TEAMCENTER

Migration of Classic Variants to Product Configurator

Teamcenter 2412

SIEMENS

About Siemens Digital Industries Software

Siemens Digital Industries Software is a global leader in the growing field of product lifecycle management (PLM), manufacturing operations management (MOM), and electronic design automation (EDA) software, hardware, and services. Siemens works with more than 100,000 customers, leading the digitalization of their planning and manufacturing processes. At Siemens Digital Industries Software, we blur the boundaries between industry domains by integrating the virtual and physical, hardware and software, design and manufacturing worlds. With the rapid pace of innovation, digitalization is no longer tomorrow’s idea. We take what the future promises tomorrow and make it real for our customers today. Where today meets tomorrow. Our culture encourages creativity, welcomes fresh thinking and focuses on growth, so our people, our business, and our customers can achieve their full potential.

Support Center: **support.sw.siemens.com**

Send Feedback on Documentation: **support.sw.siemens.com/doc_feedback_form**

Contents

1. Getting ready to adopt Product Configurator for existing classic variant data

If you already use classic variants, you can start planning your existing variability data to prepare for using Product Configurator.

> Note:
>
> To support migration, you can add variability using a Teamcenter object of the **Item** type in Product Configurator. The **Item** type object acts as a configurator context. You can associate families, features, and author and validate configurator rules in the context of this **Item**. However, you cannot create models in the **Item** context. For all new configurator data, it is recommended to create and manage variability using configurator objects, such as configurator dictionary and configurator context.

You can configure product structures by Product Configurator variants. You can use the Product Configurator variants when your administrator installs the solution template to support Product Configurator for Structure Manager and sets the preference to enable the Product Configurator variant mode.

Review the following list to ascertain if you can benefit from any of the listed Product Configurator functionality.

- **Saved variant rules (SVR)**

  An equivalent SVR is provided with the new Product Configurator to store variant configuration criteria and optional validation records. They are attached to the configurator context item or the application model (product design) by GRM relationships.

- **Multiselect families**

  This is new functionality for existing classic variant data users. They can now combine the families that were explicitly created to allow you to create conjunction of features. For example, you can create a multiselect family for accessories and keep all these features in one family and still configure structure with multiple features from the same accessory family.

- **Default rule**

  Existing feature defaults (with or without preconditions) can be represented using this rule type.

- **Inclusion rule**

This is new functionality for existing classic variant data users. An inclusion rule defines a condition for which a specified selection of one or more features implies the setting of one or more other features.

Inclusion rules express *Do* or *Must Do* conditions. They contrast with exclusion rules that express *Don't* conditions. If an inclusion rule says "If Exterior=Black → Interior=Anthracite", the system forces anthracite interiors for black exteriors. Conversely, if an exclusion rule says "If Exterior=Black → Interior=Anthracite", the system disallows anthracite interiors for black exteriors.

Existing feature defaults (with or without preconditions) can be mapped to the inclusion rules with messages and severity.

The evaluation of an inclusion rule changes the product configuration or it fails, while the evaluation of an exclusion rule leaves the product configuration unchanged or it fails. That is, exclusion rules only validate conditions, while inclusion rules validate *and* modify conditions.

- **Exclusion rule**

  Existing rule checks can be represented using exclusion rules. An exclusion rule is a convenient way to represent exclusions. If an exclusion rule says "If Exterior=Black → Interior=Anthracite", the system disallows anthracite interiors for black exteriors.

- **Availability rule**

  Existing rule checks can also be represented using availability rules. An availability rule is a convenient way to represent *May Do* conditions. If an availability rule says "Interior=Anthracite is available ? If Exterior=Black", the system requires anthracite interiors for black exteriors.

- **Free-form family**

  This is new functionality for classic variant users. Create free-form variant families and assign values during authoring or configuration. You can define optional boundaries, rather than just lists of predefined values.

- **Nondiscretionary family**

  This is new functionality for existing classic variant data users. You can classify a family to be discretionary or nondiscretionary (required) variant families. System will enforce features from nondiscretionary family during configuration.

- **Selective rule type evaluation**

  This is new functionality for existing classic variant data users. You can turn off certain rules type during configuration.

Following are the best practices to adopt Product Configurator:

- Make all options global.

