives in the suppliers sample / recipe.	Number	Product Reference Type - Additive - Ingredient - Sub-ingredient - Sub-ingredient Additive	None
PLMSuppliersName*	The name of the supplier.	Text	Product Reference Type Sample	None
PLMVersionNumber* Note: This is a specification attribute	The recipe version number sent.	Number	Product Reference Type Sample	None

List Of Values for Attributes

LOVs IDs	Name	LOV for which Attributes
PLMMandatorySupplierResponsesAdditiveLOV	Mandatory Supplier Responses, Additives	PLMMandatorySupplierResponsesAdditives

LOVs IDs	Name	LOV for which Attributes
PLMMandatorySupplierResponsesParamLOV	Mandatory Supplier Responses, Parameter	• PLMMandatorySupplierResponsesParam
PLMMandatorySupplierResponsesReqLOV	Mandatory Supplier Responses, Requirements	• PLMMandatorySupplierResponsesReq
PLMCountryCodeISO3166-1_2013	Country Code - ISO_3166-1 2013	• PLMGuaranteedCountryOfOrigin
PLMMeetsRequirementLOV	Meets Requirement	• PLMMeetsRequirement
PLMParameterTypeLOV	Parameter Type	• PLMParameterType
PLMRequirementTypeLOV	Requirement Type	• PLMRequirementType
PLMSpecifiedIngredientAllowanceLOV	Specified Ingredient Allowance	• PLMSpecifiedIngredientAllowance
PLMSpecifiedIngredientPrecisionLOV	Specified Ingredient Precision	• PLMSpecifiedIngredientPrecision

Private Label Food Solution Setup in Web UI

The private label food solution enable the customer to look at a product from two or more suppliers and view the similarities and differences of the product's makeup from one supplier to the next. Additionally, it allows the customer to compare the supplier recipes in relation to their own recipe specification. There are a number of component tabs and reference types that need to be created and properly setup in Web UI for the private label food solution to function properly.

Read the following sections below on how to properly set up the private label food solution in Web UI.

- Adding the Compare BOMs Action Button
- Adding and Mapping Tabs for Private Label Food Solution

Note: Suppliers can get a record of their specification response that they submit to the customer when they respond to a bid for a private label product. If chosen for production, the supplier's response is the basis for contractual agreements. This record outlines the bill of materials used to fulfill the private label product, and provides explanations on how the supplier will meet the customer's requirements. For more information see the Private Label Food Solution segment of the Product Lifecycle Management section in the Solution Enablement documentation.

Adding the Compare BOMs Action Button

There is a pivotal action button needed for the private label food solution, the **Compare BOMs Action** button. This action button enables users to select recipes that suppliers have returned, and evaluate the recipes against other suppliers and the original recipe specification. Users can review the ingredients, compound ingredients, and sub-ingredients used in the recipe, and evaluate the parameters and requirements to help them make a decision on which recipe to select.

This action button can be added to any Node List Properties or Task List Properties in conjunction with the compare tab child components via a Node Details screen.

Important: PLMSamples are the only object type that render data on a Node List Properties or Task List Properties properly. If other object types are used or selected, the compare tabs will not function as expected.

See the following topics in the **Web User Interface / Web UI Setup and User Guide** for more information:

- Node Details Screen
- Node List Component
- Task List Properties

Adding the Compare BOMs Action

1. Navigate to the desired Node List Properties or Task List Properties. In the following example, a Node List Properties is used.
2. Under Child Components, click on the **Add** button to add the **Compare BOMs Action** button.

Properties
Configuration Web UI style

PLM Supplier Sampl Save Close New... Delete Rename Save

Node List Properties [go to parent](#)

Child Components

Display Modes

Table Display Mode

Add.. Remove Up Down

Actions

Add.. Remove Up Down

Add Component X

- Bulk Update Template Action
- Cancel Asynchronous Job Action
- Change Reference Target Action
- Compare Boms Action**
- Confirm Duplicate From Grid Action
- Create And Link Design Specification Action
- Create Asset Action
- Create Classification Action

A screen for comparing a design specification with a number of BOMs

Filter

☐ Show deprecated components

✓ Add X Cancel

Note: If adding a Compare BOMs Action Button to a Task List Properties know that you cannot submit any item in the workflow when on any of the compare tabs. For more on the compare tabs, see **Private Label Food Solution** topic in the **Private Label Food Solution** section of the **PLM for Users** documentation. You can only submit an item in a workflow from a Node Details when accessed from a Task List Properties.

Once the Compare BOMs Action is added, the desired compare tabs need to be added and proper mapping needs to take place. For more on how to add the compare tabs or how to map everything, see the **Adding and Mapping Tabs for Private Label Food Solution** topic in this section.

Adding and Mapping Tabs for Private Label Food Solution

To support the process of the private label food solution, various tabs need to be added and configured. The private label food solution component supports Bill of Materials (BOMs) creation for the customer, the supplier, and the ability for customers to compare all BOMs against their own recipe specifications. The sections below describe how to add, configure, and map the private label food solution tabs.

Adding Tabs

1. Open the Designer in Web UI, and create a new Node Details Properties screen. In the example below, the screen is called 'PLM Compare Supplier Samples.'

Add Screen [X]

Screen ID

PLM Compare Supplier Samples

Background Process Node Details

Merge Nodes Screen

Multi Node Viewer

Node Details

Node List Browser

Top level component for creating a node editor. Can edit any node type. Also works for editors that depends on STEP Workflow.

Filter

node

☐ Show deprecated components

✓ Add [X] Cancel

2. Under Child Components > Main > select Tab Control from the dropdown and then click **go to component**.

Properties

Configuration

Web UI style

PLM Compare Suppl ▾

Save

Close

New...

Delete

Rename

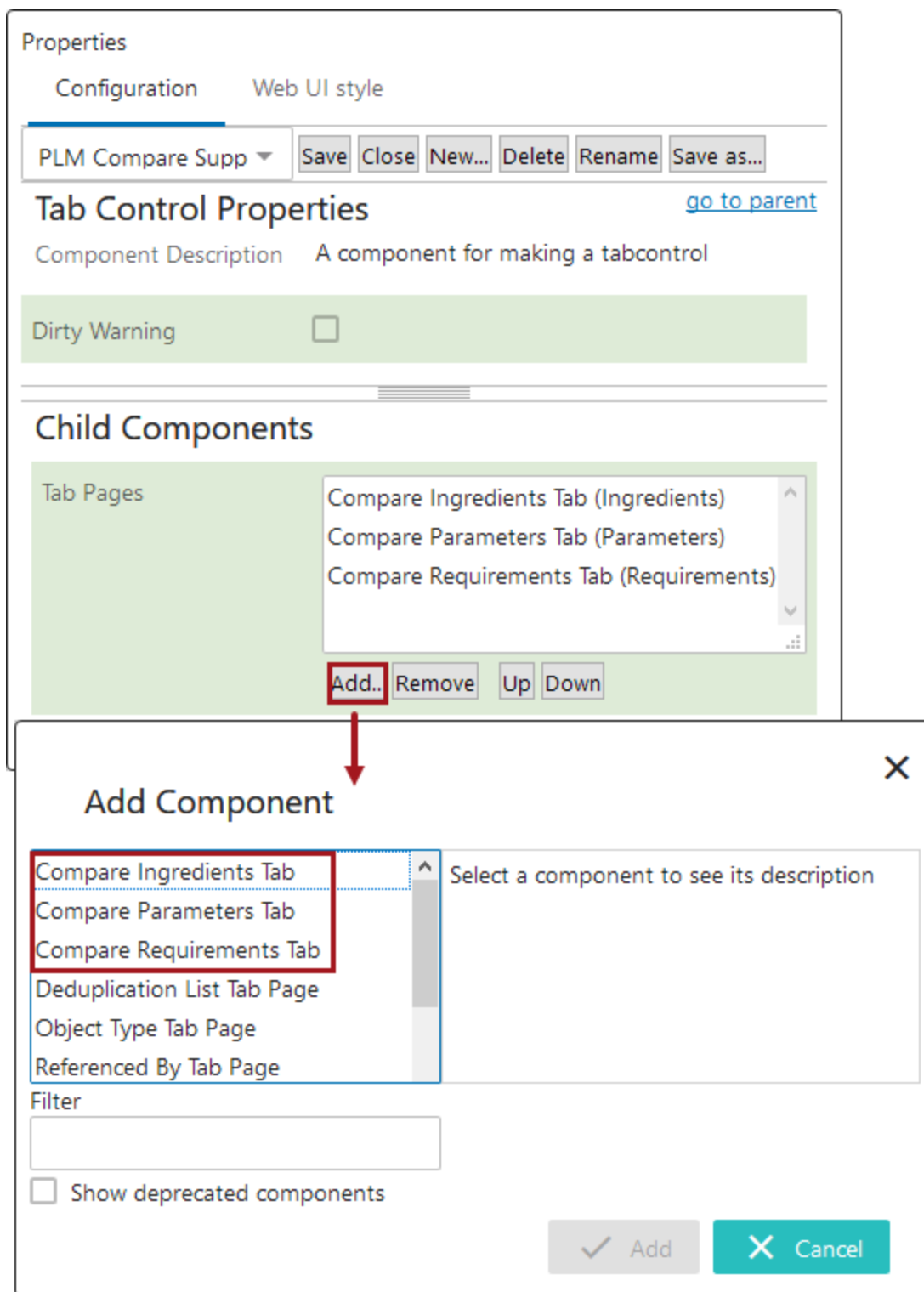
Save as...

Node Details Properties

Child Components

Below Title	<Select a child component> ▾	go to component
Main	Tab Control ▾	go to component
Buttons	<Select a child component> ▾	go to component

3. On the Tab Control Properties under Child Components in the Tab Pages field, add the following tabs for the private label food solution depending on business needs:
 - Specify Ingredients Tab
 - Supplier Ingredients Tab
 - Compare Ingredients Tab
 - Compare Parameters Tab
 - Compare Requirements Tab



4. Once a tab type is added, double click on it to configure. For more on how to configure each tab type, see the following topics:
 - Configuring the Specify Ingredients Tab
 - Configuring the Supplier Ingredients Tab

- Configuring the Compare Tabs

Note: The 'Configuring the Compare Tabs' topic describes the configurations for the Compare Ingredients Tab, Compare Parameters Tab, and Compare Requirements Tab.

- Repeat step three and four to add any other additional private label food solution tabs needed.

Properties (edited)

Configuration Web UI style

PLM Compare Supp ▼ Save Close New... Delete Rename Save as...

Tab Control Properties [go to parent](#)

Component Description A component for making a tabcontrol

Dirty Warning ☐

Child Components

Tab Pages

- Compare Ingredients Tab (Ingredients)
- Compare Parameters Tab (Parameters)
- Compare Requirements Tab (Requirements)

Add.. Remove Up Down

Mappings

In order for the private label food solution tabs to display correctly in Web UI, they need to be mapped correctly. Below explains how to map each private label food solution tab. For more details on mappings, see the **Mappings** topic in the **Using a Web UI** section of the **Web User Interfaces / Web UI Setup and User Guide** documentation.

Specify Ingredients Tab

For the Specify Ingredients Tab to work properly in Web UI, the following mapping needs to take place:

Note: This could work with either an Object Type Condition or a Workflow State Condition. In the example below, Object Type Condition is used.

1. In the designer under ---[MAIN]--- > Mappings > Add > Conditions > Add an **Object Type Condition**.
2. Select the object type that represents the Design Specification Variant, in this example it is called PLMDesignSpecificationVariant.
3. Next, select the designated Node Details screen that was created for the Specify Ingredients Tab, in this case the Node Details screen is called PLM Specify Variant. Click **Save**.

Edit component [X]

Screen Mapping Properties

Component Description: A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

*Conditions: ObjectType = PLMDesignSpecificationVariant

[Add...] [Edit...] [Remove] [Up] [Down]

*Screen: PLM Specify Variant [Add]

[✓ Save] [X Cancel]

4. When finished, the mapping should look similar to the picture below.

Properties (edited)

Configuration Web UI style

---[MAIN]---

Save Close New... Delete Rename Save as...

Main Properties

Component Description

MAIN is used for configuring the overall behaviour of the Web UI. For example by setting up conditional mappings it is possible to decided the behaviour when navigating the Web UI. In addition the different side panels (left, right, top, bottom) and Corner bar can be configured on MAIN.

Mappings

PLM Specify Variant (ObjectType = PLMDesignSpecificationVariant)

Add... Edit... Remove Up Down

Supplier Ingredients Tab

For the Supplier Ingredients Tab to work properly in Web UI, the following mapping needs to take place.

Note: This could work with either an Object Type Condition or a Workflow State Condition. In the example below, Object Type Condition is used.

1. In the designer under ---[MAIN]--- > Mappings > Add > Conditions > Add an **Object Type Condition**.
2. Select the object type that represents the sample. In this example it is called PLMSample.
3. Next, select the designated Node Details screen that was created for the Supplier Ingredients Tab. In this case the Node Details screen is called PLM Food Sample. Click **Save**.

✕

Edit component

Screen Mapping Properties

Component Description

A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

*Conditions

ObjectType = PLMSample

Add...

Edit...

Remove

Up

Down

*Screen

PLM Food Sample

Add

✓ Save

✕ Cancel

4. When finished, the mapping should look similar to the picture below.

Properties

Configuration

Web UI style

---[MAIN]---

Save

Close

New...

Delete

Rename

Save as...

Main Properties

Component Description

MAIN is used for configuring the overall behaviour of the Web UI. For example by setting up conditional mappings it is possible to decided the behaviour when navigating the Web UI. In addition the different side panels (left, right, top, bottom) and Corner bar can be configured on MAIN.

Mappings

PLM Food Sample (ObjectType = PLMSample)

Add...

Edit...

Remove

Up

Down

Compare Tabs

In order for the Compare Ingredients Tab, Compare Parameters Tab, and Compare Requirements Tab to work properly in Web UI when making a selection, then using the Compare BOMs Action, the following mapping needs to take place. An example using the Compare BOMs Condition is below:

Note: The Compare BOMs Action figures out which design specification variant is needed to display for grouping the recipe(s), and passes that information along with which samples were selected to the compare tab screens. The Compare BOMs Condition determines which screen should be used.

1. In the designer under ---[MAIN]--- > Mappings > Add > Conditions > Add the **Compare BOMs Condition**.
2. Next, select the designated Node Details screen that was created for the compare tabs, in this case the Node Details screen is called PLM Compare Supplier Samples. Click **Save**.

Edit component [X]

Screen Mapping Properties

Component Description: A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

*Conditions: Compare BOMs Condition [Add... Edit... Remove Up Down]

*Screen: PLM Compare Supplier Samples [Add]

[Save] [Cancel]

3. When finished, the mapping should look similar to the picture below.

Properties

Configuration

Web UI style

---[MAIN]---

Save

Close

New...

Delete

Rename

Save as...

Main Properties

Component Description

MAIN is used for configuring the overall behaviour of the Web UI. For example by setting up conditional mappings it is possible to decided the behaviour when navigating the Web UI. In addition the different side panels (left, right, top, bottom) and Corner bar can be configured on MAIN.

Mappings

PLM Compare Supplier Samples (Compare BOMs Condition)

Add...

Edit...

Remove

Up

Down

Configuring the Specify Ingredients Tab

Recipe specifications from customers aid suppliers in knowing exactly what to include or exclude in their sample recipes back to customers.

To set up the Specify Ingredients tab, and customize information provided to the supplier, a number of fields need to be configured in the designer:

Properties

Configuration
Web UI style

PLM Food Var Reci
Save
Close
New...
Delete
Rename
Save

Specify Ingredients Tab Properties
[go to parent](#)

Component Description
A tab screen to specify ingredients

Business Condition
... Clear

Title

Ingredients Table Title

Show Ingredients Table Title
☒

Ingredients Table Columns

PLMSpecifiedIngredientAllowance
PLMSpecifiedIngredientPrecision
PLMSpecifiedIngredientQuantity
PLMSpecificationDetails

Add... Remove Up Down

* Edit Dialog Attributes

PLMSpecifiedIngredientAllowance
PLMSpecifiedIngredientPrecision
PLMSpecifiedIngredientQuantity
PLMSpecificationDetails

Add... Remove Up Down

Read Only
☐

Read Only Exceptions
... Clear

Post Copy Business Action
... Clear

Business Function for Ingredient Copy
... Clear

* Ingredient Group Object Type
...

- **Business Condition:** Select a preconfigured business condition that determines whether the tab will display. If the condition returns true, the tab displays. If it returns false, the tab does not display. The business condition is evaluated when a screen is saved or refreshed.
- **Title:** Enter a name for the Specification Tab.
- **Ingredients Table Title:** Enter a name for the title of the recipe specification table.
- **Show Ingredients Table Title:** Enabling this shows the Ingredients Table title.
- **Ingredients Table Columns:** Allows for customizable columns in the recipe specification.

Note: The columns for 'Ingredient Name' and 'Ingredient Type' are default. They cannot be removed from the table or changed in Web UI.

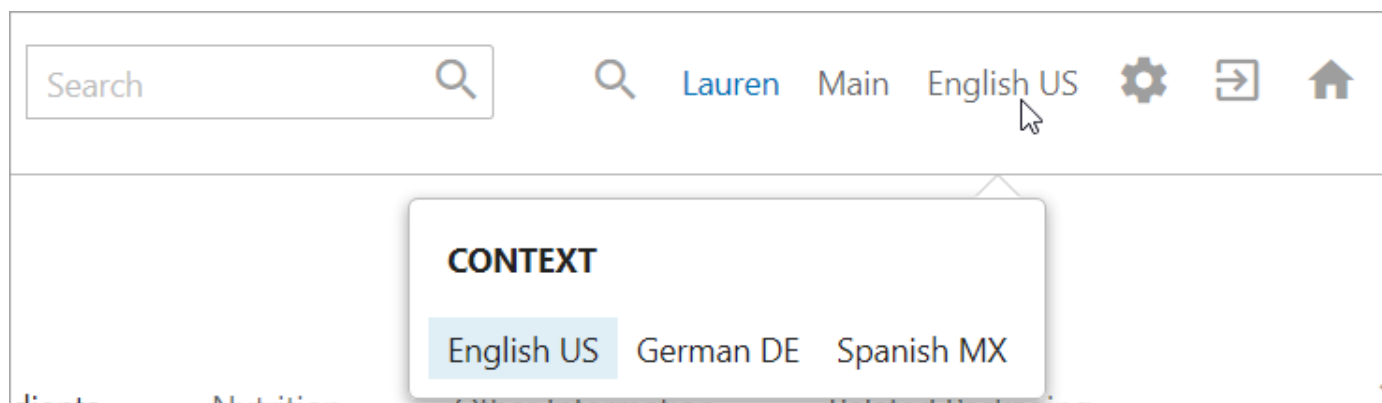
- **Edit Dialog Attributes:** Controls which attributes are added to the 'Add Ingredients' dialog in Web UI and allows for them to be ordered according to needs.

Important: The attributes with the IDs PLMSpecifiedIngredientAllowance, PLMSpecifiedIngredientPrecision, and PLMSpecifiedIngredientQuantity must be configured in the Edit Dialog Attributes in order for validations to work against the supplier's recipe and the Compare Tabs.

- **Read Only:** When enabled, all fields are not able to be edited unless there is an attribute group selected for the Read Only Exceptions.
- **Read Only Exceptions:** Select an attribute group that has description attributes valid for the PLMIngredient reference type. Any description attributes in this group that were added to the 'Edit Dialog Attributes' field can still be edited even when the 'Read Only' parameter is selected. However, if the 'Read Only' parameter is not selected, then any attributes in this group are not able to be edited.
- **Post Copy Business Action:** A business action that is invoked after a library ingredient is copied to a specified ingredient.
- **Business Function for Ingredient Copy:** A business function that returns attribute groups and contexts that is used for copying values maintained on the library ingredient to the specified ingredient.
- **Ingredient Group Object Type:** The object type for ingredient groups used for recipe specifications and supplier validations.

Language Translations

It is now possible for quality managers to select ingredients or additives in their own language, such as English, change the context in the corner bar of a Web UI to a different context, such as Spanish, and if the ingredients have been translated, view the translated ingredient names and additives. This enables quality managers to work in their own language, and suppliers to see the selected ingredients or additives in their preferred language.



Copying Pre-translated Ingredients

To copy pre-translated ingredient names and attribute values maintained on the ingredients, a business function is used. When an ingredient is added to the recipe specification, a business function is called that will return one or more pairs of attribute groups to be copied for a specific context. The business function can return multiple attribute groups, and ingredient and additive names are copied in all available contexts.

Note: Attribute values are only copied in the contexts that are returned by the business function.

The business function is created in workbench and will look similar to the example below:

View Operation

JavaScript Function

Bind

Binds

Variable name	Binds to
manager	STEP Manager
logger	Logger

Message

Messages

Variable name	Message	Translations
---------------	---------	--------------

Input Parameters

Parameters

Parameter name	Type	Description
source	Node	Source node
target	Node	Target node

Return Type

Return Type

Return Type
List<String>

JavaScript

```
1 var retVal = new java.util.ArrayList();
2
3 retVal.add("CopyIngredientAttributes");
4 retVal.add("Context1");
5
6 retVal.add("CopyIngredientAttributes");
7 retVal.add("Context2");
8
9 return retVal;
```

Edit externally

Close

Input parameters for the business function are:

- **source:** this is the Node that will contain the source object. It can be a design specification variant or a specified compound ingredient.
- **target:** this is the Node containing the target object for the reference. It can be a specified ingredient, specified compound ingredient, or a specified additive.

In Web UI on the Specified Ingredients Tab, there is a field in the designer to select the desired business function.

Business Function for Ingredient Copy ... Clear

Post Copy Business Action

This business action can be used when copying parameter values when an ingredient is added to a specification and the ingredient has associated parameters.

The business action is optional. It may be used to replace the handling of parameters attached to ingredients when they are specified or proposed by suppliers, or for other purposes.

As an example, if parameters are maintained on ingredients, the business action can:

- Determine which parameter values should be taken
- Determine which contexts the parameter values should be copied from and sent to
- Determine which reference types should be applied from the specified ingredient to the parameter

The business action will be invoked after each time one of the following instances occurs:

- An ingredient is selected from the library.
- A specified ingredient, compound ingredient, or additive is created.
- A specified ingredient reference is created between a library ingredient and a specified ingredient, compound ingredient, or additive.
- A reference is created between a design specification variant and a specified ingredient, compound ingredient, additive or additive class.
- A reference is created between a compound ingredient and an ingredient.
- The business function for 'ingredient copy' is invoked.

Input parameters for the business action are:

- **ingredientReference**: this is a reference type that was created between a design specification variant and a specified ingredient, or a reference between a specified compound ingredient and a specified ingredient before the business action was invoked.

This business action can be added to Web UI on a Specified Ingredients Tab by selecting the desired Post Copy Business Action.

Post Copy Business
Action

... Clear

Note: The Specified Ingredients Tab in Web UI will not refresh any data when the business action is invoked. Therefore, the action should not be used to affect data on the screen.

Configuring the Supplier Ingredients Tab

Recipe samples from suppliers detail exactly what was put into the proposed recipe. It is important to customize the display and sequence for the suppliers' view to ensure that the supplier provides back to the customer any necessary information on the ingredient, additive, or compound ingredient being added to their sample recipe.

To set up the Supplier Ingredients tab properly, a number of fields need to be configured in the designer:

Properties

Configuration
Web UI style

PLM Food Supplier
Save
Close
New...
Delete
Rename
Save as...

Supplier Ingredients Tab Properties
[go to parent](#)

Component Description
A tab screen to supply ingredients

Business Condition
... Clear

Title

Show Ingredients
☒

Table Title

Ingredients Table Title

Ingredients Table Columns

Add... Remove Up Down

Add... Remove Up Down

* Edit Dialog Attributes

Read Only
☐

Read Only Exceptions
... Clear

Post Copy Business Action
... Clear

... Clear

Business Function for Ingredient Copy

Caller parameters:
Function input parameters:

Source Node

Target Node

* Ingredient Group Object Type
...

Ingredient Reference Type
... Clear

Additive Reference Type
... Clear

Ingredient Root
... Clear

Additive Root
... Clear

- **Business Condition:** Select a preconfigured business condition that determines whether the tab will display. If the condition returns true, the tab displays. If it returns false, the tab does not display. The business condition is evaluated when a screen is saved or refreshed.
- **Title:** Fill in the field with an appropriate name for the tab.
- **Show Ingredients Table Title:** Select this if you want the table title to show.
- **Ingredient Table Title:** Fill in the field with an appropriate name for the table.
- **Ingredients Table Columns:** Add and order any Ingredient Table Columns needed.

Note: The column for Ingredient Name is default. It cannot be removed from the table or changed in Web UI.

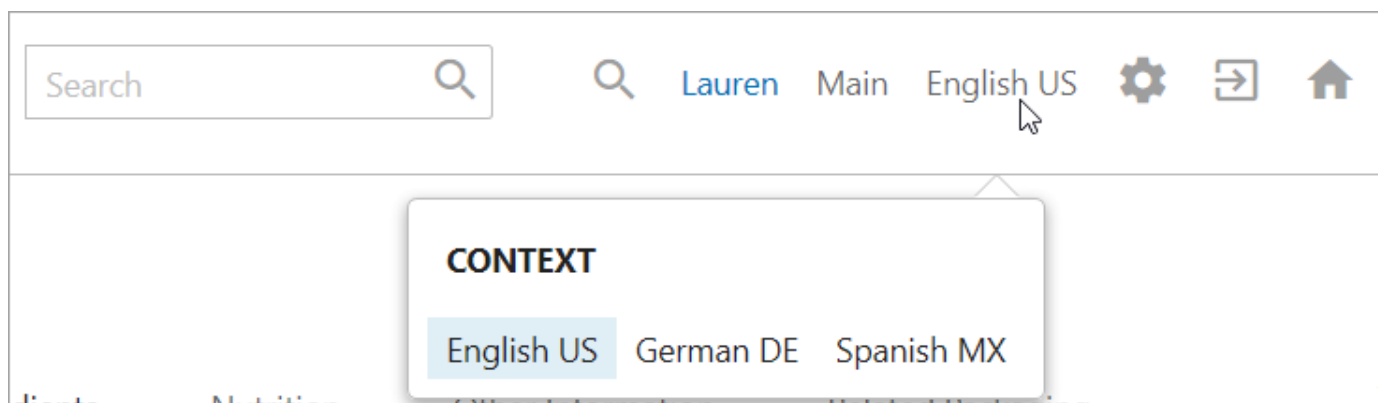
- **Edit Dialog Attributes:** Controls which attributes are added to the 'Add Ingredients' dialog in Web UI and allows for them to be ordered according to needs.

Important: The attribute with the ID PLMSuppliersIngredientQuantity must be in this group for the validations and alerts to work. Additionally, the display order in the pop-up dialog must follow the sort order of the parameter.

- **Read Only:** When enabled, all fields are not able to be edited unless there is an attribute group selected for the Read Only Exceptions.
- **Read Only Exceptions:** Select an attribute group that has description attributes valid for the PLMIngredient reference type. Any description attributes in this group that were added to the 'Edit Dialog Attributes' field can still be edited even when the 'Read Only' parameter is selected. However, if the 'Read Only' parameter is not selected, then any attributes in this group are not able to be edited.
- **Post Copy Business Action:** A business action that is invoked after a library ingredient is copied to a specified ingredient.
- **Business Function for Ingredient Copy:** A business function that returns attribute groups and contexts that is used for copying values maintained on the library ingredient to the supplier's ingredient.
- **Ingredient Group Object Type:** The object type for ingredient groups used for recipe specifications and supplier validations.
- **Ingredient Reference Type:** Add the appropriate reference for the Ingredient Reference Type.
- **Additive Reference Type:** Add the appropriate reference for the Additive Reference Type.
- **Ingredient Root:** Select the appropriate Ingredient Root folder to search below.
- **Additive Root:** Select the appropriate Additive Root.

Language Translations

It is now possible for quality managers to select ingredients or additives in their own language, such as English, change the context in the corner bar of a Web UI to a different context, such as Spanish, and if the ingredients have been translated, view the translated ingredient names and additive classes. This enables suppliers to work in their own language, and customers to see the selected ingredients or additive classes in their preferred language.



Copying Pre-translated Ingredients

To copy pre-translated ingredient names and attribute values maintained on the ingredients, a business function is used. When an ingredient is added to the recipe specification, a business function is called that will return one or more pairs of attribute groups to be copied for a specific context. The business function can return multiple attribute groups, and ingredient and additive names are copied in all available contexts.

Note: Attribute values are only copied in the contexts that are returned by the business function.

The business function is created in workbench and will look similar to the example below:

JavaScript Function

Binds:

Variable name	Binds to
manager	STEP Manager
logger	Logger

Messages:

Variable name	Message	Translations
---------------	---------	--------------

Input Parameters:

Parameter name	Type	Description
source	Node	Source node
target	Node	Target node

Return Type:

Return Type
List<String>

JavaScript:

```

1 var retVal = new java.util.ArrayList();
2
3 retVal.add("CopyIngredientAttributes");
4 retVal.add("Context1");
5
6 retVal.add("CopyIngredientAttributes");
7 retVal.add("Context2");
8
9 return retVal;
10

```

Edit externally

Input parameters for the business function are:

- **source:** this is the Node that will contain the source object. It can be a design specification variant or a specified compound ingredient.
- **target:** this is the Node containing the target object for the reference. It can be a specified ingredient, specified compound ingredient, or a specified additive.

In Web UI on the Supplier Ingredients Tab, there is a field in the designer to select the desired business function.

Business Function for Ingredient Copy ... Clear

Post Copy Business Action

This business action can be used when copying parameter values when an ingredient is added to a specification and the ingredient has associated parameters.

The business action is optional. It may be used to replace the handling of parameters attached to ingredients when they are specified or proposed by suppliers, or for other purposes.

As an example, if parameters are maintained on ingredients, the business action can:

- Determine which parameter values should be taken
- Determine which contexts the parameter values should be copied from and sent to
- Determine which reference types should be applied from the specified ingredient to the parameter

The business action will be invoked after each time one of the following instances occurs:

- An ingredient is selected from the library.
- A supplier response ingredient, compound ingredient, or additive is created.
- An ingredient reference is created between a library ingredient and a supplier response ingredient, compound ingredient, or additive.
- A reference is created between a sample and a supplier response ingredient, compound ingredient, additive, additive class, or ingredient group.
- A reference is created between a compound ingredient and an ingredient.
- The business function for 'ingredient copy' is invoked.

Input parameters for the business action are:

- **ingredientReference**: this is a reference type that was created between a design specification variant and a specified ingredient, or a reference between a specified compound ingredient and a specified ingredient before the business action was invoked.

This business action can be added to Web UI on a Supplier Ingredients Tab by selecting the desired Post Copy Business Action.

Post Copy Business
Action
... Clear

Note: The Specified Ingredients Tab in Web UI will not refresh any data when the business action is invoked. Therefore, the action should not be used to affect data on the screen.

Configuring the Compare Tabs

While the compare tabs can be used individually, they are often used together to help a customer see the similarities and differences from the suppliers' recipe samples in comparison to the recipe specification. To configure each compare tab see below.

Compare Ingredients Tab

This tab enables the customer to directly compare the suppliers' recipes against their own recipe specification.

Properties

Configuration

Web UI style

PLM Food Var Reci

SaveCloseNew...DeleteRenameSave

Compare Ingredients Tab

[go to parent](#)

Properties

Component Description

A tab screen for comparing ingredients across samples

Business Condition

... Clear

Title

Compare Recipes

Supplied Ingredient Reference Types

PLMIngredient

Add...RemoveUpDown

Supplied Additive Ingredient Reference Types

PLMAdditive

Add...RemoveUpDown

- **Business Condition:** Select a preconfigured business condition that determines whether the tab will display. If the condition returns true, the tab displays. If it returns false, the tab does not display. The business condition is evaluated when a screen is saved or refreshed.
- **Title:** Enter in a title for the Compare Ingredients tab.
- **Supplied Ingredient Reference Types:** References used for supplier ingredients. Add the appropriate references.
- **Supplied Additive Ingredient Reference Types:** References used for supplier additives. Add the appropriate references.

Note: More than one additive or ingredient reference type can be added. If more than one additive or ingredient reference type is added, it does not matter what order they are displayed in.

Compare Parameters Tab

This tab enables the customer to compare the supplier's parameters against one another and against the recipe specification.

Properties

Configuration
Web UI style

PLM Food Var Rec
Save
Close
New...
Delete
Rename
Save

Compare Parameters Tab Properties

[go to parent](#)

Component Description

A tab screen for comparing parameters across samples

Business Condition

...

Clear

Title

Compare Parameters

Show Ingredients Parameters

☒

Ingredient Parameter Reference Type

PLMParameter

...

Clear

Specified Parameters

PLMParameter

...

...

Add...

Remove

Up

Down

Sample Parameters

PLMParameter

...

...

Add...

Remove

Up

Down

Child Components

Additional Display Row Attributes

Reference Metadata Attribute ((PL

PLM Flex Defining Attribute ((PLM

...

...

Add..

Remove

Up

Down

- **Business Condition:** Select a preconfigured business condition that determines whether the tab will display. If the condition returns true, the tab displays. If it returns false, the tab does not display. The business condition is evaluated when a screen is saved or refreshed.
- **Title:** Add an appropriate title for the Compare Parameters tab.
- **Ingredient Parameter Reference Type:** The reference type(s) between the supplier response ingredients and the Parameter, from where the values for the Parameter Description in the comparison table will be displayed.
- **Specified Parameters:** Add the appropriate specified parameter references.

Note: More than one parameter reference type can be added. If more than one reference type is added, it does not matter what order they are displayed in.

- **Sample Parameters:** Add the appropriate sample parameter references.
- **Additional Display Row Attributes:** Add any needed attributes that should appear in the Detail Requirements Compare dialog. The attributes can be either a PLM Flex Defining Attribute or a Reference Metadata Attribute.

Note: When adding a PLM Flex Value Attribute you are adding a defining attribute. A defining attribute is an LOV that contains a list of other attribute IDs that are used to control the format of data in a column in a Multi-Reference Editor. Users can choose one of the LOV values to define how the attribute's format should be controlled for the supplier's response.

Add Component

PLM Flex Defining Attribute
Reference Metadata Attribute

^
v

Select a component to see its description

Filter

☐ Show deprecated components

Cancel

Add

For more information on a defining attribute, see the **Multi-Reference Editor** topic in this documentation.

Compare Requirements Tabs

This tab enables the customer to directly compare the suppliers' requirements against one another and against the recipe specification.

Properties

Configuration
Web UI style

PLM Food Var Reci
Save
Close
New...
Delete
Rename
Save

Compare Requirements Tab
[go to parent](#)

Properties

Component Description
A tab screen for comparing ingredients across samples

Business Condition

...
Clear

Title

Compare Requirements

Specified Requirements

PLMRequirement

Add...
Remove
Up
Down

Recipe Requirements

PLMRequirement

Add...
Remove
Up
Down

Child Components

Additional Display Row Attributes

Add..
Remove
Up
Down

- **Business Condition:** Select a preconfigured business condition that determines whether the tab will display. If the condition returns true, the tab displays. If it returns false, the tab does not display. The business condition

is evaluated when a screen is saved or refreshed.

- **Title:** Add an appropriate title for the Compare Requirements tab.
- **Specified Requirements:** Add the appropriate references for the Specified Requirements field.
- **Recipe Requirements:** Add the appropriate recipe references.

Note: More than one requirement reference type can be added. If more than one reference type is added, it does not matter what order they are displayed in.

- **Additional Display Row Attributes:** Add any needed attributes that should appear in the Detail Requirements Compare dialog. The attributes can be either a PLMFlexDefiningAttribute or a ReferenceMetadataAttribute.

Note: When adding a PLM Flex Value Attribute you are adding a defining attribute. A defining attribute is an LOV that contains a list of other attribute IDs that are used to control the format of data in a column in a Multi-Reference Editor. Users can choose one of the LOV values to define how the attribute's format should be controlled for the supplier's response.