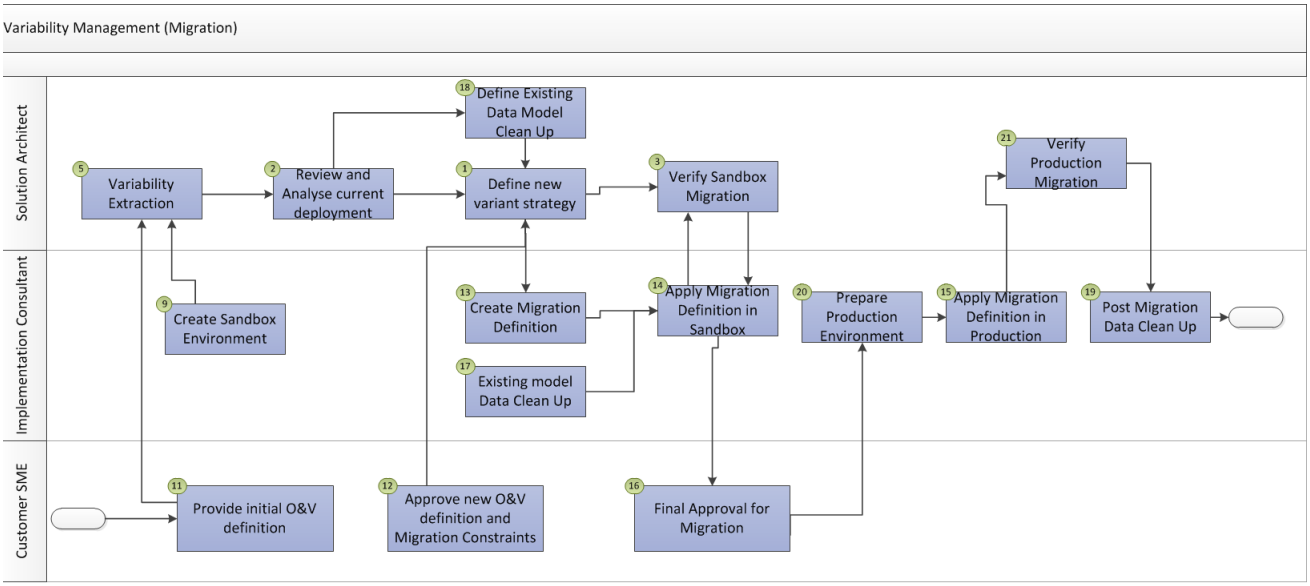
- Do not define any variability within the product structure (in general).
- Attach all options to one or more global option items. Create global option items to correspond to your projected future configurator context needs.
  - For a given product, collect all variability in a single Global Option Item (GOI).
- Conflicting scenarios may occur due to rules or constraints when variability for all products is added to single GOI.
  - In such scenario, the variability should be defined within the product structure.
  - The top item could then be a future configurator context.

  For all other scenarios, the variability could be defined in a single GOI for all products that belong to same product line.
- Position options for new Product Configurator from the start. Think about:
  - Numeric features you are out of necessity storing as strings.
  - Any other features you are storing as strings but are really intended to be dates, Booleans, etc.
  - Be aware of any options you are creating that represent product models.
- Do not revise Global Option Items.

2. Process to migrate classic variant to Product Configurator

To migrate classic variant data to Product Configurator, you can adopt the following migration process.

For information on how to utilize this process at your site, contact your Siemens Digital Industries Software representative.



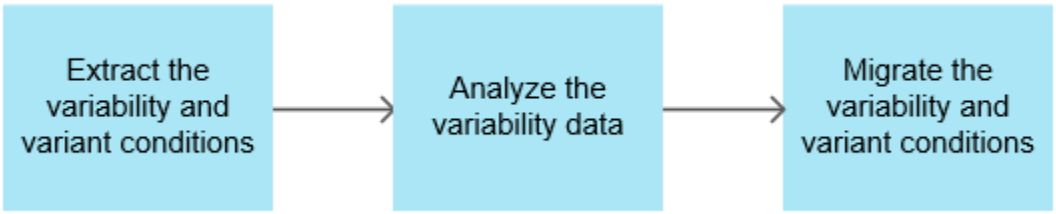
Business process step	Description	Roles
1. Define new variant strategy	Define custom code strategy Decide on a BVR, 4GD, or mixed mode approach Define the variant item strategy Define new feature types Define new properties Define new logical variant model mapping Present the new variant model for approved Define sample configurations Define positive or negative biased model Define migration test cases Create new description configuration document	Solution architect

Business process step	Description	Roles
	Check the current variant model complies completely with the new variant model Note: At the same time as you define your new variant strategy, you should define any steps necessary to clean up your data model. This activity varies between customer sites and is out of the scope of this documentation.	
2. Review and analyze current deployment	Review customer variability document Review extracted features and rules Review extracted SVRs Review referenced features Review extracted variant expressions Review Multi-Site requirements Review applications in the current deployment that use variants	Solution architect
3. Verify migration	Compare new model configuration with Excel export Refine variant model Record sample configurations against test plan Extrapolate sandbox times to production Produce sandbox migration report	Solution architect
5. Variability extraction	Extract features and rules Extract SVRs Extract variant expressions Create exported data reports	Solution architect
9. Create sandbox environment	Install Teamcenter in sandbox with customer configuration Import customer data and configuration Export sample configurations to Excel	Implementation consultant

Business process step	Description	Roles
	Convert variant data to new variant model if required	
11. Provide initial features and variants (O&V) definition	Analyze initial O&V definition Produce customer variability document	Customer SME
12. Approve new O&V definition and migration constraints	Approve new O&V definition Agree migration constraints	Customer SME
13. Create migration definition	Create variant condition mapping file	Implementation consultant
14. Apply migration definition in sandbox	Record migration timings	Implementation consultant
15. Apply migration definition in production	Record migration timings	Implementation consultant
16. Final approval for migration	Review sandbox migration report	Customer SME
17 and 18. Define and execute clean up of existing data model	Clean up conflicting data	Solution architect and implementation consultant
19. Clean up data after migration	Delete unreferenced migrated data	Implementation consultant
20. Prepare production environment	Perform new variant model conversion if required	Implementation consultant
21. Verify production migration	Compare new model configuration using Microsoft Excel export Record sample configurations against test plan	Solution architect

3. About migrating variability

Migrating variability involves migrating variant options, defaults, derived defaults, rule checks, and saved variant rules from the classic format to Product Configurator format. This is a three-step process as follows:



1. Extract the variability first and then the variant conditions.

   Variability includes options and values; fixed defaults and derived defaults; external options and global options; and saved variant rules (SVRs).

   Generate a report of the existing classic variability by using the **variant_data_analysis** utility.

2. Analyze the variability data.

   Analyze the variability of the classic variant data. This is a manual process and may have to be repeated iteratively.

3. Migrate the variability first and then the variant conditions.

   Migrate the classic variant data for a **single site** or **multisites** using the **variant_migration** utility.

> Note:
>
> If you have created variability expressions using the **Variability** tab in Platform Designer, such data is not migrated. Additionally, this data is not visible after migration from classic variants to Product Configurator.

4. Extracting classic variant data

Use the **variant_data_analysis** utility to extract variability or variant conditions, or both. By default, it is variability.

1. From the Teamcenter command prompt, switch to the directory where you want to extract the classic variant data.

2. Run the **variant_data_analysis** utility to extract the variant data.

Argument	Description
-mode=	Extracts variability or variant conditions, or both. By default, it is variability. Valid values are **Variability**, **VariantCondition**, **Variability_VariantCondition** and **Split**. Split is used to split the variant condition XML file based on split criteria if the file has numerous expressions to be migrated. The number of split files may be decided by the number of processes that you can run in parallel. To extract only variability, specify **-mode=Variability**. To extract both conditions, specify **-mode=Variability_VariantCondition**.
-item=	Specifies the product Item ID for which you want to extract classic variant data.
-revRule=	Specifies the revision rule for extracting option defaults and rule checks for the input item. The latest revision of the input item is used if this parameter is not provided. This parameter is applicable only to **-mode=Variability**.
-uidFile=	Specifies the full file path and the text file name in which the owning item UIDs are generated. This is applicable for only **Variability** and **Variability_VariantCondition** modes. If the **-xmlFilePath=** argument is not provided, the **-uidFile=** is the output. The owning item UIDs for the entire database are generated in the UID file. This file is used as an input for generating the analysis XML file.
-xmlFilePath=	Specifies the folder path where the output XML file is generated. Along with this argument, you must provide the **-uidFile** argument if you are using the **Variability** or **Variability_VariantCondition** mode.