Add Component

PLMFlexDefiningAttribute
ReferenceMetadataAttribute

Select a component to see its description

Filter

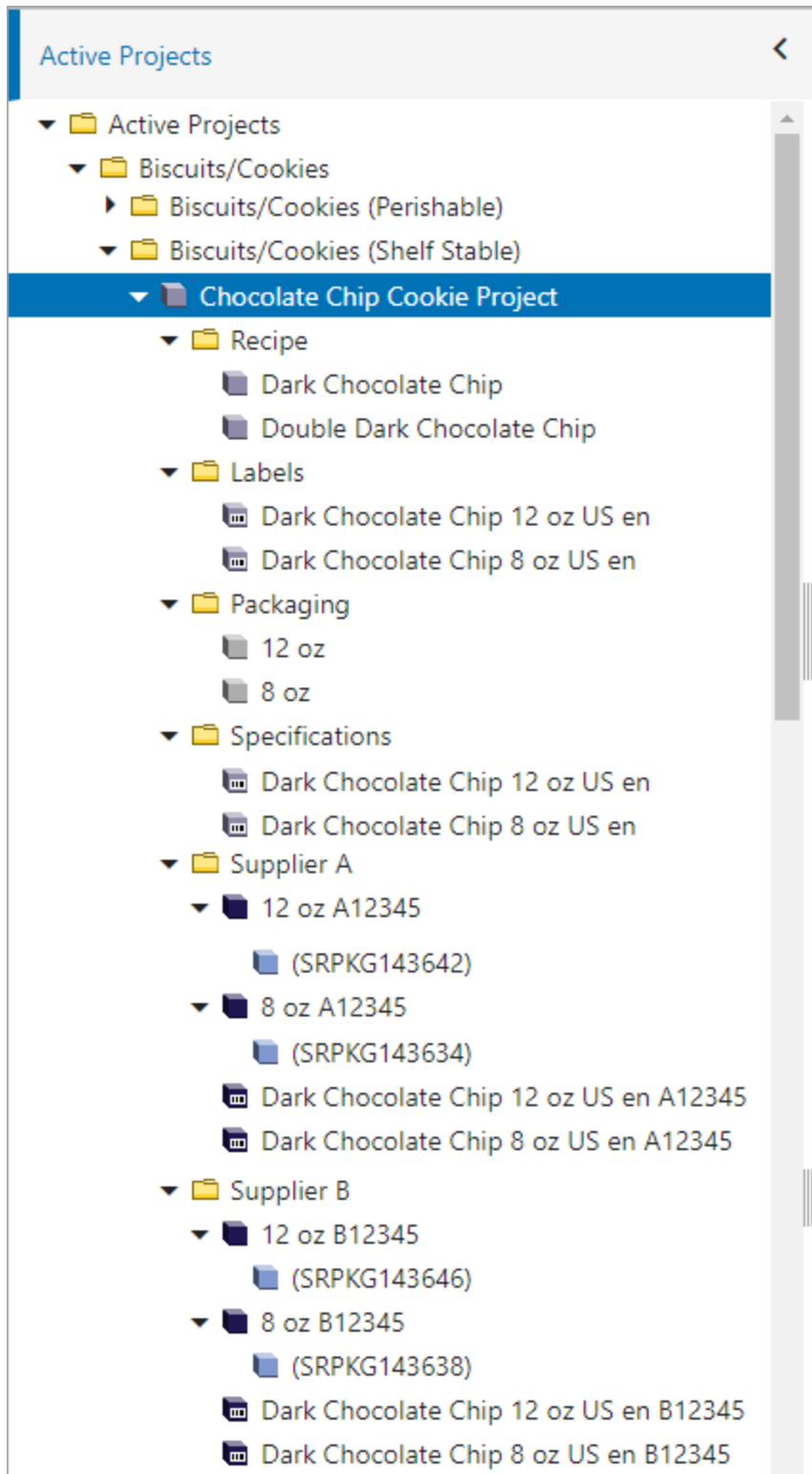
☐ Show deprecated components

Cancel Add

For more information on a defining attribute, see the **Multi-Reference Editor** topic in this documentation.

Configuring Project Navigator

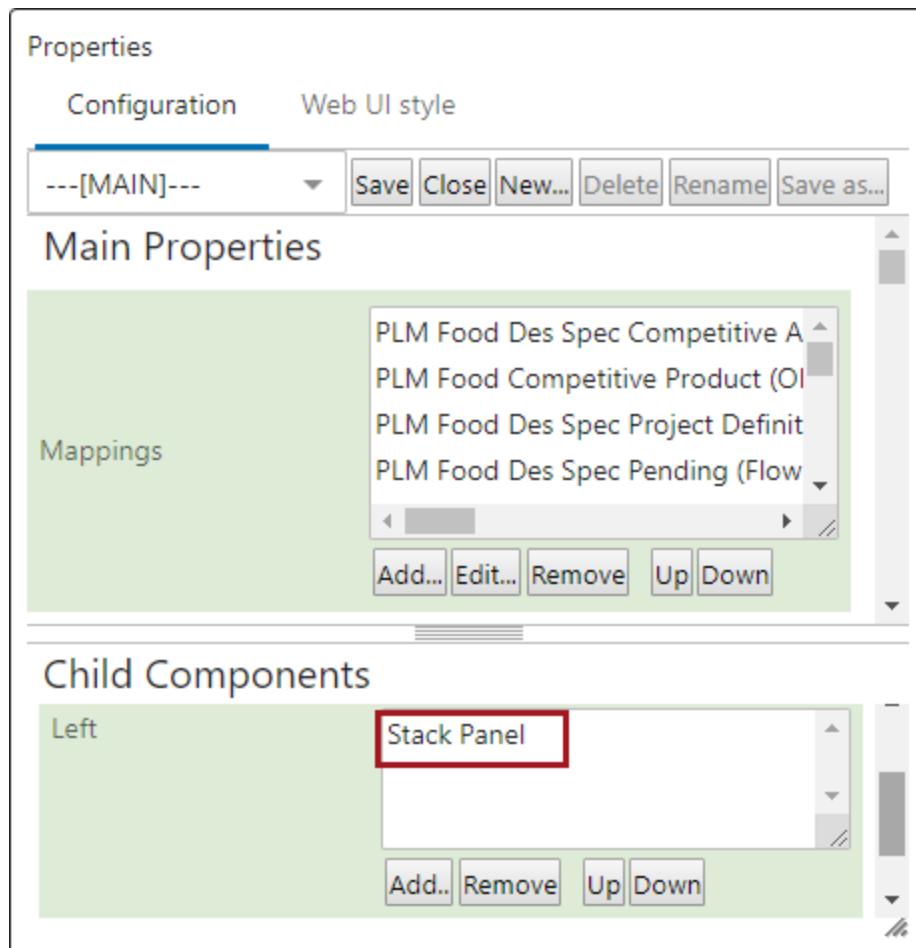
Configure Project Navigator to access design specifications (active projects) in workflows via a classification folder in the Tree navigation. Users get a complete view of all the projects, objects, and references associated to the design specification, such as recipes, packaging, labels, or suppliers. This enables users to quickly navigate to the Tree where they can select the classification folder configured to house active projects.



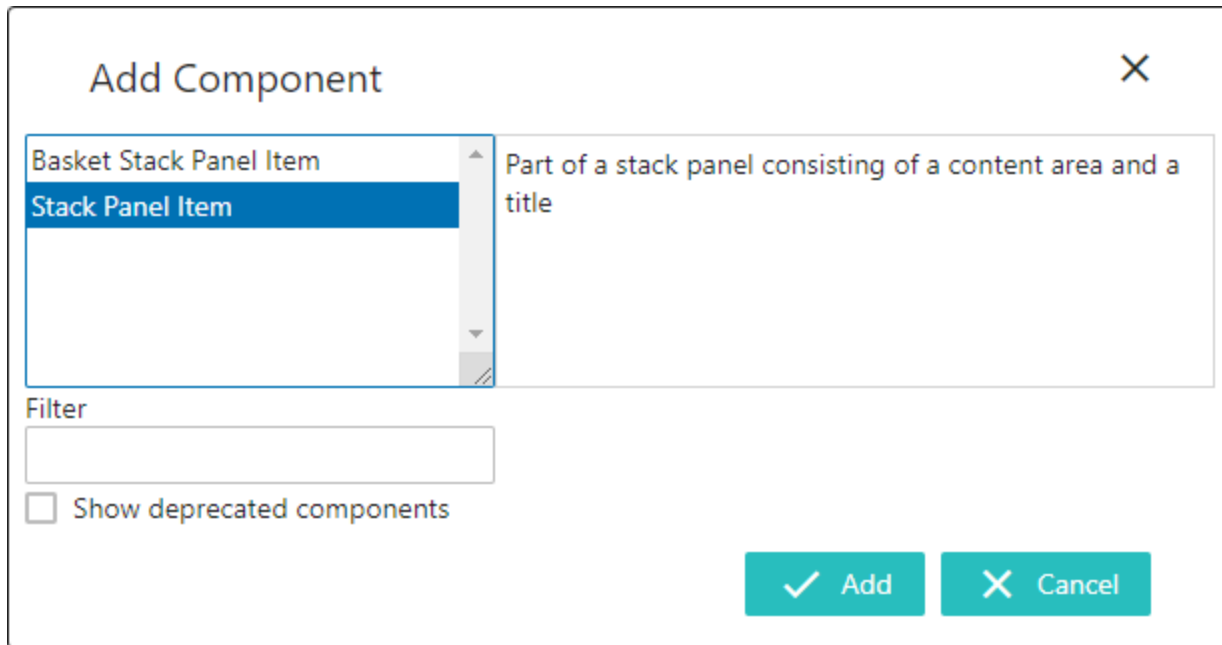
To configure Project Navigator, follow the directions below:

Note: Suppliers will only have access to the active projects they are associated with and will not have access to other supplier's active projects.

1. In Web UI, go to the designer and select --[MAIN]-- from the dropdown. Under Child Components, add a **Stack Panel** to either the Right or Left field.

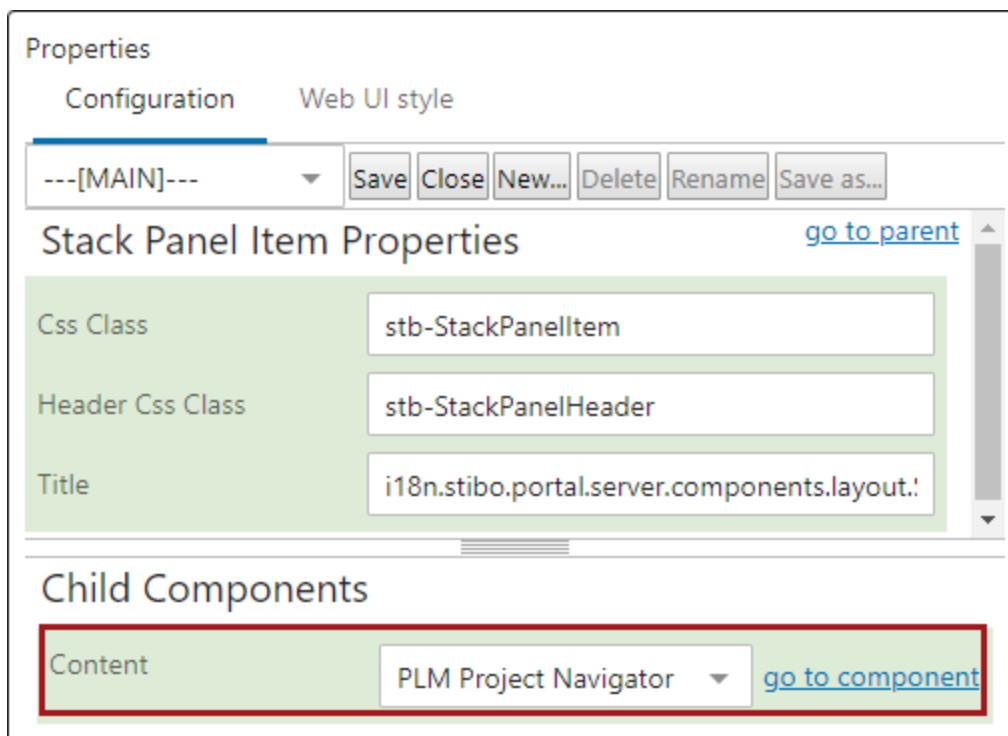


2. Double click the Stack Panel and add a **Stack Panel Item**.



The 'Add Component' dialog box features a title bar with a close button (X). It contains a list box on the left with 'Basket Stack Panel Item' and 'Stack Panel Item' (the latter is selected). To the right of the list box is a description: 'Part of a stack panel consisting of a content area and a title'. Below the list box is a 'Filter' text field and a checkbox labeled 'Show deprecated components'. At the bottom right are two buttons: 'Add' (with a checkmark icon) and 'Cancel' (with an X icon).

3. Double click the Stack Panel Item, and in the Child Components > Content dropdown field, select **PLM Project Navigator**, and then click on **go to components**.



The 'Stack Panel Item Properties' dialog box has two tabs: 'Configuration' (selected) and 'Web UI style'. Below the tabs is a dropdown menu showing '---[MAIN]---' and several action buttons: 'Save', 'Close', 'New...', 'Delete', 'Rename', and 'Save as...'. The main area is titled 'Stack Panel Item Properties' and contains three input fields: 'Css Class' with value 'stb-StackPanellItem', 'Header Css Class' with value 'stb-StackPanelHeader', and 'Title' with value 'i18n.stibo.portal.server.components.layout.'. A 'go to parent' link is on the right. Below this is a section titled 'Child Components' which contains a 'Content' dropdown menu set to 'PLM Project Navigator' and a 'go to component' link. This 'Child Components' section is highlighted with a red rectangle.

4. There are two sections in the PLM Project Navigator Properties: Active Projects and Current Project. Fill out the fields in both of these sections appropriately.

Active Projects Flipper

Expand the Active Project flipper and fill out the fields in this section:

Properties

Configuration
Web UI style

---[MAIN]---
Save
Close
New...
Delete
Rename
Save as...

PLM Project Navigator Properties
[go to parent](#)

▼ Active Projects

Active Projects Label
i18n.stibo.plm.webui.navigator.ActiveProjects.Label

*Active Projects Root
step://classification?id=PLMProductClassificationsRoot
...

*Project Object Type
PLMDesignSpecification
...

*Number Of Display Levels
2

Project Name Attribute
... Clear

► Current Project

- **Active Projects Label:** Enter in the name for the folder that will hold active projects.
- **Active Projects Root:** Select the root for all active project. This folder is used for the different classification groupings.
- **Project Object Type:** The object type selected to display under the selected active project root. In this example, the 'PLMDesignSpecification' object type is used.
- **Number of Display Levels:** The level of folders up from the selected 'Product Object Type' that will be displayed in the classification structure when users browse to find projects..
- **Project Name Attribute:** Select an attribute to display a different name for the active project. This displays instead of the 'Project Object Type' name.

Current Project Flipper

Expand the Current Project flipper and fill out the fields in this section:

Properties

Configuration
Web UI style

---[MAIN]---
Save
Close
New...
Delete
Rename
Save as...

Active Projects

Current Project

Groups

Related Group ([PLMRelatedSpecificatio
Related Group ([PLMRelatedSpecificatio
Related Group ([PLMRelatedSpecificatio
Related Group ([PLMRelatedSpecificatio

Add...
Edit...
Remove
Up
Down

Group Supplier Responses By Supplier
☒

Related Supplier Responses (PLMSampl
Related Supplier Responses (PLMLabelV
Related Supplier Responses (PLMPackag

Add...
Edit...
Remove
Up
Down

No Suppliers Group Name

i18n.stibo.plm.webui.navigator.CurrentPi

Related Groups Attributes

Add...
Remove
Up
Down

Groups

This field is used to configure the groups you want to display in the active projects classification folder. For example, you could have groupings for specifications, labels, and packaging.

Properties

Configuration
Web UI style

---[HOMEPAGE]---
Save
Close
New...
Delete
Rename
Save as...

▶ Active Projects
▼ Current Project

Groups

Related Group ([PLMRelatedSpecification] / [PLMRecipeNumber] / Recipe /
Related Group ([PLMRelatedSpecification] / Labels / [PLMDesignSpecificatic
Related Group ([PLMRelatedSpecification] / Packaging / [PLMDesignSpecific
Related Group ([PLMRelatedSpecification] / Specifications / [PLMDesignSpe

Add...
Edit...
Remove
Up
Down

Group Supplier Responses By Supplier
☒

Related Supplier Responses (PLMSample / PLMSuppliersName / PLMSuppli
Related Supplier Responses (PLMLabelVariantSample / PLMSuppliersName
Related Supplier Responses (PLMPackagingVariantSample / PLMSuppliersN

Add...
Edit...
Remove
Up
Down

No Suppliers Group Name

i18n.stibo.plm.webui.navigator.CurrentProject.NoSuppliersGroup

Related Groups Attributes

Add...
Remove
Up
Down

To configure this field, click **Add**. Configure the following fields as necessary:

Edit component

Related Group Properties

Component Description

The Related Group component allows you to define groups and the object types you wish to display within each group.

*Group Label

Recipe

*Reference Types

PLMRelatedSpecification

Add... Remove Up Down

*Target Object Types

PLMDesignSpecificationVariant

Add... Remove Up Down

Related Groups Attributes

PLMRecipeNumber

Add... Remove Up Down

Save

Cancel

Group Label: Enter the display name for the group.

Reference Types: Select the reference type needed for the group. The reference needed are from the project (for example design specifications), to the specifications in the project.

Target Object Type: Select the object types to display within each group.

Related Group Attributes: Select the attributes to display when a user hovers over the related group node.

© Stibo Systems - Confidential - Release 9.3-MP3 (April 2020)

212

Group Supplier Responses By Supplier

When selected, this will group supplier samples by each supplier instead of in one main grouping. The name of the supplier groupings is dependent on the attribute selected when configuring the Related Supplier Responses field.

Note: If unselected, then the following fields are disabled: Supplier Name, Related Supplier Responses, and Supplier Response Display Name.

Properties

Configuration
Web UI style

---[HOMEPAGE]---
Save
Close
New...
Delete
Rename
Save as...

▶ Active Projects

▼ Current Project

Groups

Related Group ([PLMRelatedSpecification] / [PLMRecipeNumber] / Recipe /
Related Group ([PLMRelatedSpecification] / Labels / [PLMDesignSpecificatic
Related Group ([PLMRelatedSpecification] / Packaging / [PLMDesignSpecific
Related Group ([PLMRelatedSpecification] / Specifications / [PLMDesignSpe

Add...
Edit...
Remove
Up
Down

Group Supplier Responses By Supplier

Related Supplier Responses (PLMSample / PLMSuppliersName / PLMSuppli
Related Supplier Responses (PLMLabelVariantSample / PLMSuppliersName
Related Supplier Responses (PLMPackagingVariantSample / PLMSuppliersN

Add...
Edit...
Remove
Up
Down

No Suppliers Group Name

i18n.stibo.plm.webui.navigator.CurrentProject.NoSuppliersGroup

Related Groups Attributes

Add...
Remove
Up
Down

Related Supplier Responses

This field is used to configure the sample object types to display under each supplier. In this example, the object types are sample, packaging sample, or label sample.

Properties

Configuration
Web UI style

---[HOMEPAGE]---
Save
Close
New...
Delete
Rename
Save as...

▶ Active Projects
▼ Current Project

Groups

Related Group ([PLMRelatedSpecification] / [PLMRecipeNumber] / Recipe /
Related Group ([PLMRelatedSpecification] / Labels / [PLMDesignSpecificatic
Related Group ([PLMRelatedSpecification] / Packaging / [PLMDesignSpecific
Related Group ([PLMRelatedSpecification] / Specifications / [PLMDesignSpe

Add...
Edit...
Remove
Up
Down

Group Supplier Responses By Supplier
☒

Related Supplier Responses

Related Supplier Responses (PLMSample / PLMSuppliersName / PLMSuppli
Related Supplier Responses (PLMLabelVariantSample / PLMSuppliersName
Related Supplier Responses (PLMPackagingVariantSample / PLMSuppliersN

Add...
Edit...
Remove
Up
Down

No Suppliers Group Name
i18n.stibo.plm.webui.navigator.CurrentProject.NoSuppliersGroup

Related Groups Attributes

Add...
Remove
Up
Down

To configure this field, click **Add** and configure the following fields as necessary.