Argument	Description
-xml_file=	Specifies the name of the variant condition XML file that must be split. This input is valid only for **-mode=Split**.
-fileCount=	Specifies the criteria for splitting the files, that is, the number of split files. You must provide either this option or **-exprsCount=**. This input is valid only for **-mode=Split**.
-exprsCount=	Specifies the number of variant conditions to be included in each split file. You must provide either this option or **-fileCount**. This input is valid only for **-mode=Split**.

3. Split the files.

   Split is used to split the variant condition XML file based on split criteria if the file has numerous expressions to be migrated. The number of split files may be decided by the number of processes that you can run in parallel.

   You can use one of the following arguments:

   - Split files: **variant_data_analysis -xml_file=***Input Variant Condition XML file* **-mode=***Split* **-fileCount=***Number of split files*

     You can use the **variant_data_analysis** utility to generate the UIDs of the owning items in a text file. Owning items are items that might have options and allowed values, option defaults, rule checks, derived defaults, and saved variant rules (SVRs). If the UID file has a large number of owning items, for example, 100,000 owning items, you can use the *Split* argument to split the files.

   - Specify the number of variant conditions in each split file: **variant_data_analysis -xml_file=***Input Variant Condition XML file* **-mode=***Split -exprsCount=Specify the number of variant conditions in each split file*

4. **Analyzing the variability data**.

> Caution:
>
> The variability and variant condition are extracted as XML files. Do not modify these files.

Examples for extracting classic variant data

Task	Arguments
Generate the owning items by scanning the entire database	**variant_data_analysis -uidFile=***Directory_Path\fulldb.txt*
Generate the owning items and the XML file by scanning the entire database	**variant_data_analysis -uidFile=***Directory_Path\fulldb.txt* **-xmlFilePath=***Specify_Folder_Path_where_XML_File_is_Generated*
Generate the owning items in the structure with the specified item ID	**variant_data_analysis -item=***Specify_Product_Item_ID* **-uidFile=***itemid.txt*
Generate the XML file for the product owning items	**variant_data_analysis -uidFile=***itemid.txt* **-xmlFilePath=***Specify_Folder_Path_where_XML_File_is_Generated*
Generate owning item UIDs and extract both the variants and variant conditions by scanning the entire database	**variant_data_analysis -uidfile=***fulldb.txt* **-mode=***Variability_VariantCondition* **-xmlFilePath=***Specify_Folder_Path_where_XML_File_is_Generated*
Extract only variant conditions by scanning the entire database	**variant_data_analysis -mode=***VariantCondition* **-xmlFilePath=***Specify_Folder_Path_where_XML_File_is_Generated*
Generate the variability and variant condition XML files for a specified item ID	**variant_data_analysis -item=***itemID* **-uidfile=***itemID.txt* **-mode=***Variability_VariantCondition* **-xmlFilePath=***Specify_Folder_Path_where_XML_File_is_Generated* **-uidfile=** and **-xmlFilePath=** are mandatory inputs.
Generate the variant condition XML files for the specified item ID	**variant_data_analysis -item=***itemID* **-mode=***VariantCondition* **-xmlFilePath=***Specify_Folder_Path_where_XML_File_is_Generated* The **-xmlFilePath=** is a mandatory input.

5. Analyzing the variability data

You can use the **variant_data_analysis** utility to analyze the variability data. It helps you understand how the classic data is going to be mapped to Product Configurator.

You can use this utility to generate an analysis report from the data you want to migrate. This report can be generated for the complete database or top-level programs, that is, specific product revisions that you want to migrate. The report can also be used to gauge the amount of data that is to be migrated. This helps to plan the migration activity effectively.

If there are any owning items that are not used anywhere, remove them from the UID file and generate the XML file again.

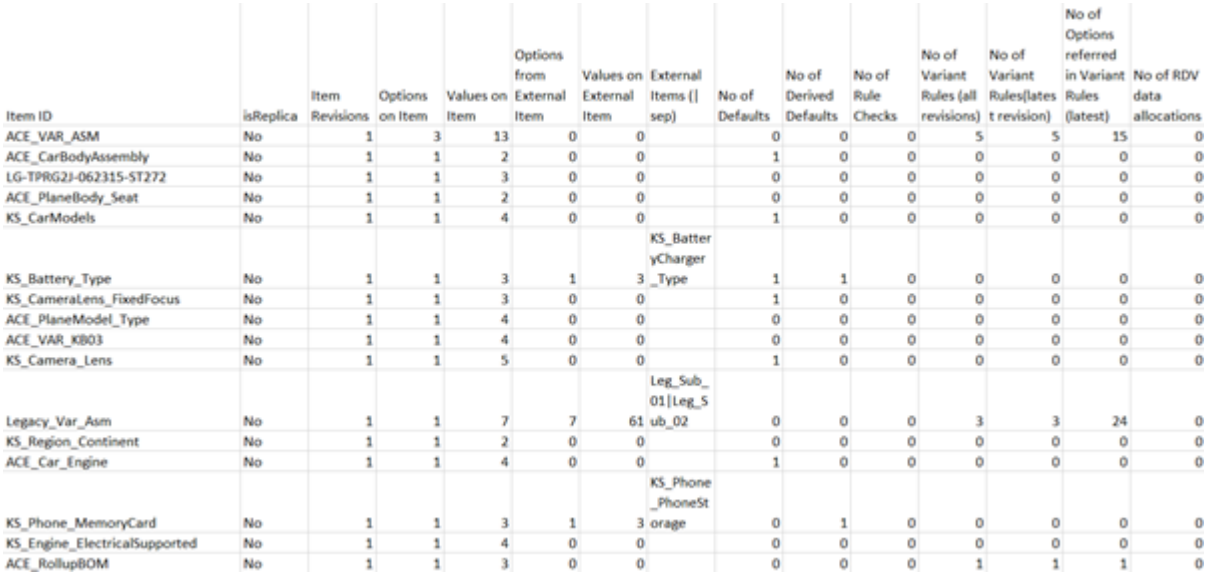
After running this utility, you can consolidate variant data from multiple sites and clean up the data. This is a manual process. For example, an option such as color can be used with certain values in one part of the structure and used with different values in another part of the structure. After migration, the multiple color options are retained, and the migrated structure might have multiple families with the same name as color and features.

Classic variant data	Examples	Migrated?
Variant conditions	This is a condition that an engineering user sets on an occurrence to specify the option values that configure the occurrence (for example, **Load IF engine = 1200**). More complex condition statements may also be defined.	Yes
Variant rules	These are rules that are used to configure a particular variant of a structure. The variant rule is a group of options and values such as **color = red, material = cotton**.	Yes
Options and allowed values	These are options that are used to configure structures at the lower level. For example, the **color** option allows values such as **red**, **blue**, or **green**.	Yes
Fixed defaults	This is a value that you specify as a default. For example, the option **engine** could have a default value set to **2.0 L**.	Yes
Derived defaults	This is a default value that depends on a certain condition (for example, **radio = stereo IF car type = GLX**). A derived default is attached to an item revision but applies globally to a loaded product structure.	Yes
External options	This is an option that is typically placed at the top-level module and sets the value of options that reference it in lower level modules.	Yes

Classic variant data	Examples	Migrated?
Global options	This is an option that defines a set of standard allowed values in a central module. These values can be used in other modules.	Yes
Stored option sets	This is a logical grouping of a set of options.	No