Edit component

Related Supplier Responses Properties

Component Description

The Related Supplier Responses component allows you to define which supplier response object types you wish to display under the supplier groups.

Name Attribute

... Clear

*Reference Type

...

*Target Object Type

...

Related Object Types

Add... Edit... Remove Up Down

Related Groups Attributes

Add... Remove Up Down

Supplier Name Attribute

... Clear

Save

Cancel

Attribute Name: Select this attribute to display a different attribute name instead of the Target Object Type ID.

Reference Type: Select the appropriate reference type for the group. This reference type is acting as the source which must be the project (for example, design specification).

Target Object Type: Select the object types to display within the group.

Related Object Types: Select other object types to further break down what was selected for the target object type.

Related Groups Attributes: Select the attribute to display when a user hovers over the related groups node.

Supplier Name Attribute: Select the attribute to use as the display name for the supplier.

© Stibo Systems - Confidential - Release 9.3-MP3 (April 2020)

215

Note: The Supplier Name Attribute is also the name used when grouping by supplier responses.

Packaging variant or packaging sample can have nested levels in the PLM data model. To configure nesting, once either packaging variant or packaging sample is configured, double click on it, and configure the dialog that displays with the necessary nested level. Continue to do this for each necessary nested level.

Properties (edited)

Configuration Web UI style

---[MAIN]---

Save Close New... Delete Rename Save as...

▼ Current Project

Group Supplier
Responses By Supplier ☐

Related Supplier
Responses

No Supplier
Name

Related Gro
Attributes

Related Object Types

Related Supplier Responses (PLMSample / PLMSuppliersI ^

Related Supplier Responses (PLMLabelVariantSample / PL

Related Supplier Responses (PLMPackagingVariantSampl

Edit component
Related Supplier Responses Properties

Name Attribute ... Clear

*Reference Type ...

*Target Object Type ...

Related Objects (PLMSupplierResponsePackagingElement / PLM

Edit component
Related Objects Properties

Name Attribute ... Clear

*Reference Type ...

*Target Object Type ...

Related Object Types

Add... Edit... Remove Up Down

Related Groups
Attributes

Add... Remove Up Down

✓ Save ✕ Cancel

No Suppliers Group Name

A customer may receive supplier responses but not know which supplier sent the response. If this is the case, these responses need to be grouped together under one folder. This field provides a name to the folder to hold unclaimed supplier responses. The default title for this folder is called 'No Supplier.'

Properties

Configuration

Web UI style

---[MAIN]---

Save

Close

New...

Delete

Rename

Save as...

▶ Active Projects

▼ Current Project

Groups

Related Group ([PLMRelatedSpecification] / [

Related Group ([PLMRelatedSpecification] / [

Related Group ([PLMRelatedSpecification] / [

Related Group ([PLMRelatedSpecification] / [

Add...

Edit...

Remove

Up

Down

Group Supplier Responses By Supplier

☒

Related Supplier Responses

Related Supplier Responses (PLMSample / PI

Related Supplier Responses (PLMLabelVariar

Related Supplier Responses (PLMPackagingV

Add...

Edit...

Remove

Up

Down

No Suppliers Group Name

i18n.stibo.plm.webui.navigator.CurrentProjec

Related Groups Attributes

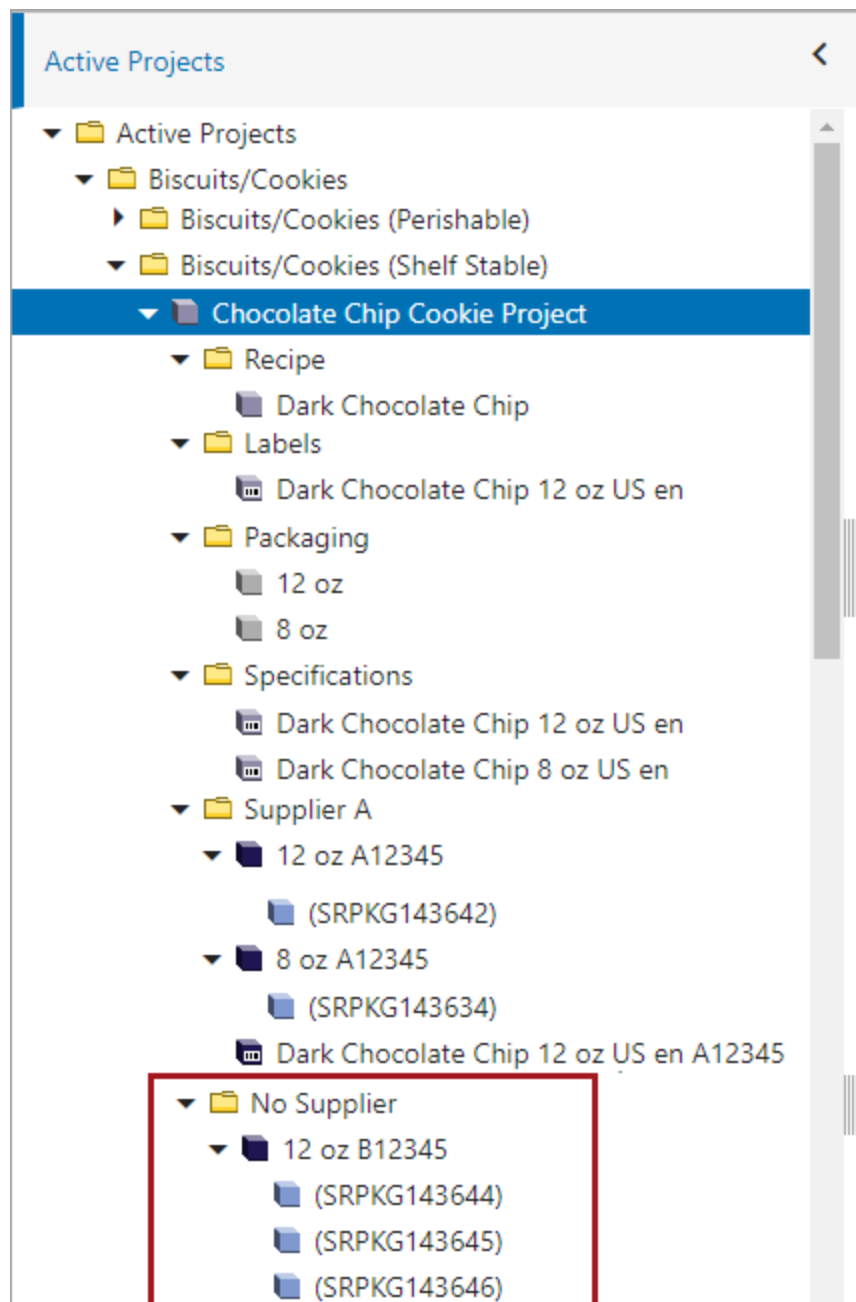
Add...

Remove

Up

Down

For example, if a supplier responded, but their name was not associated with the response, their response would appear in the 'No Suppliers' group name folder.



Related Groups Attributes

Select the attributes to display when a user hovers over the related groups' node.

Properties

Configuration Web UI style

---[MAIN]---

Save Close New... Delete Rename Save as...

grouped by supplier.

► Active Projects

▼ Current Project

Groups

Related Group ([PLMRelatedSpecification] / [

Related Group ([PLMRelatedSpecification] / [

Related Group ([PLMRelatedSpecification] / [

Related Group ([PLMRelatedSpecification] / [

Add... Edit... Remove Up Down

Group Supplier Responses By Supplier ☒

Related Supplier Responses

Related Supplier Responses (PLMSample / PI

Related Supplier Responses (PLMLabelVariar

Related Supplier Responses (PLMPackagingV

Add... Edit... Remove Up Down

No Suppliers Group Name

i18n.stibo.plm.webui.navigator.CurrentProjec

Related Groups Attributes

Add... Remove Up Down

Once configured, map to the needed screen. For more on mapping, see the **Mappings** topic in the **Using a Web UI** section of the **Web User Interfaces / Web UI Setup and User Guide**.

For more on how to use project navigation in Web UI, see the **Using Project Navigator** topic in the **PLM for Users** section of the **Product Lifecycle Management** documentation.

Storyboard Setup

Storyboards allows users to create idea concepts for early product development. When setup properly, a user is able to interact with storyboards, and upload content of various object types. To set up this interface for users, follow the directions in this documentation.

Important: It is recommended that admin users read through the **PLM for Users** documentation before starting setup. It is important to understand how PLM works prior to beginning any configuration. Additionally, it is important to be familiar with standard STEP functionality.

All of the setup described in this documentation is done in the STEP Workbench. If the steps are not followed (including naming conventions, except where notated), then PLM will not function as expected.

Users of PLM will not see the workbench, and will only interact with PLM through the web. Proper workbench setup is vital for the user having a seamless experience while working.

The following licenses and components are needed for storyboards to work:

Required License	Components needed
X.STEPIllustrator Note: Only needed if also utilizing the ability to integrate with Adobe Illustrator	illustrator Note: Only needed if also utilizing the ability to integrate with Adobe Illustrator
N/A	plm-ui-rfa

Configurations, Object Types, and Business Rules

All of the setup described below is done in the STEP Workbench. If the steps are not followed (including naming conventions, except where notated), then PLM will not function as expected.

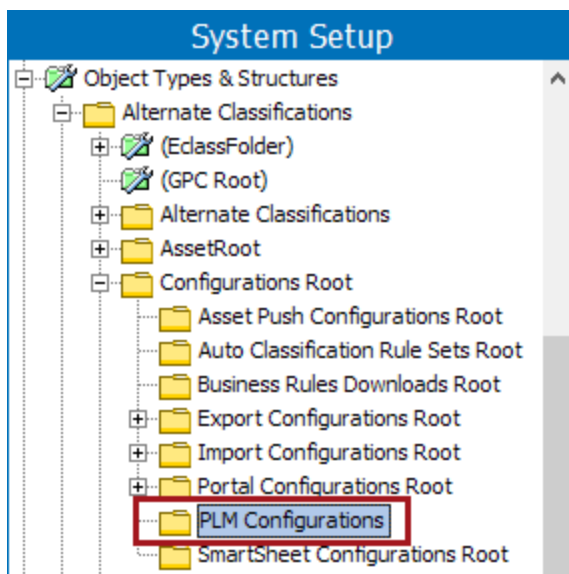
Below are directions on how to get started implementing PLM.

PLM Configurations Setup

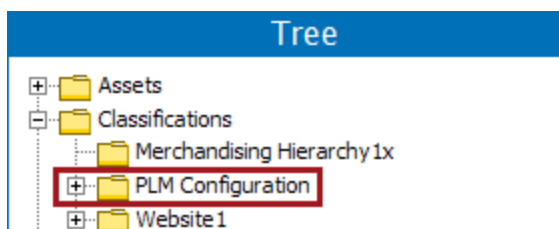
Follow the steps below to upload the configuration files.

Important: Talk to your implementation team before uploading the provided configuration files and configuring anything else for PLM.

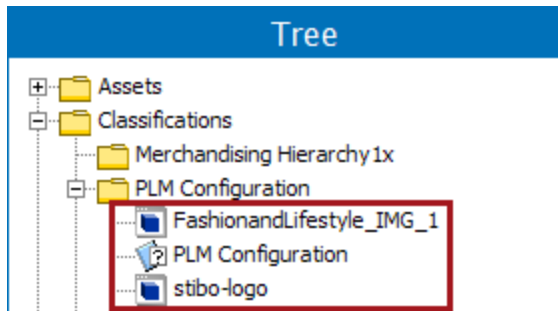
1. In **System Setup**, go to Object Types & Structures > Alternate Classifications > Configurations Root > and create a folder to house PLM configurations.



2. Next, in **Tree**, below the classifications node, create a new classification folder to house the PLMConfiguration.



3. Into this folder, import the background login screen image, the PLM configuration file, and the desired logo using file names as STEP IDs. Make sure that all changes that are needed to be made to the PLM configuration file are made before the file is uploaded. This includes changing any IDs necessary; otherwise, PLM will not work properly.



Note: Depending on your implementation team, these steps may or may not be handled by them. If the implementation team does not handle the steps above, they will be the ones to provide the configuration files. Additionally, both the background image on the login screen and the logo are configurable.

Creating PLM Storyboard Object Types

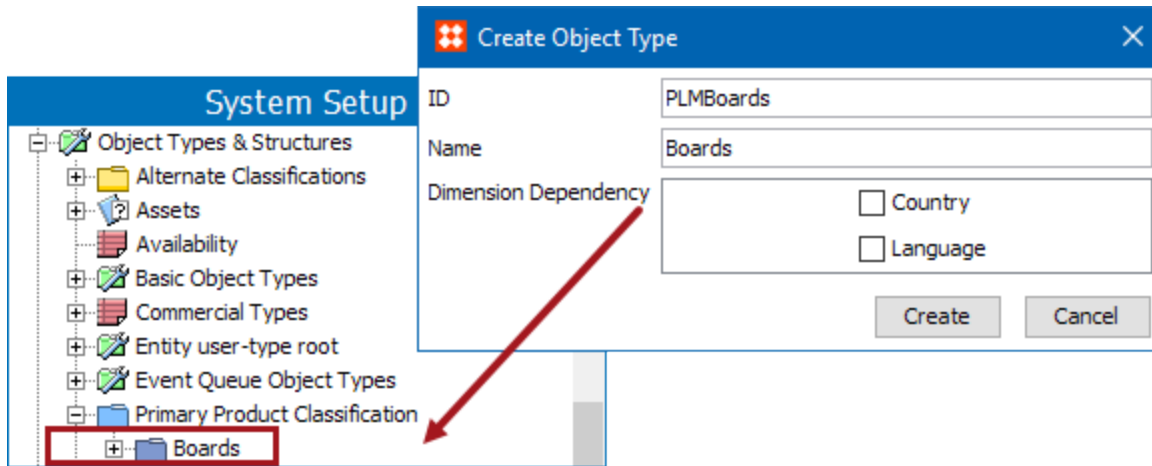
This section describes how to create object types that are needed for the PLM setup. While the object types are different, the steps to create these object types are all the same. The following object types will need to be created:

- PLMBoards
 - PLMBoard
 - PLMBoardCollectionsRoot
 - PLMBoardCollection
- PLMColorsRoot
 - PLMColorType
 - PLMColor
- PLMMaterialsRoot
 - PLMFabricRoot
 - PLMLabelRoot
 - PLMLabelSubType
 - PLMPackagingRoot
 - PLMShippingRoot
 - PLMTrimRoot
 - PLMTrimSubType
- PLMProducts
 - PLMProduct

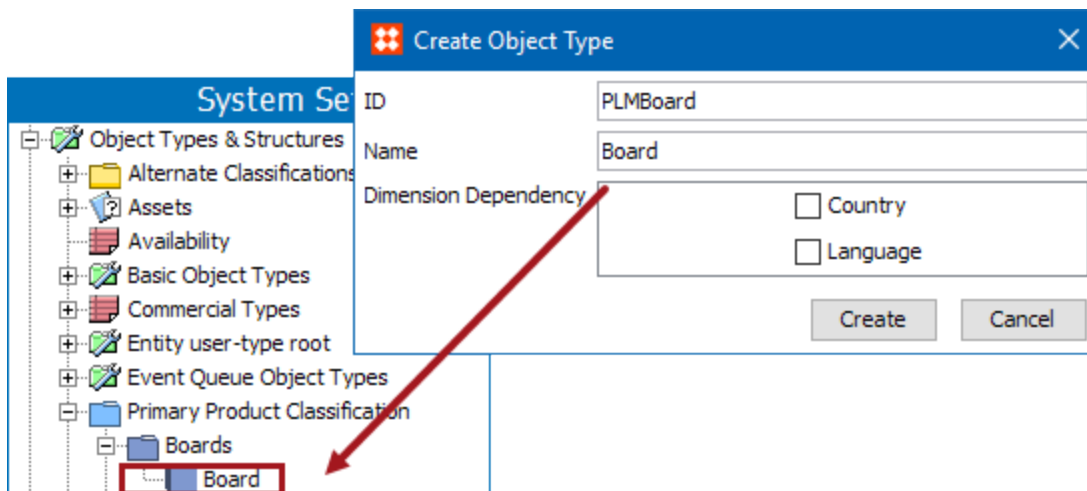
Note: The IDs / names of the above object types can be changed according to company needs. Notify the implementation team of any changes, so updates can be made to the uploaded configuration documents.

In the example below, the 'Boards' and 'Board' object types are used.

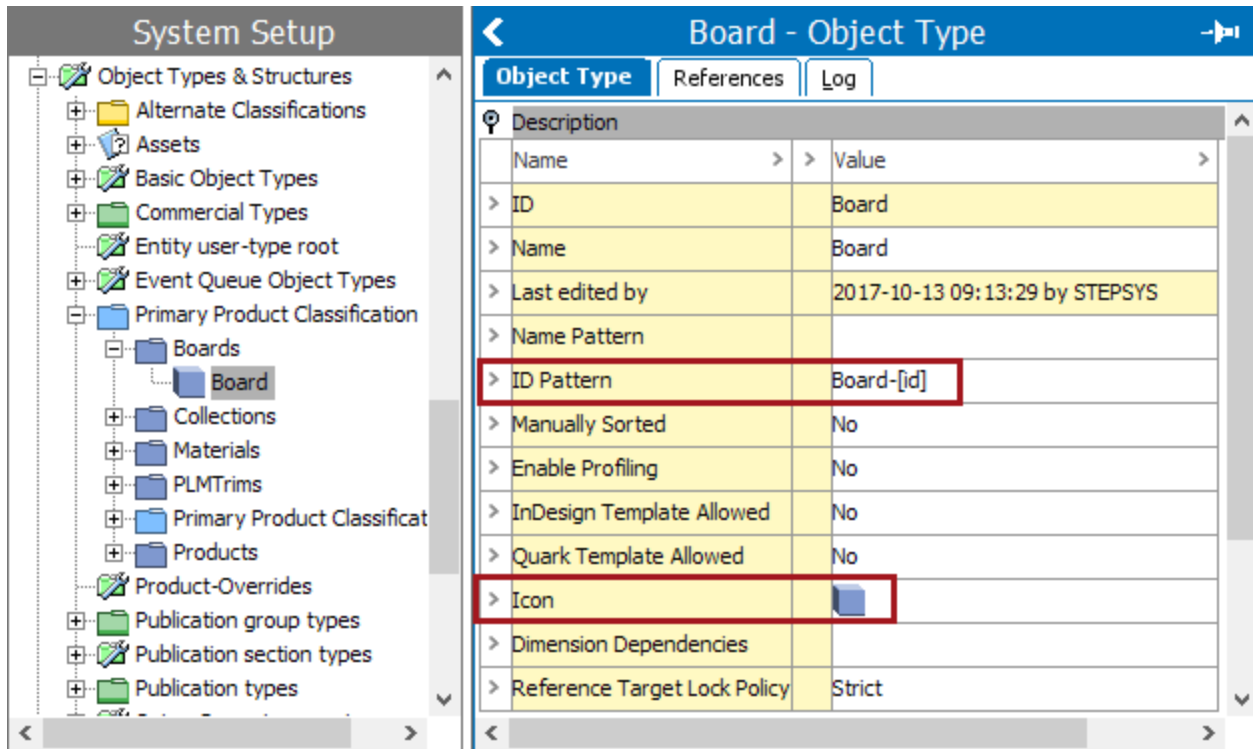
1. In **System Setup** under Object Types & Structures > Primary Product Classification> create a new product object type called **Boards** with the ID of **PLMBoards**.