1. Run the **variant_data_analysis utility** to extract variant data.

2. Open the CSV file from the folder you specified using the **-xmlFilePath=** argument.



Item ID	isReplica	Item Revisions	Options on Item	Values on Item	Options from External Item	Values on External Item	External Items (	sep)	No of Defaults	No of Derived Defaults	No of Rule Checks	No of Variant Rules (all revisions)	No of Variant Rules(latest revision)	No of Options referred in Variant Rules (latest)	No of RDV data allocations
ACE_VAR_ASM	No	1	3	13	0	0		0	0	0	5	5	15	0
ACE_CarBodyAssembly	No	1	1	2	0	0		1	0	0	0	0	0	0
LG-TPRG2J-062315-ST272	No	1	1	3	0	0		0	0	0	0	0	0	0
ACE_PlaneBody_Seat	No	1	1	2	0	0		0	0	0	0	0	0	0
KS_CarModels	No	1	1	4	0	0		1	0	0	0	0	0	0
KS_Battery_Type	No	1	1	3	1	3	KS_BatteryCharger_Type	1	1	0	0	0	0	0
KS_CameraLens_FixedFocus	No	1	1	3	0	0		1	0	0	0	0	0	0
ACE_PlaneModel_Type	No	1	1	4	0	0		0	0	0	0	0	0	0
ACE_VAR_KB03	No	1	1	4	0	0		0	0	0	0	0	0	0
KS_Camera_Lens	No	1	1	5	0	0		1	0	0	0	0	0	0
Legacy_Var_Asm	No	1	1	7	7	61	Leg_Sub_01|Leg_Sub_02	0	0	0	3	3	24	0
KS_Region_Continent	No	1	1	2	0	0		0	0	0	0	0	0	0
ACE_Car_Engine	No	1	1	4	0	0		1	0	0	0	0	0	0
KS_Phone_MemoryCard	No	1	1	3	1	3	KS_Phone_PhoneStorage	0	1	0	0	0	0	0
KS_Engine_ElectricalSupported	No	1	1	4	0	0		0	0	0	0	0	0	0
ACE_RollupBOM	No	1	1	3	0	0		0	0	0	1	1	1	0

The CSV files displays the following for each owning item in the generated UID text file.

- Options from external items, values on external items, and the external items (item IDs)
- The number of defaults, derived defaults, and rule checks.
- The number of variant rules from all revisions and the latest revisions, and the number of options referred in the latest variant rule.
- The number of allocations from repeatable digital validation (RDV).

6. Mapping classic variant data to Product Configurator data

In Product Configurator, you can create variant data that is independent of any application domain. This variant data, including features, families, and rules, must be available before you can map classic variant data to Product Configurator data.

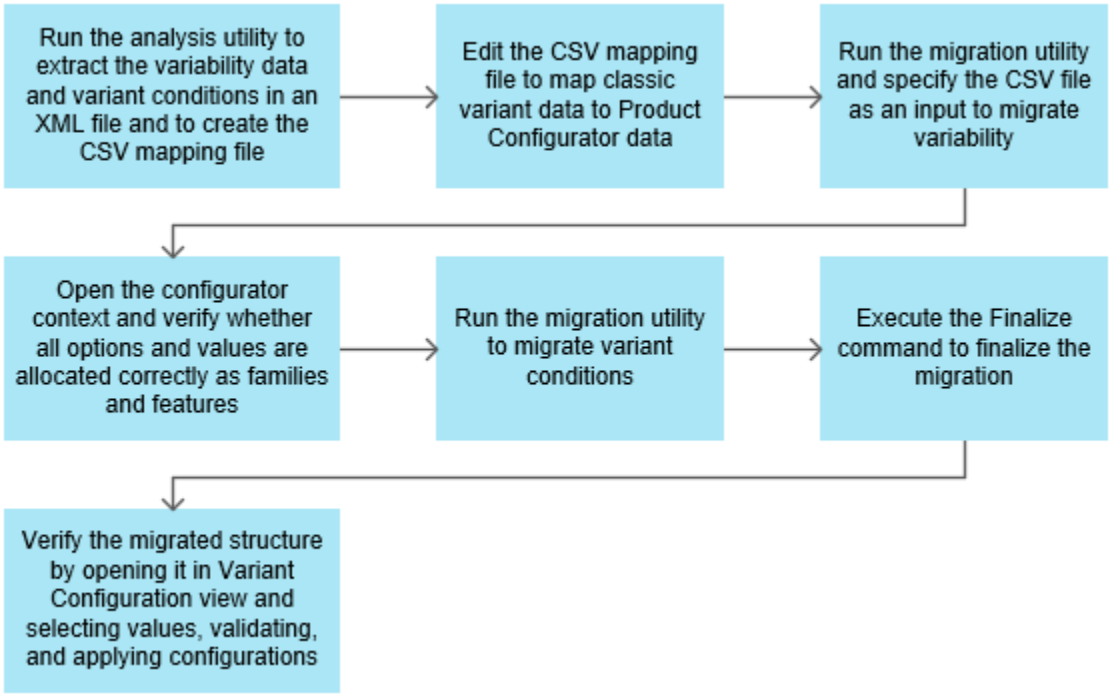
Options and values in classic variant data are used to configure structures at the lower level. For example, the **color** option allows values such as **Red**, **Blue**, or **Green**. They must be mapped to the corresponding families and features in Product Configurator during the migration process. Usually, classic variant data contains a lot of redundant or duplicate options and values. You can use the CSV mapping file to consolidate such data during the migration. This file is generated for each product or master structure when you run the analysis utility. It is available in the same directory where the **Variant Analysis XML** file is generated.