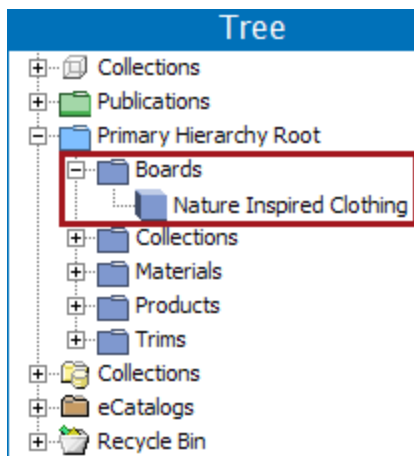
2. Below the new Boards object type create a child called object type **Board** with the ID **PLMBoard**.



3. Make sure that the object type Board has an auto ID of **Board-[id]** in the ID Pattern field, and pick an appropriate icon. For more on how to create an auto ID or how to add icons, see **Autogenerate Using Name Pattern and ID Pattern** topic in the **Object Types and Structures** section of the **System Setup / Super User Guide** documentation.



4. In **Tree**, under the Primary Product Hierarchy, make a folder to hold the **Boards** object type created in PLM.



Adding a Hero Image Business Rule

For users of PLM, the 'hero' image is the feature image of the storyboard. This is the image that is displayed at the top of the storyboard, and is the one that represents the storyboard when being viewed in the board gallery.

In order to make the hero image function properly, a business rule needs to be created and applied in System Setup > Global Business Rules > Actions. It should have the ID **PLMEnsureHero Image**.

System Setup

- Attribute Groups
- Attribute Transformations
- Action Sets
- Contexts
- InDesign Queue
- Lists of Values / LOVs
- Completeness Metrics
- Global Business Rules
 - Actions
 - PLMEnsureHeroImage**
 - Conditions
 - Libraries
- Inbound Integration Endpoints
- Match Codes and Matching Algorithms
- Outbound Integration Endpoints
- Web UIs
- Workflow Profiles
- Workflows
- Derived Events
- Table

SpirePLMEnsureHeroImage rev.0.1 - Business Rule

Business Rule
Usage
Statistics
Log
Status

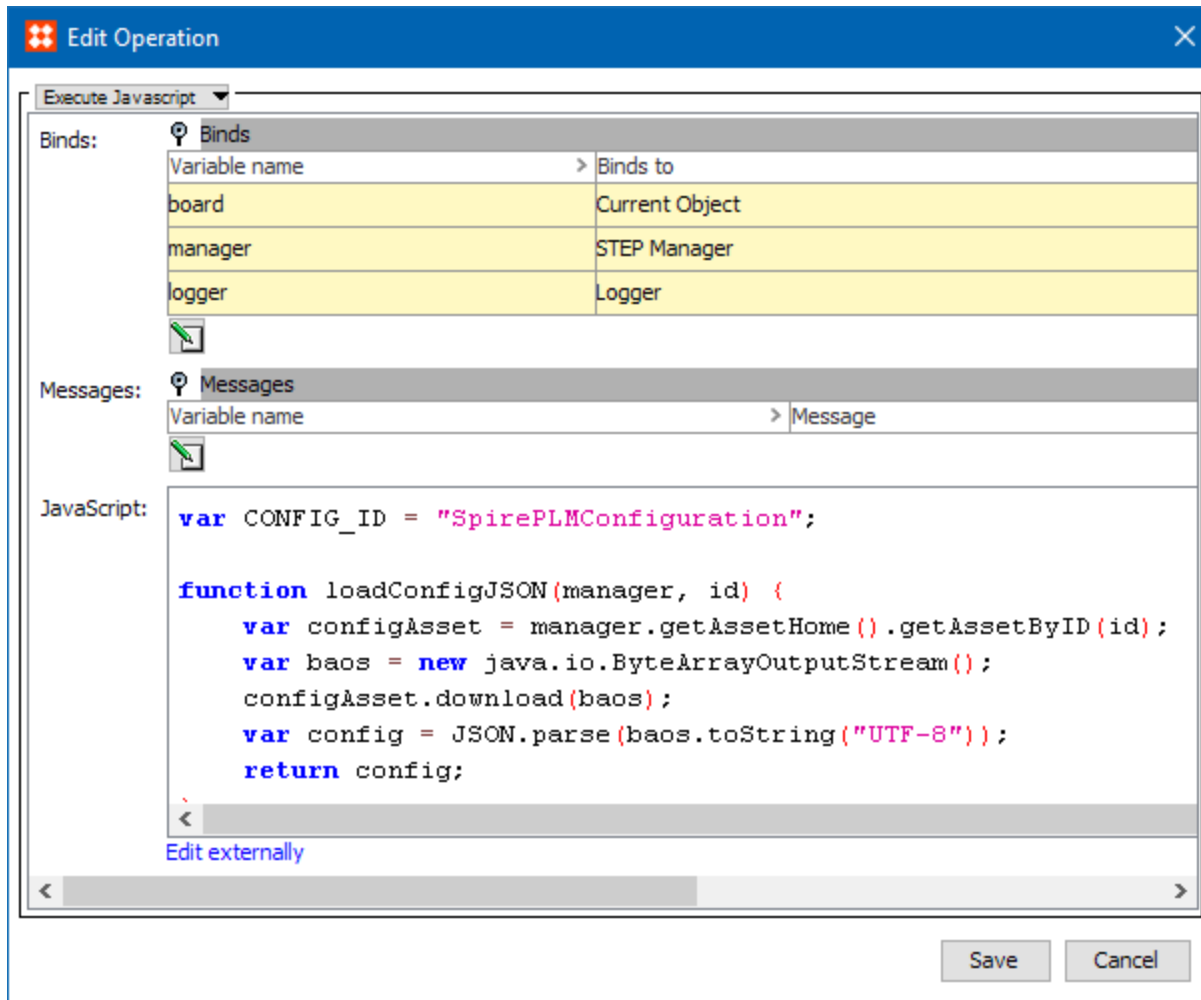
Name	Value
ID	PLMEnsureHeroImage
Name	PLM Ensure Hero Image
Revision	0.3 Last edited by STEPSYS on Fri Feb 23 17:14:00 UTC 2018
Description	
Type	Action
Valid Object Types	No object types valid
On Approve	Not Executed
Scope	Global
Run as privileged	☐

Operations
Dependencies
Applies if

JavaScriptBusinessActionWithBinds: Bindings, 0 messages, var CONFIG_ID = "Project11...

[Edit Business Rule](#)

Below is a picture of the business rule configurations for the hero image:



Important: The following script is only an example and should not be used as-is without thorough testing, including updating to match object and link types that exist on your system.

```
var CONFIG_ID = "SpirePLMConfiguration";

function loadConfigJSON(manager, id) {
    var configAsset = manager.getAssetHome().getAssetByID(id);
    var baos = new java.io.ByteArrayOutputStream();
    configAsset.download(baos);
    var config = JSON.parse(baos.toString("UTF-8"));
    return config;
}

function removeDuplicates(list) {
    return list.filter(function(item, pos) {
        return list.indexOf(item) === pos;
    });
}
```

```

}

function lookupReferenceType(manager, id) {
return manager.getReferenceTypeHome().getReferenceTypeByID(id);
}

var config = loadConfigJSON(manager, CONFIG_ID);
var contentModel = config.applications.spireplm.model.boards.boardContent;
var allContentReferenceTypeIds = [
contentModel.image.boardReferenceId,
contentModel.trim.boardReferenceId,
contentModel.material.boardReferenceId,
contentModel.product.boardReferenceId
];
var allHeroImageReferenceTypeIds = [
contentModel.image.heroImageReferenceId,
contentModel.trim.heroImageReferenceId,
contentModel.material.heroImageReferenceId,
contentModel.product.heroImageReferenceId
];
logger.info("content types ids" + JSON.stringify(allContentReferenceTypeIds));
var contentReferenceTypeIds = removeDuplicates(allContentReferenceTypeIds);
var heroImageReferenceTypeIds = removeDuplicates(allHeroImageReferenceTypeIds);
logger.info("content types ids" + JSON.stringify(contentReferenceTypeIds));
logger.info("hero types ids" + JSON.stringify(heroImageReferenceTypeIds));
var contentReferenceTypes = contentReferenceTypeIds.map(function (id) { return
lookupReferenceType(manager, id); });
var heroImageReferenceTypes = heroImageReferenceTypeIds.map(function (id) {
return lookupReferenceType(manager, id); });
function getReferences(source, referenceTypes) {
var result = new java.util.ArrayList();
for (var i = 0; i < referenceTypes.length; i++) {
var refs = source.getReferences(referenceTypes[i]);
result.addAll(refs);
}
return result;
}

function getHeroImageReferenceTypeFor(reference) {

```

```

var refTypeId = reference.getReferenceType().getID();
var idx = -1;
for (var i = 0; i < allContentReferenceTypeIds.length; i++) {
    if (allContentReferenceTypeIds[i] == refTypeId) {
        idx = i;
        break;
    }
}

logger.info("Index of " + refTypeId + " is " + idx);
return heroImageReferenceTypeIds[idx];
}

var heroImageReferences = getReferences(board, heroImageReferenceTypes);
if (heroImageReferences.size() > 1) logger.warning("Board has more than one
hero image");
if (heroImageReferences.size() === 0) {
    logger.info("Board has no hero images");
    var contentReferences = getReferences(board, contentReferenceTypes);
    if (contentReferences.size() > 0) {
        var anyReference = contentReferences.get(0);
        var refTypeId = getHeroImageReferenceTypeFor(anyReference);
        var anyContent = anyReference.getTarget();
        board.createReference(anyReference.getTarget(), refTypeId);
        logger.info("Created hero image reference " + refTypeId + " to " +
anyContent.getTitle());
    }
} else {
    var contentReferences = getReferences(board, contentReferenceTypes);
    if (contentReferences.size() === 0) {
        logger.info("Has hero image, but no content. Going to remove hero images");
        for (var i = 0; i < heroImageReferences.size(); i++) {
            var ref = heroImageReferences.get(i);
            logger.info("Deleting reference to " + ref.getTarget().getTitle());
            ref.delete();
        }
    } else {
        for (var i = 0; i < heroImageReferences.size(); i++) {

```

```

var heroImage = heroImageReferences.get(i).getTarget();
var found = false;
for (var j = 0; j < contentReferences.size(); j++) {
var content = contentReferences.get(j).getTarget();
if (heroImage.equals(content)) {
found = true;
break;
}
}
if (!found) {
logger.info("Hero image is not content. Deleting reference to " +
heroImageReferences.get(i).getTarget().getTitle());
heroImageReferences.get(i).delete();
if (i === 0) {
var anyReference = contentReferences.get(0);
var refTypeId = getHeroImageReferenceTypeFor(anyReference);
var anyContent = anyReference.getTarget();
board.createReference(anyReference.getTarget(), refTypeId);
logger.info("Created hero image reference " + refTypeId + " to " +
anyContent.getTitle());
}
}
}
}
}
}
}

```

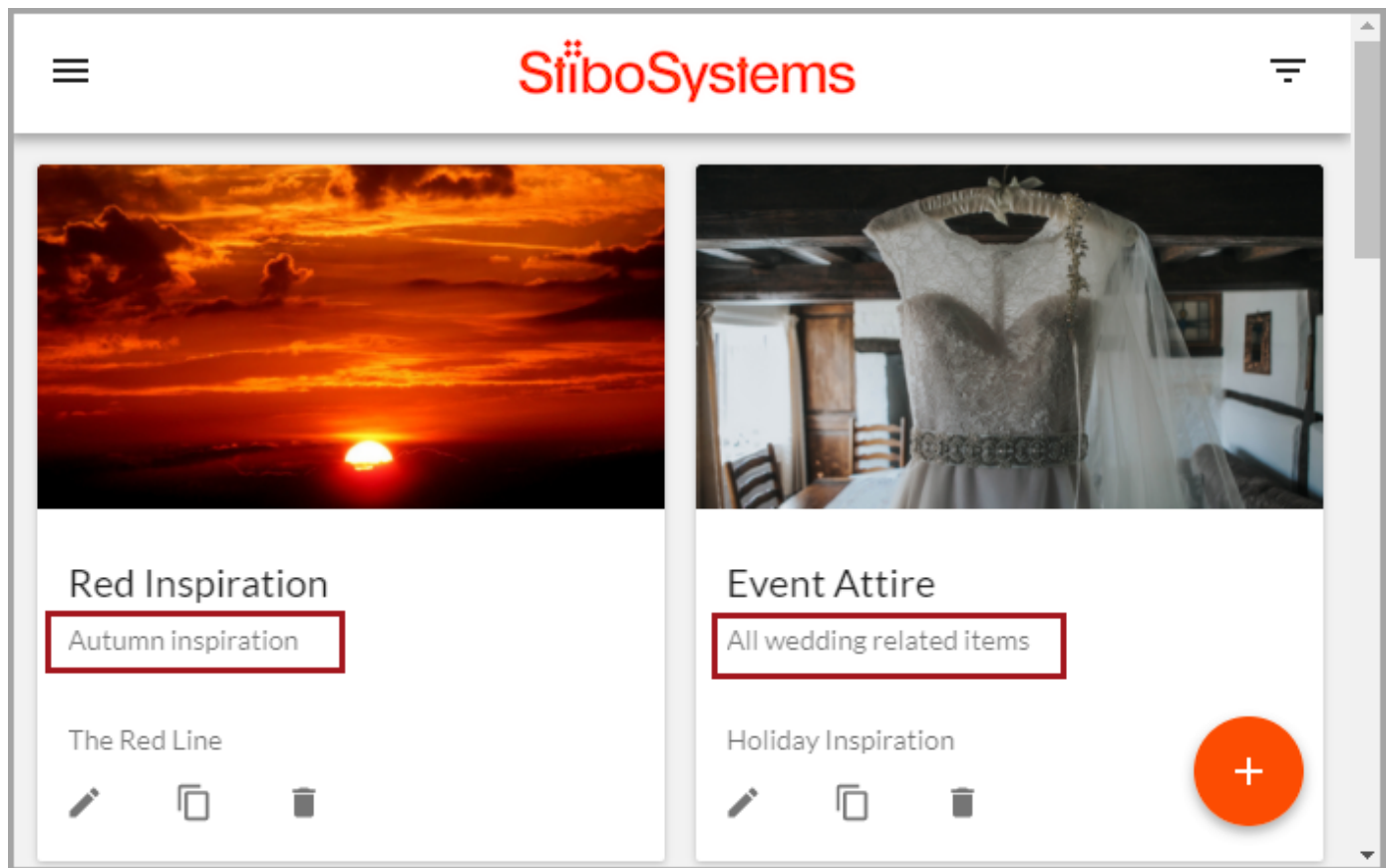
To learn more about business rules, see the **Business Rules** topic in the **Business Rules** documentation.

Setting Up Storyboard Attributes

In order for the end user to create data for storyboards, attributes need to be created.

Storyboard Attributes

When creating attributes for storyboards, it is important to keep in mind that the 'Description' attribute is the only attribute that will show on the storyboard when in the board gallery. This attribute ID is used in the uploaded PLM configuration documents, so if the ID needs to be changed, it is important to let the implementation team know; otherwise, this attribute will not work properly.



All other attributes for storyboards will only display on the Create New Board or Edit Board dialogs. See below for how to implement Storyboard attributes.

Create New Board

INFO COLLECTIONS

Name *

Description

Season

CANCEL CREATE

Edit Board

INFO COLLECTIONS

Name *

Fashionable Shoes

Description

Shoes for everyday wear and comfort.

Season

Fall

CANCEL OK

Configuring the Description Attribute

1. In **System Setup**, create a new attribute group to hold all necessary attributes for storyboards called 'Board Essential Attributes.'

System Setup

Attribute Groups

- Basic Attributes
- Board Essential Attributes

Create Attribute Group

ID BoardEssentialAttributes

Name Board Essential Attributes

Create Cancel

2. Next, in the configuration file that is uploaded to STEP, make sure that the correct attribute ID displays in the file under the 'descriptionAttribute' field on the board object. In the example below, the ID 'Description' is used for the 'descriptionAttribute.' It is usually a Text base, Description type attribute that is valid for Boards.

Create Attribute

Steps

1. Select Attribute Group
- 2. Enter ID and Name**
3. Enter Validation Rule
4. Select Unit(s)
5. Valid For
6. Apply Dimension Dependencies
7. Link to Hierarchy

Enter ID and Name

ID

Description

Name

Description

Validation Base Type

Text

Multi Valued

No

☐ Specification
 ☒ Description

Cryptographic Key

<No Encryption>

Back

Next

Finish

Cancel

This attribute is the only attribute that displays on the storyboard itself when viewed in the board gallery.

```

},
"model": {
  "context": "Context1",
  "workspace": "Main",
  "boards": {
    "board": {
      "nodeType": "Product",
      "objectTypeId": "Board",
      "parentId": "Boards",
      "attributeGroupId": "BoardEssentialAttributes",
      "descriptionAttribute": "Description"
    }
  }
}

```



Pre Fall 2018

Brightly colored clothes

Summer 18

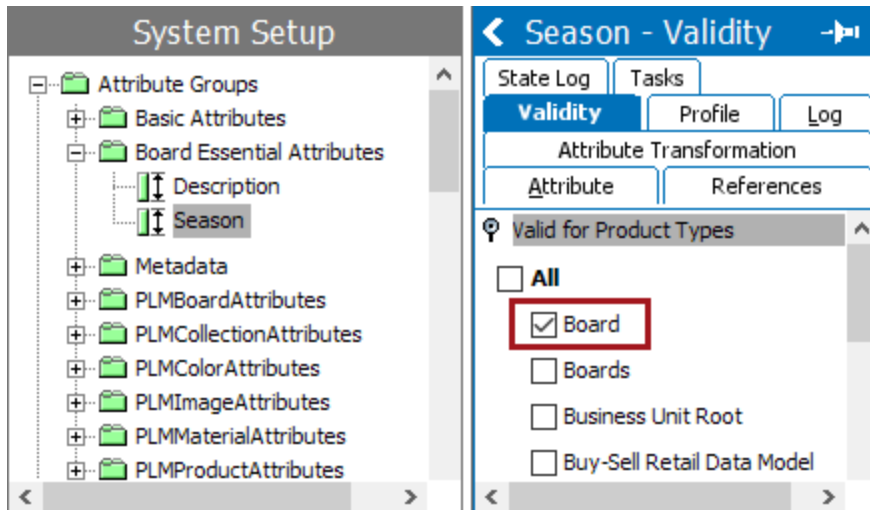




Configuring other storyboard attributes

All other description type attributes that are created and have the 'Board Essential Attributes' folder as their parent will show up on the Create New Board or Edit Board dialog only.

Create any needed description attributes in the 'Board Essential Attributes' folder. Make the attributes valid for the 'Board' object type, and link the attributes to 'Boards'.



System Setup

- Attribute Groups
 - Board Essential Attributes
 - Description
 - Season**

Edit Board

INFO COLLECTIONS

Name *
Fashionable Shoes

Description
Shoes for everyday wear and comfort.

Season
Fall

CANCEL OK

For more information on how to create attributes, see the **Attributes** topic in the **System Setup / Super User Guide** documentation.

Creating Tag Attributes

Applying tags to images, materials, trims, colors, and other object types placed on storyboards in PLM is a useful tool to help users organize and find content easier. Each time a user creates a new tag, it is stored in a List of Values (LOV) that is applied to a multi-valued attribute.

Creating the Tag Multi-Valued Attribute

In order for PLM users to create tags, the proper multi-valued attribute needs to be created first. Follow the steps below to create the attribute.

Creating an LOV for a Tag Attribute

In System Setup (in the appropriate folder set up to house LOVs), create an LOV to hold the tags that users will create when using PLM. The validation base type should be set to 'text' to obtain optimal performance, though it is possible to set the validation base type to the following depending on company needs: number, fraction, number range, numeric text, and numeric text (exclude tags). For more on how to create LOVs, see the **List of Values (LOVs)** topic in the **System Setup / Super User Guide** documentation.

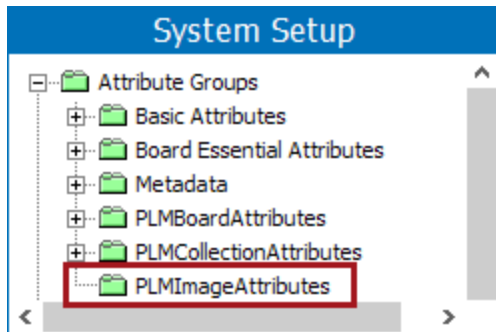
It is important to note that when creating the LOV to hold tags, the value to **Allow Users to Add Values** must be set to 'Yes.' This allows PLM users to add any needed values or 'tags' to content that might not already exist in the system.

Name	Value
ID	PLMTagLOVs
Name	PLMTagLOVs
Edited by	2017-10-18 15:07:45 by STEPSYS
Path	Lists of Values / LOVs/PLMTagLOVs
Dimension Dependencies	

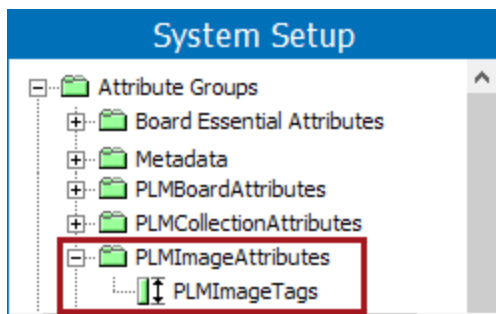
Name	Value
Validation Base Type	Text
Allow Users to Add Values	Yes
Mask	
Minimum Value	N/A
Maximum Value	N/A
Maximum Length	100

Creating a Tag Attribute

1. Go to System Setup and create an attribute group to hold PLM image attributes.



2. In this attribute group, create a multi-valued attribute. See the **Single and Multi-Valued Attributes** topic in the **Attributes** section of the **System Setup / Super User Guide** documentation to learn more on how to create a multi-valued attribute. In this example it is PLMImageTags.



3. When prompted to select an LOV, select the LOV created to hold tags in PLM (created in the previous section). Make it valid for the following object types:

Product Types

- PLMFabricMaterial
- PLMLabelMaterial
- PLMPackagingMaterial
- PLMShippingMaterial
- PLMTrimMaterial

Asset Types

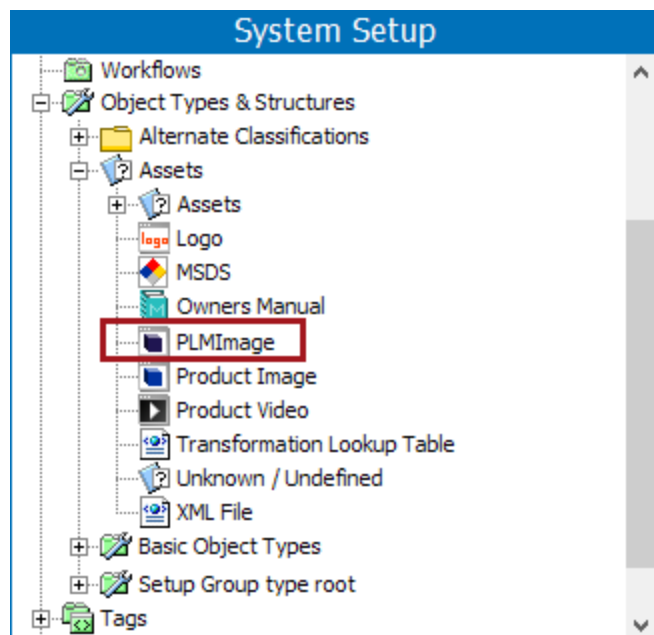
- PLMColorImageAsset

Creating Storyboard Assets

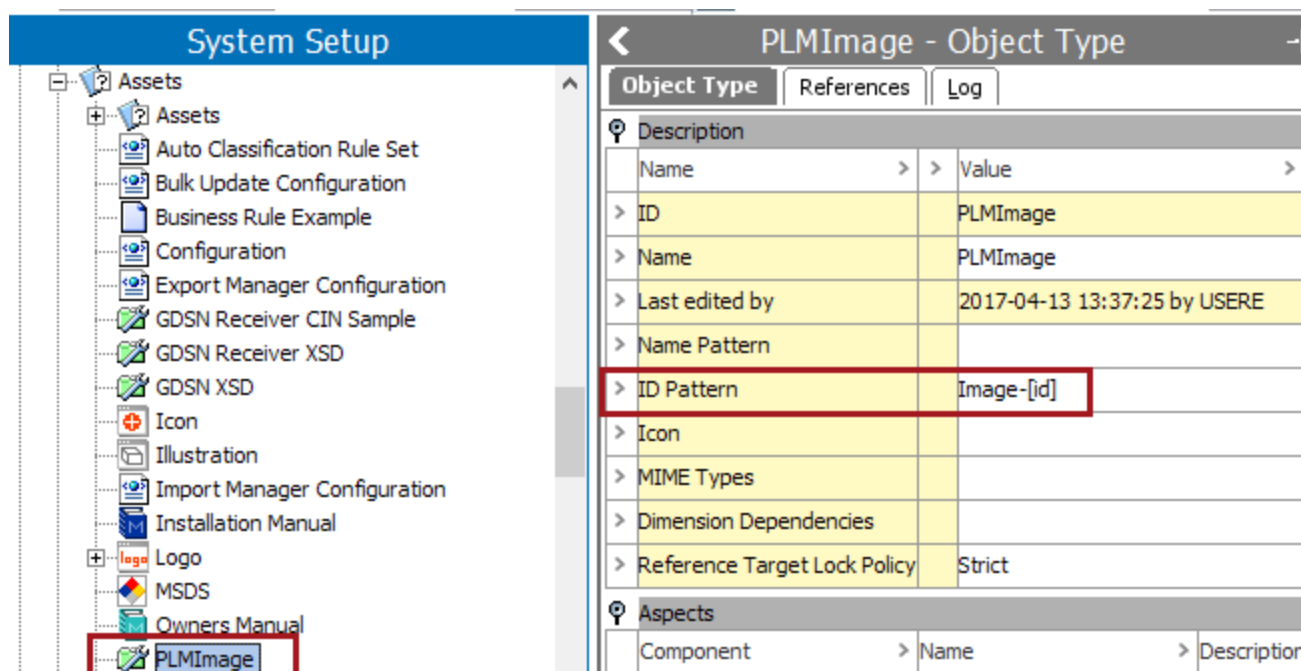
In order for users to import images and colors, assets need to be set up in STEP. All of the setup described in this topic is done in the STEP Workbench.

The following asset object type will need to be created:

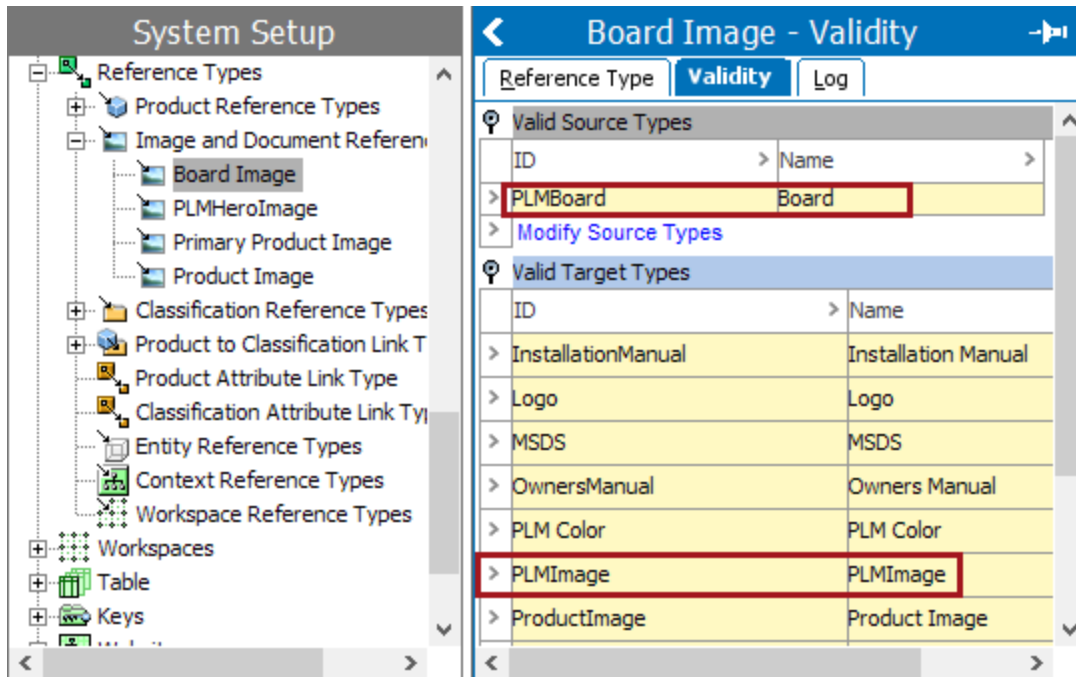
- PLMImage



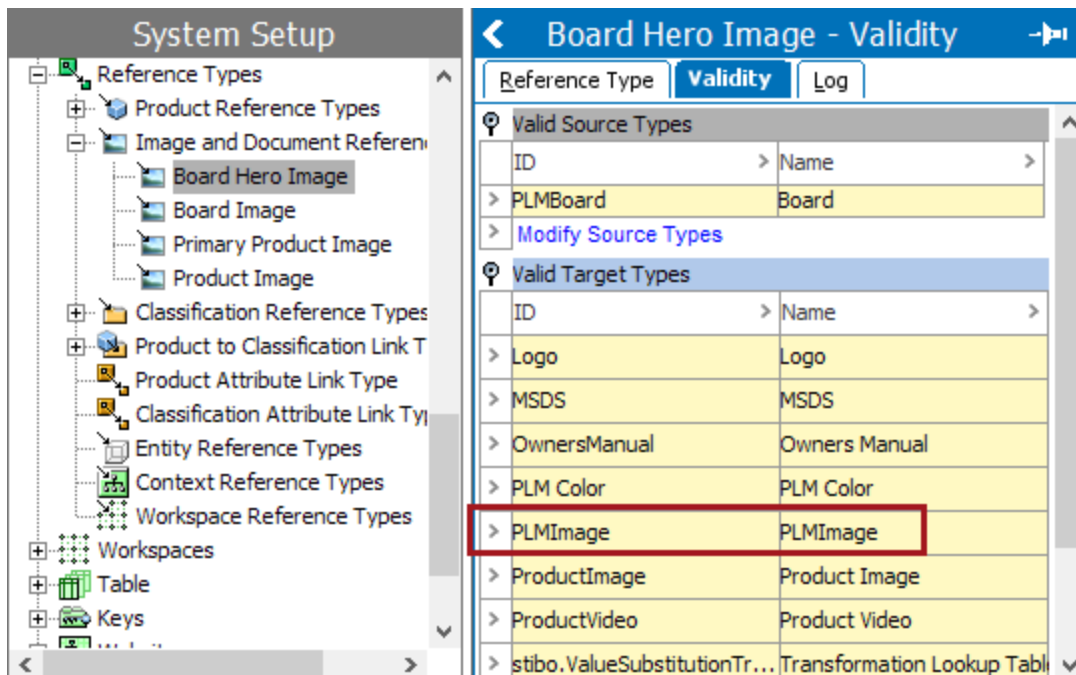
1. In System Setup, go to Object Types and & Structures > Assets > create a new asset object type with **PLMImage** as the ID. Set the ID Pattern field to have the following auto generated ID: **Image-[id]**.



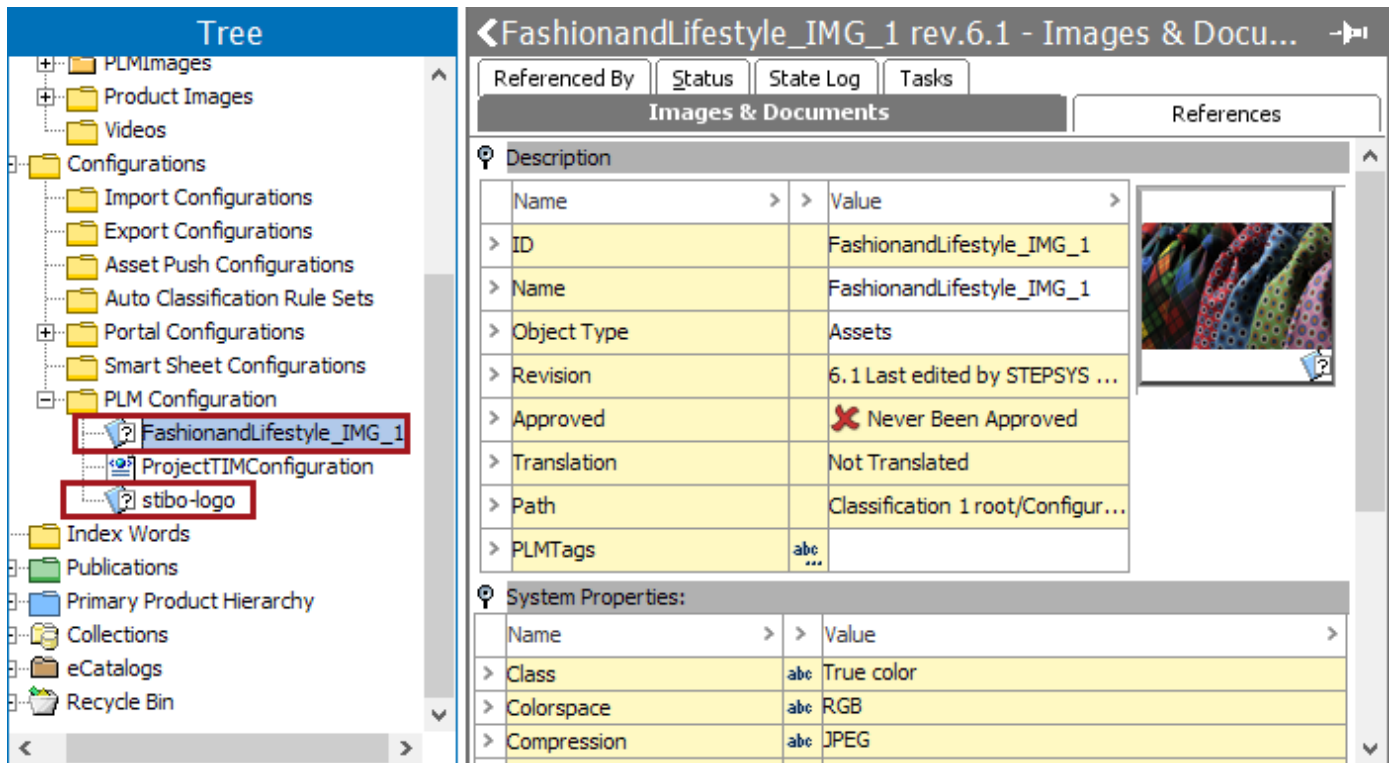
- Next, go to System Setup > Reference Types > Images and Document Reference Types > create a new reference called **PLMBoardImage**. Ensure that **Board** is a valid source type of this reference. **PLMImage** and all other asset object types should be valid targets.



- Also under Image and Document Reference Types, create a new asset cross reference with the ID **PLMHeroImage**. Ensure that **Board** is a valid source type of this reference. **PLMImage** and all other asset object types should be valid targets.



4. Upload the PLM Configuration asset content. Once complete, setup for using assets on storyboards is finished.



The screenshot displays the Stibo Systems interface for managing assets. On the left, a 'Tree' view shows a hierarchy of folders including 'PLMImages', 'Product Images', 'Videos', 'Configurations', and 'PLM Configuration'. Under 'PLM Configuration', two assets are highlighted with red boxes: 'FashionandLifestyle_IMG_1' and 'stibo-logo'. On the right, the 'FashionandLifestyle_IMG_1 rev.6.1 - Images & Docu...' window is open, showing the 'Images & Documents' tab. This tab contains a table with asset details and a 'System Properties' section.

Name	Value
ID	FashionandLifestyle_IMG_1
Name	FashionandLifestyle_IMG_1
Object Type	Assets
Revision	6.1 Last edited by STEPSYS ...
Approved	✗ Never Been Approved
Translation	Not Translated
Path	Classification 1 root/Configur...
PLMTags	abc ...

Name	Value
Class	True color
Colorspace	RGB
Compression	JPEG

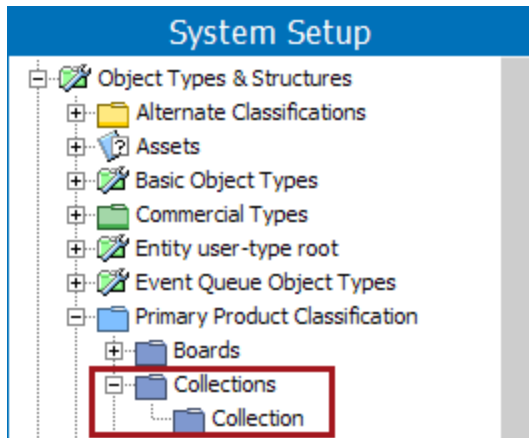
Note: Talk to your implementation team for the PLM Configuration asset content file if you do not already have it on your system.