The CSV mapping file has six columns. The first three columns are *prepopulated* with classic variant option namespace, classic option name, and classic value name. The other three columns are for Product Configurator variability and they are *empty*. You must edit this file to map classic variant data to Product Configurator data. You provide the mapping by specifying the values from the corresponding namespace, families, and features in Product Configurator.

When you specify the CSV mapping file with the correct mapping values as an input while running the migration tool, family and features are retrieved from the global dictionary in Product Configurator and allocated to the configurator context. If you do not specify the CSV mapping file, the system creates families and features with the same name as the classic variant option names and values.

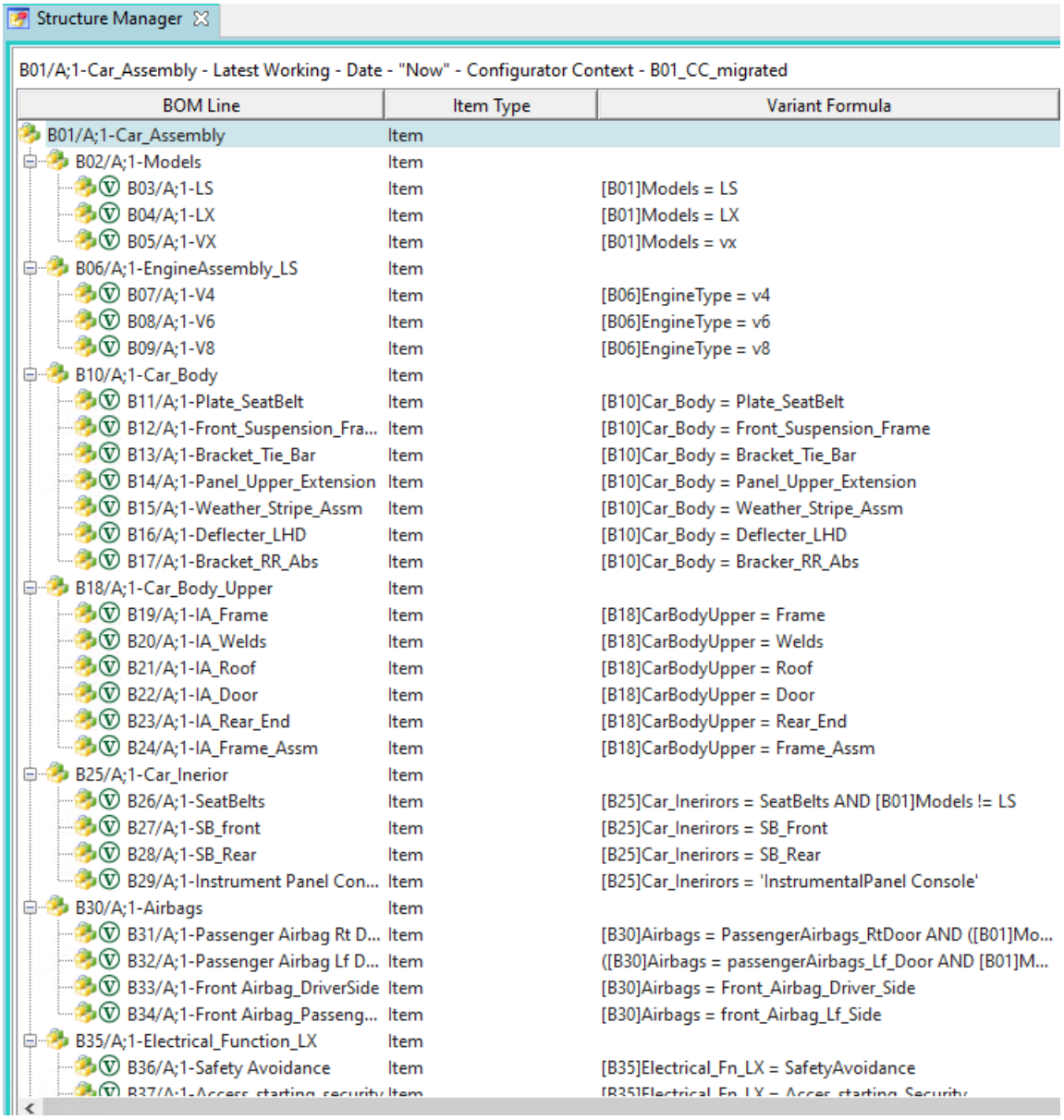
You must specify this edited CSV mapping file as an input while running the migration tool to migrate variability. Subsequently, all the fixed defaults, derived defaults, and the rule checks also inherit the prefix specified for features and families in the CSV mapping file (see **PCA_** prefix examples that follow).