Creating Storyboard Collection Filters

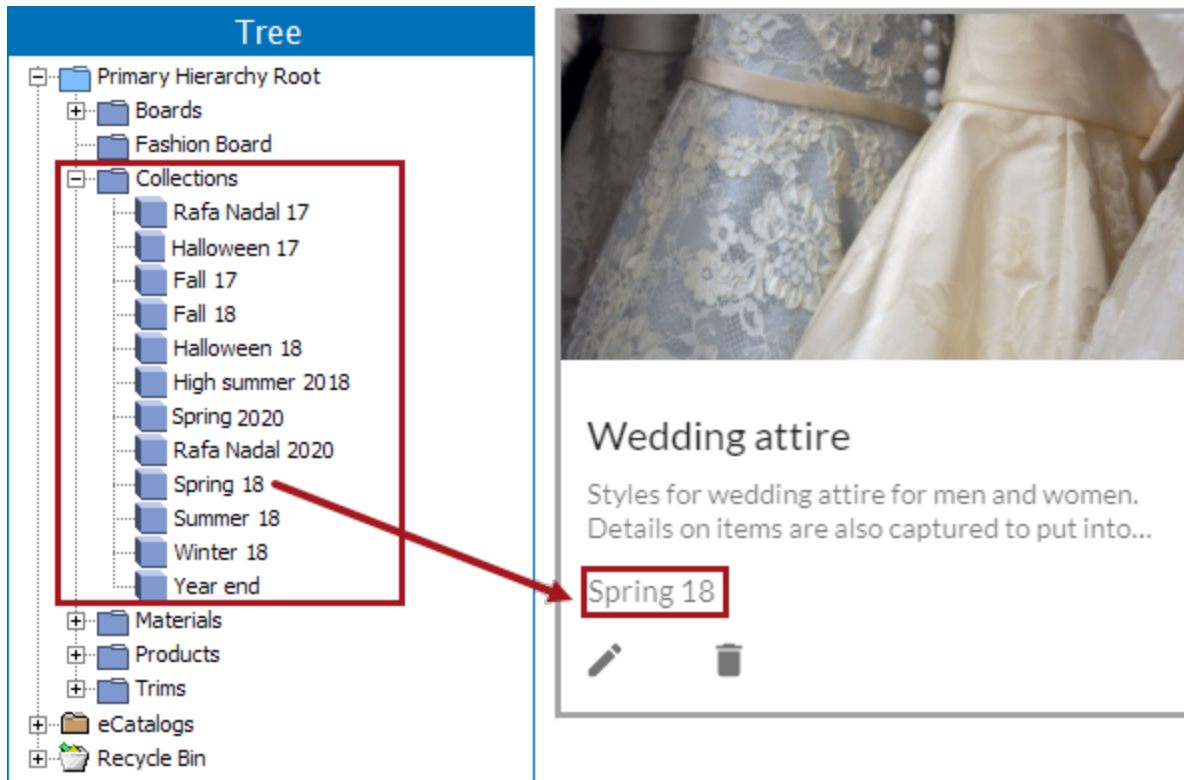
When working with boards in PLM, users can add and filter applied collections to help them find certain storyboards. Collection labels are created on the administration STEP Workbench side ONLY, and are accessed by the user on PLM via the provided web address.

To create collection labels for users:

1. In **System Setup**, go to Object Types & Structures > Primary Product Classifications > create the object type **Collections** with a child called **Collection**.



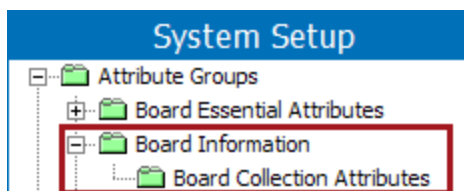
2. Go to **Tree** > Primary Hierarchy Root > add the **Collections** folder and create any needed 'collection children' under this parent folder. The names of these collection children are what will appear on the storyboards in the board gallery on PLM.



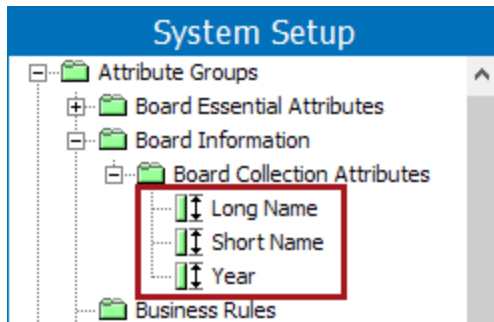
Option to Use Concatenated Attribute Values as Collection Filters

It is sometimes necessary to concatenate attribute values to create new collection filter options. If this is the case, follow the steps below:

1. Go to System Setup > Attribute Groups > and create two folders. One with the ID **PLMBoardInformation** with the name of **Board Information**, and a child folder with the ID **PLMBoardCollectionAttributes** with the name of **Board Collection Attributes**.



2. In the child folder, **PLMBoardCollectionAttributes**, create any needed collection attributes. These need to be created as description attributes, and they need to be made valid for **Board Collections** and **Board Collection** object types. See the **Attributes** topic in the **System Setup / Super User Guide** documentation for more information on how to create attributes and make them valid for certain object types.



- After the attributes are applied to the collection object type, fill in any data needed in the value fields.

Tree

- Assets
- Configurations
- Index Words
- Boards
 - Boards Sandbox
- Collections
- Publications
- Primary Hierarchy Root
 - Boards
 - Fashion Board
 - Collections
 - Fall 17
 - Fall 18**
 - Halloween 18
 - High summer 2018
 - Rafa Nadal 2020
 - Spring 18
 - Summer 18
 - Winter 18
 - Year end

Fall 18 rev.0.4 - Product

Commercial | Tables | Proof View | Status | State Log | Tasks

Product | References | Referenced By | Images & Documents

Name	Value
ID	collection-102577
Name	Fall 18
Object Type	Collection
Revision	0.4 Last edited by STEPSYS on Thu Nov 09 ...
Approved	✗ Never Been Approved
Translation	Not Translated
Path	Primary Hierarchy Root/Collections/Fall 18
Long name	abc Rise & Fall 2018
Year	2017
Short Name	abc
Created By	abc UserL
Created Date	2017-11-09
Description	abc Fall Inspiration

- The values for the attributes in the PLMCollection List Attributes folder are then concatenated in the PLM configuration file.

Note: Talk to your implementation team for the order that you need the attribute concatenation to take place.

- Once concatenated in the PLM configuration file, when a user goes to apply a collection to a storyboard, they only see the full concatenated value when creating or editing a storyboard. When selected, only the collection name appears on the storyboard in the board gallery.

Edit Board

INFO

COLLECTIONS

Summer 18 - Summer 2018 - SU18 - 2018 

High summer 2018 - High summer - HS18 - 2018

Spring 18 - Spring 2018 - SP18 - 2018



OK

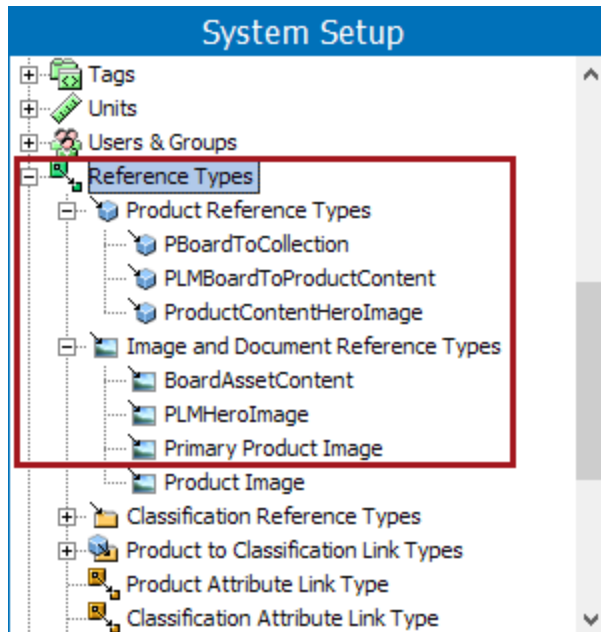
Pre Fall 2018

Summer 18



PLM Reference Types

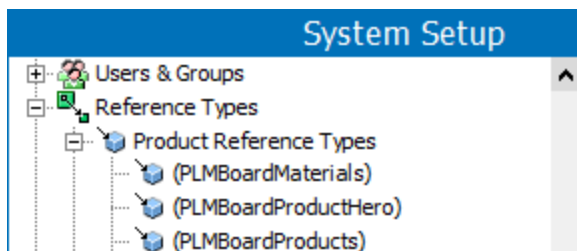
Reference types allow content to appear on storyboards. As part of the setup for PLM, certain **Product Reference Types** and **Image and Document Reference Types** need to be setup to allow PLM to operate correctly for end users.



Below, the necessary reference types are listed:

Product Reference Types

The following product reference types are needed for PLM to function properly. All product reference types listed in this section should have the Board object type as the valid source type. The valid target types are listed within each bullet.



- **PLMBoardMaterials:** PLMColor, PLMOperation, PLMFabricMaterial, PLMLabelMaterial, PLMPackageingMaterial, PLMShippingMaterial, PLMTTrimMaterial
- **PLMBoardProductHero:** PLMDesignerSpecification, PLMFabricMaterial, PLMLabelMaterial, PLMPackageingMaterial, PLMShippingMaterial, PLMTTrimMaterial
- **PLMBoardProducts:** PLMDesignerSpecification, PLMFabricMaterial, PLMLabelMaterial, PLMPackageingMaterial, PLMShippingMaterial, PLMTTrimMaterial

Image and Document Reference Types

The following Image and Document Reference Types are needed for PLM to function properly. Any valid source or valid target types needed are listed below within each bullet.

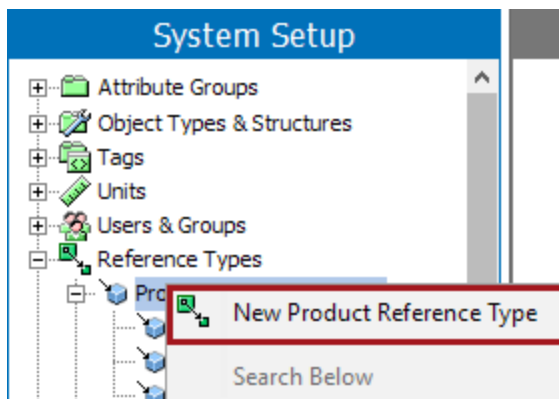
- **BoardAssetContent:** Board object type should be a valid source type; PLMImage object type should be a valid target type.
- **PLMHerolImage:** Board object type should be a valid source type; PLMImage object type should be a valid target type.
- **Primary Product Image:** PLMMaterial, PLMProduct, and PLMTrip object types should be valid source types; ProductImage should be a valid target type.

Note: The names of all mentioned reference types can be changed as long as they are also updated in the configuration documents. Notify your implementation team of any changes.

Creating a Reference Type

To create these needed reference types, follow the steps outlined below. Note that in this example, a Product Reference Type is used, though the steps are the same for creating all new reference types. To learn more about different reference types, see the **Reference and Link Types** topic in the **System Setup / Super User Guide** documentation.

1. Go to System Setup > Reference Types > Product Reference Types > right-click and select **New Product Reference Type**.



2. Continue to work through the 'Create Product Reference Type' wizard, entering in the necessary ID, Name, valid source types, and valid target types. Select **Finish** when complete.

Create Product Reference Type

Steps

1. Enter ID and Name

2. Select valid source types

3. Select valid target types

4. Apply Dimension Dependencies

5. Advanced

Enter ID and Name

ID

PBoardToCollection

Name

PBoardToCollection

Back

Next

Finish

Cancel

© Stibo Systems - Confidential - Release 9.3-MP3 (April 2020)

248

Multi-Reference Editor in PLM

The Multi-Reference Editor in Web UI has two actions unique to PLM:

- PLM Create References Action
- PLM Edit Reference Action

This topic will focus on configuring these actions within a pre-configured Multi-Reference editor.

Prerequisites

This topic assumes that the user has the necessary permissions to access the designer as well as familiarity with the Web UI designer. For more information on the designer, see the **Design Access** topic in the **Web User Interfaces / Web UI Getting Started** documentation. As mentioned, this topic will add actions to an existing Multi-Reference editor. For more information, see the **Multi-Reference Editor** topic in the **Web User Interfaces / Web UI Getting Started** documentation. Finally, this topic also assumes that the user has configured a Reference Metadata Flex Value Header component. see the **Reference Metadata Flex Value Header Component** topic in this documentation.

Configurations

1. In Web UI Designer, from a screen containing a Multi-Reference Editor, navigate to a node list screen > child components > Actions. Click **Add**.

Properties

Configuration
Web UI style

PLM Food Recipe Sa
Save
Close
New...
Delete
Rename
Save as...

Node List Properties
[go to parent](#)

Component Description

The Node List displays objects presented in table or in a grid. Different Display Modes can be applied and customised with a range of headers allowing for different information about the listed objects to be displayed.

Hide Standard Buttons

☐

*ID

Include Labels

☒

Lookup Screen Type For Navigation

☒

Page Size

Use Details Overlay

☐

Child Components

Display Modes

Multi Edit Display Mode

Add..
Remove
Up
Down

Actions

Remove Reference Action

Add..
Remove
Up
Down

- From the 'Add Component' dialog, type 'PLM' into the Filter field below the Action box. Either select **PLM Create References Action** or **PLM Edit Reference Action**, and click Add to configure further. See below

for details.

Add Component

PLM Create References Action

PLM Edit Reference Action

Select a component to see its description

Filter

PLM

☐ Show deprecated components

✓ Add

✕ Cancel

PLM Create References Action

The 'PLM Create References Action' allows users to configure a button that adds references to an object when used with the Multi-Reference Editor. This action can be configured to specify business rules for consistent creation, object nodes to restrict applicable references, and validation methods to ensure that the created reference is allowed.

Edit component



PLM Create References Action Properties

Component Description

This action can be added to a Node List and allows the user to create references with metadata attributes

Button Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReference:
Context Help	i18n.stibo.spireplm.webui.server.action.reference.CreateReference:
Custom Icon	... Reset
Dialog Header	i18n.stibo.spireplm.webui.server.action.reference.CreateReference:
Target Node Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReference:
Action Add Another Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReference:
Action Cancel Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReference:
Action Save Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReference:
*Root Node URLs	Add... Remove Up Down
*Get Reference Type - Business Function	...
*Copy From Template - Business Action	...
Validation - Business Function	... Clear
Copy From Template (Context) - Business Action	... Clear

Note: While most fields are optional, the 'Root Node URLs' field, the 'Get Reference Type - Business Function' field, and the 'Copy from Template - Business Action' field are required.

1. Fill out the necessary fields:

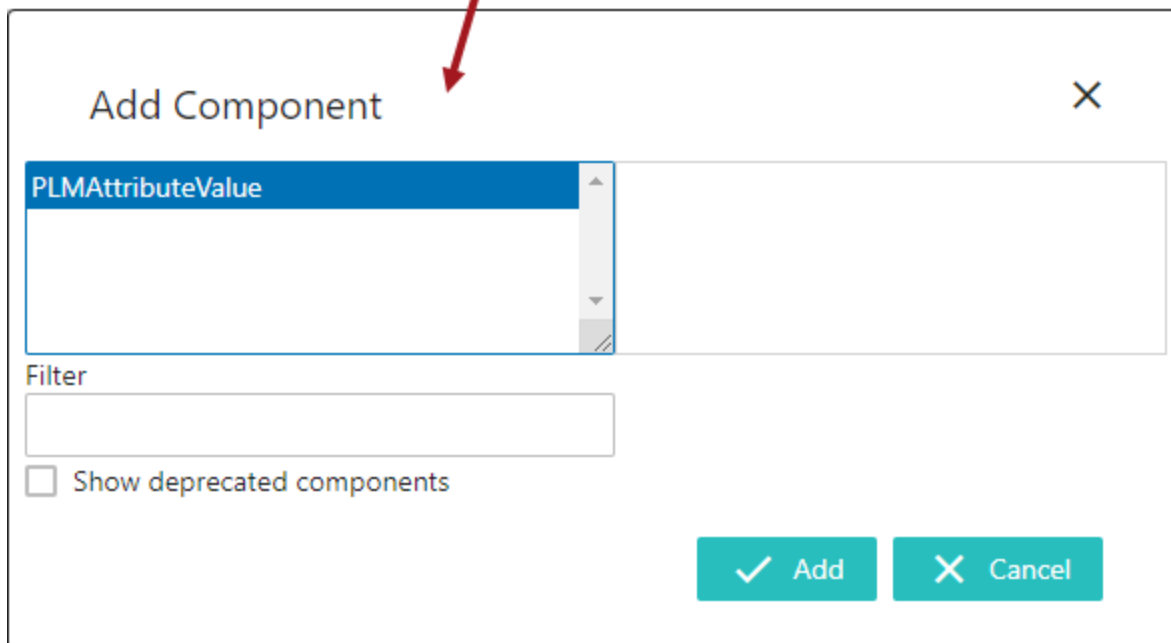
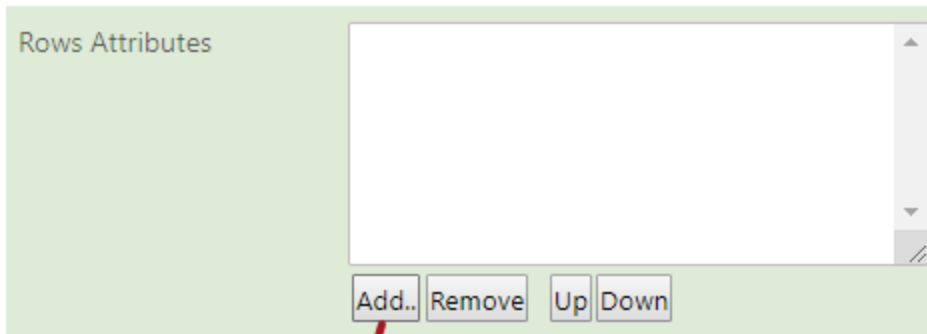
- **Button Label:** This field sets the displaying name for the action button. This field only works if the 'Include Label' parameter on the parent Node List is selected.
- **Context Help:** This field provides a description when the user hovers over the action button to assist users in the their tasks.
- **Custom Icon:** This allows users to select an icon to represent the action.
- **Dialog Header:** When the action is selected, this field will be populated as the title of the dialog that displays.
- **Target Node Label:** This field will is the one displayed as the title for the target node.
- **Action Add Another Label:** This field changes the label of the 'create another' button on the dialog that appears for this action. The default label is 'Add Another.'
- **Action Cancel Label:** This field changes the label of the cancel button on the dialog that appears for this action. The default label is 'Cancel.'
- **Action Save Label:** This field changes the label of the save button on the dialog that appears for this action. The default label is 'Save.'
- **Root Node URLs:** This field will establish from where the user may select the targets of the created references. This field is required.
- **Get Reference Type - Business Function:** This field sets which business function is used when getting the reference type ID. This is a required field.
- **Copy From Template - Business Action:** This field sets the business action that is used for copying the defaults to be displayed in the dialog box. This is a required field.
- **Validation - Business Function:** This field sets the business function that validates the data for the reference.
- **Copy From Template (Context) - Business Action:** This field sets the business action that is executed for copying default values from other contexts.

In this example, the action was titled as 'Add Parameter.' The completed configured Create Reference action looks like the following:

Button Label	Add Parameter							
Context Help	Add Parameter							
Custom Icon	... Reset							
Dialog Header	Add Parameter							
Target Node Label	Parameter							
Action Add Another Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReferencesA							
Action Cancel Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReferencesA							
Action Save Label	i18n.stibo.spireplm.webui.server.action.reference.CreateReferencesA							
*Root Node URLs	step://product?id=PLMParametersRoot Add... Remove Up Down PLMGetRequirementParameterReferenceType ...							
*Get Reference Type - Business Function	Caller parameters: Function input parameters: <table border="1"> <tr> <td>Target</td> <td>target</td> <td>▼</td> </tr> <tr> <td>Source</td> <td>source</td> <td>▼</td> </tr> </table>		Target	target	▼	Source	source	▼
Target	target	▼						
Source	source	▼						
*Copy From Template - Business Action	PLMFoodCopyRequirementParameterTemplate ... PLMFoodValidation ... Clear							
Validation - Business Function	Caller parameters: Function input parameters: <table border="1"> <tr> <td>Target</td> <td>target</td> <td>▼</td> </tr> <tr> <td>Source</td> <td>source</td> <td>▼</td> </tr> </table>		Target	target	▼	Source	source	▼
Target	target	▼						
Source	source	▼						
Copy From Template (Context) - Business Action	PLMFoodCopyRequirementParameterTemplate ... Clear							

2. Once created, under Child Components in the Rows Attributes field select 'Add.' The only option is 'PLMAttributeValue.'

Child Components



3. Select 'Add,' and the PLMAttributeValue Properties dialog will display.

Add component - configure required properties

Required properties (*) must be set before the component can be added to the configuration.

PLMAttributeValue Properties

*** Attribute**

Label

Mandatory ☐

Read Only ☐

▼ Conditions

Conditionally Driving Attribute

Conditionally Driving Values

Conditionally Mandatory ☐

- **Attribute:** Select the desired attribute type to be answered.
- **Label:** Sets the display name for the attribute.
- **Mandatory:** If selected, this make the field required to fill out before a user can submit their answers.
- **Read Only:** If selected, this means that the field cannot be edited.
- **Conditionally Driving Attribute:** If an attribute is selected for this field it will only display if the attribute that was selected for the Attribute field is answered in a certain way that is determined by the Conditional Attribute Value field.
- **Conditionally Driving Values:** The values of the Conditionally Driving Attribute that trigger the current attribute to be displayed.

- **Conditionally Mandatory:** If checked, if the attribute that was selected for the Attribute field is answered in a certain way that is determined by the Conditional Attribute Value field, then the attribute set for the Conditional Attribute field will display and must hold a value before the dialog can be submitted.

Once configured as needed, the button will be ready for adding new references. For more on how to use the PLM Create References Action, see the **Multi-Reference Editor Actions in Web UI** topic in this documentation.

For more information on how to use the Conditional settings, see the **Setting Conditional Settings in a MRE** topic in this documentation.

PLM Edit Reference Action

The 'PLM Edit Reference Action' allows users to configure metadata to append to the dialog menu, limiting what values on an object's references may be modified.

Properties

Configuration
Web UI style

PLM Food Supplier
Save
Close
New...
Delete
Rename
Save as...

PLM Edit Reference Action Properties
[go to parent](#)

Component Description
This action can be added to a Node List and allows the user to edit references with metadata attributes

Button Label
Answer Requirements

Context Help
i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi

Custom Icon
...
Reset

Dialog Header
Answer Requirement

Target Node Label
i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi

Action Save Label
i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi

Action Next Label
i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi

Action Previous Label
i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi

Action Cancel Label
i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi

Action Close Label
i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi

Help Text For Target Object
...
Clear

Show Help Texts For Attributes
☐

PLMFoodConditionalMandatoryRequirement
...
Clear

Conditional Rules - Business Function

Caller parameters:
Function input parameters:

Source
source

Target
target

PLMMandatoryResponse
...
Clear

Validation - Business Function

Caller parameters:
Function input parameters:

Source
source

Target
target

Child Components

Metadata Attributes

PLMAttributeValue (Requirement Description)
PLMAttributeValue (Meets Requirement?)
PLM Flex Value Attribute (Response Detail)
PLMAttributeValue (Help Text)

Add...
Remove
Up
Down

1. Fill out the necessary fields:

- **Button Label:** This field sets the displaying name of the action button. This field only works if the 'Include Label' parameter on the parent Node List is selected.
- **Context Help:** This field will provide a description when the user hovers over the action button to assist users in the their tasks.
- **Custom Icon:** This allows users to select an icon to represent the action.
- **Dialog Header:** When the action is selected, this field will be populated as the title of the dialog that displays.
- **Target Node Label:** This field is the one displayed as the title for the target node.
- **Action Save Label:** This field changes the label of the save button on the dialog that appears for this action. The default label is 'Save.'
- **Action Next Label:** This field changes the label of the next button on the dialog that appears for this action. The default label is 'Next.'
- **Action Previous Label:** This field changes the label of the previous button on the dialog that appears for this action. The default label is 'Previous.'
- **Action Cancel Label:** This field changes the label of the cancel button on the dialog that appears for this action. The default label is 'Cancel.'
- **Action Close Label:** This field is for the label on the close action button. It is shown when the last reference is edited.
- **Help Text for Target Object:** This field selects the attribute that holds the help text value of the target object.
- **Show Help Texts for Attributes:** If this option is selected, then the help text metadata of the selected attributes will be included.
- **Conditional Rules- Business Function:** The business function that provides the conditionally mandatory rules. If a business function is configured, any values configurations configured on any of the Child Components in the Metadata Attributes section are ignored, including any parameters set for mandatory or read only.
- **Validation - Business Function:** This field sets the business function that validates the data for the reference.
- **Metadata Attributes:** This child component allows for multiple metadata attributes to be added to the edit reference dialog.

2. Once created, under Child Components in the Metadata Attributes field, select 'Add.' The options to choose from are 'PLMAttributeValue' or 'PLM Flex Value Attribute.'

Add Component

PLM Flex Value Attribute

PLMAttributeValue

Filter

☐ Show deprecated components

Cancel

Add

- For the PLM Flex Value Attribute, the following fields can be configured:

Note: When adding a PLM Flex Value Attribute you are adding a defining attribute. A defining attribute is an LOV that contains a list of other attribute IDs that are used to control the format of data in a column in a Multi-Reference Editor. Users can choose one of the LOV values to define how the attribute's format should be controlled for the supplier' response.

Properties

Configuration Web UI style

PLM Food Supplier Save Close New... Delete Rename Save

PLM Flex Value Attribute Properties [go to parent](#)

Metadata Flex Value - Label

* Metadata Flex Value - Defining Attribute

▼ Conditions

Conditionally Driving Attribute ... Clear

Conditionally Driving Values

Add Remove Up Down

Conditionally Mandatory ☐

- **Metadata Flex Value - Label:** Sets the display name for the defining attribute.
- **Metadata Flex Value - Defining Attribute:** Select the desired defining attribute to be answered.
- **Conditionally Driving Attribute:** If an attribute is selected for this field it will only display if the attribute that was selected for the Attribute field is answered in a certain way determined by the Conditional Attribute Value field.
- **Conditionally Driving Values:** The values of the Conditionally Driving Attribute that trigger the current attribute to be displayed.
- **Conditionally Mandatory:** If checked, if the attribute that was selected for the Attribute field is answered in a certain way determined by the Conditional Attribute Value field, then the attribute set for the Conditional Attribute field will display and must hold a value before the dialog can be submitted.

For the PLM Flex Value Attribute, the following fields can be configured:

Add component - configure required properties

Required properties (*) must be set before the component can be added to the configuration.

PLMAttributeValue Properties

*** Attribute**

Label

Mandatory ☐

Read Only ☐

▼ Conditions

Conditionally Driving Attribute

Conditionally Driving Values

Conditionally Mandatory ☐

- **Attribute:** Select the desired attribute type to be answered.
- **Label:** Sets the display name for the attribute.
- **Mandatory:** If selected, this make the field required to fill out before a user can submit their answers.
- **Read Only:** If selected, this means that the field cannot be edited.
- **Conditionally Driving Attribute:** If an attribute is selected for this field it will only display if the attribute that was selected for the Attribute field is answered in a certain way which is determined by the Conditional Attribute Value field.
- **Conditionally Driving Values:** The values of the Conditionally Driving Attribute that trigger the current attribute to be displayed.

- **Conditionally Mandatory:** If checked, and the attribute that was selected for the Attribute field is answered in a certain way, which is determined by the Conditional Attribute Value field, then the attribute set for the Conditional Attribute field will display and must hold a value before the dialog can be submitted.

For more on how to use the PLM Edit References Action, see the **Multi-Reference Editor Actions in Web UI** topic in this documentation.

For more information on how to use the Conditional settings, see the **Setting Conditional Settings in an MRE** topic in this documentation.

For more on how to use the PLM Create References Action or PLM Edit Reference Action buttons, see the **Multi-Reference Editor Actions in Web UI** topic in this documentation.

Reference Metadata Flex Value Header Component

Prerequisites

All the steps provided in this topic assume the Web UI designer is in design mode and on a Multi Reference Editor screen prior to starting the configuration process. It is also assumed that all users (designers and end users) have the proper privilege to work with the Multi Reference Editor and its features. For more information about privileges and user setup, see the **Users and Groups** section and **Adding User Privileges for a Group** section of the **System Setup / Super User Guide** documentation.

Defining Attribute

A defining attribute is an LOV that contains a list of other attribute IDs that are used to control the format of data in a column in a Multi-Reference Editor. Users can choose one of the LOV values to define how the attribute's format should be controlled for the supplier' response.

The Reference Metadata Flex Value Header is a defining attribute to be added on a Multi Reference Editor as a header component.

Configuration

The Reference Metadata Flex Value Header is a component to be added on a Multi Reference Editor as a header component.

Add Component

Path Header

Reference Header

Reference Metadata Flex Value Header

Reference Metadata Value Header

Reference Suppression Header

Reference Type Header

Reference Visibility Header

Revision Header

Table header that shows flexible value of a metadata attribute on the reference or classification product link.

Filter

☐ Show deprecated components

✓ Add

✗ Cancel

Once added, the Reference Metadata Flex Value Header Properties dialog will display.

Add component - configure required properties

Required properties (*) must be set before the component can be added to the configuration.

Reference Metadata Flex Value Header Properties

Component Description Table header that shows flexible value of a metadata attribute on the reference or classification product link.

*Defining attribute

Dimensions

<Select an option>
Edit...

Help Text

Label

Mandatory

<Select a value>

Readonly

☐

▼ Advanced

Enable Locale Formatting

☐

Show Invalid Inherited Values

☐

Show LOV IDs

☐

No Wrap

☐

Context Help

Display Context Help

☒

✓ Add

✕ Cancel

While these fields are the same as other Header properties, the 'Defining attribute' field is a unique and required field. To understand how this attribute works, consider the following use case.

	Parameter	Parameter Description	Responses Required For	Supplier Response Detail
☐	Haemolytic Streptococci	Not detectable in 25g	Method Meets Requirement? Response Detail	August 8, 2020
☐	Mercury	Not detectable in 25g		0 µg
☐	Coliforme Germs	Not detectable in 25g	Meets Requirement?	0 µg

The defining attribute must be a LOV, and it is used to define what the 'Supplier Response Detail' attribute base type is for each parameter. For example, with the parameter 'heamolytic streptococci,' the 'Supplier Response Detail' attribute is defined as 'PLMDate' with a base type of 'ISO Date.' Any value entered for the 'Supplier Response Detail' are stored in the 'PLMDate' attribute. However, for the parameter 'Mercury,' the 'Supplier Response Detail' is defined as 'PLMNumberAndUnit,' which is a number with a unit. Values entered are stored in the 'PLMNumberAndUnit' attribute.

Setting Conditional Settings in a MRE

Product Developers can ensure the completeness of suppliers' responses to project requirements in a Multi-Reference Editor (MRE) based on conditions. Attributes (including flex attributes) can be configured to be conditional, conditional mandatory, or conditional with mandatory.

This can be configured in the Web UI designer on the PLM Create Reference Action or PLM Edit Reference Action on either the PLMAttributeValue or PLM Flex Value Attribute child components.

For example, the attribute, 'Additional Comments,' is set with a condition where it will only display if a specific value of 'No' is given for 'Meets Requirements.' This 'Additional Comments' field is not mandatory, and the response can still be saved if left blank.

The image shows two versions of the 'Answer Requirement' form (1 of 1) side-by-side. The left form shows the 'Meets Requirement?' dropdown menu open. A red arrow points from this dropdown to the 'No' option in the right form. Another red arrow points from the 'No' option down to the 'Additional Comments' text field, which is only visible in the right form. Both forms have the same other fields: 'Requirement' (Fair Trade Certified), 'Requirement Description' (Must be Fair Trade Certified. If Yes, please indicate expiration date), 'Response Detail' (12/07/2018), and 'Help Text' (The fair trade model requires rigorous protection of local food). The bottom buttons are 'Back', 'Save', 'Cancel', and 'Close'.

To configure the conditional setting based on the previous example, the PLMAttributeValue Properties screen in the designer would look like the following:

Properties (edited)

Configuration
Web UI style

PLM Food Supplier
Save
Close
New...
Delete
Rename
Save as...

PLMAttributeValue Properties
[go to parent](#)

* Attribute
PLMAdditionalSupplierComments

Label
Additional Comments

Mandatory
☐

Read Only
☐

Conditional

Conditional Attribute
PLMMeetsRequirement
Clear

Conditional Attribute Value

N

Add
Remove
Up
Down

Conditional Mandatory
☐

If the attribute is set as conditional mandatory, then when a supplier answers the 'Meets Requirements' attribute with 'No', the attribute field, 'Additional Comments' becomes mandatory, and must have a value to save the dialog.

Answer Requirement
1 of 1

Requirement
Fair Trade Certified

Requirement Description
Must be Fair Trade Certified. If Yes, please indicate expir

Meets Requirement?

Response Detail
12 / 07 / 2018

Additional Comments

Help Text
The fair trade model requires rigorous protection of local fa

Back Save Cancel Close

Answer Requirement
1 of 1

Requirement
Fair Trade Certified

Requirement Description
Must be Fair Trade Certified. If Yes, please indicate expir

Meets Requirement?
No

Response Detail
12 / 07 / 2018

* Additional Comments

This field is required.

Help Text
The fair trade model requires rigorous protection of local fa

Back Save Cancel Close

To configure the conditional mandatory setting, the designer would look similar to the following:

Properties (edited)

Configuration
Web UI style

PLM Food Supplier
Save
Close
New...
Delete
Rename
Save

PLMAttributeValue Properties
[go to parent](#)

* Attribute
PLMAdditionalSupplierCommen

Label
Additional Comments

Mandatory
☐

Read Only
☐

Conditional

Conditional Attribute
PLMMeetsRequirement
Clear

Conditional Attribute Value

N

Add
Remove
Up
Down

Conditional Mandatory
☒

If the attribute was set as conditional with mandatory, then when a supplier answers the 'Meets Requirements' attribute with 'No', the additional attribute field, 'Additional Comments,' displays, and it must have a value to save the dialog.

Answer Requirement1 of 1

Requirement

Fair Trade Certified

Requirement Description

Must be Fair Trade Certified. If Yes, please indicate expir

Meets Requirement?

Response Detail

12 / 07 / 2018

Help Text

The fair trade model requires rigorous protection of local fa

Back

Save

Cancel

Close

Answer Requirement1 of 1

Requirement

Fair Trade Certified

Requirement Description

Must be Fair Trade Certified. If Yes, please indicate expir

Meets Requirement?

No

Response Detail

12 / 07 / 2018

* Additional Comments

This field is required.

Help Text

The fair trade model requires rigorous protection of local fa

Back

Save

Cancel

Close

To configure the conditional with mandatory setting, the designer would look similar to the following:

Properties

Configuration
Web UI style

PLM Food Supplier
Save
Close
New...
Delete
Rename
Save

PLMAttributeValue Properties
[go to parent](#)

* Attribute
PLMAdditionalSupplierCommen

Label
Additional Comments

Mandatory
☒

Read Only
☐

Conditional

Conditional Attribute
PLMMeetsRequirement
Clear

Conditional Attribute Value

N

Add
Remove
Up
Down

Conditional Mandatory
☐

An additional option for configuring conditional or conditionally mandatory attributes or flex attributes is the use of the 'Conditional Rules- Business Function' field found on the PLM Edit Reference Action. If this is configured, it will override any child component configurations on the PLM Edit Reference Action, including anything configured on the PLMAttributeValue or the PLM Flex Value Attribute child components (mandatory, read only, and any conditional properties).

Properties

Configuration
Web UI style

PLM Food Supplier
Save
Close
New...
Delete
Rename
Save as...

PLM Edit Reference Action Properties [go to parent](#)

Component Description This action can be added to a Node List and allows the user to edit references with metadata attributes

Button Label	Answer Requirements	
Context Help	i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi	
Custom Icon		... Reset
Dialog Header	Answer Requirement	
Target Node Label	i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi	
Action Save Label	i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi	
Action Next Label	i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi	
Action Previous Label	i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi	
Action Cancel Label	i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi	
Action Close Label	i18n.stibo.spireplm.webui.server.action.reference.EditReferenceActi	
Help Text For Target Object		... Clear
Show Help Texts For Attributes	☐	
	PLMFoodConditionalMandatoryRequirement ... Clear	
Conditional Rules - Business Function	Caller parameters: Function input parameters: Source source Target target	
	PLMMandatoryResponse ... Clear	
Validation - Business Function	Caller parameters: Function input parameters: Source source Target target	

Child Components

Metadata Attributes	PLMAttributeValue (Requirement Description) PLMAttributeValue (Meets Requirement?) PLM Flex Value Attribute (Response Detail) PLMAttributeValue (Help Text)
	Add... Remove Up Down

See the **Business Rules** documentation for more on how to create business rules.