The process for mapping classic options and values during migration is as follows:

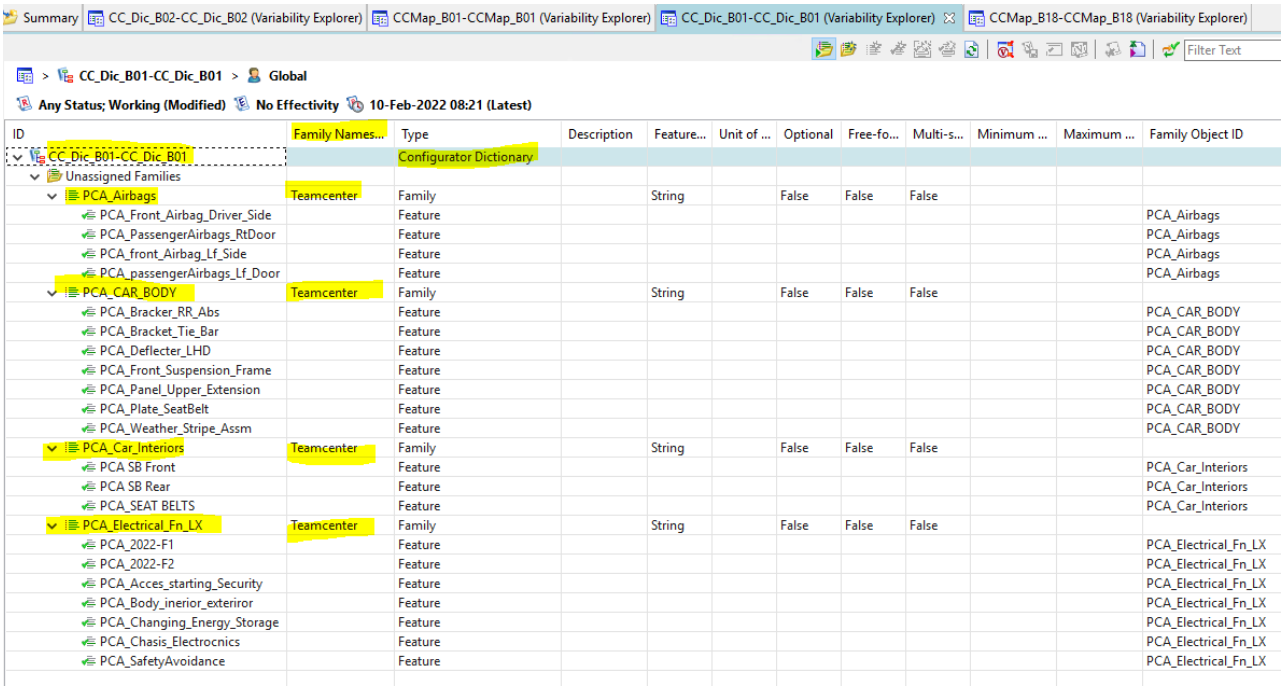


Migration example using the CSV mapping file

1. Open or create a structure as follows. This is the structure for which you want to map classic options and values.



2. Open or create variability in Product Configurator as follows.



ID	Family Names...	Type	Description	Feature...	Unit of ...	Optional	Free-fo...	Multi-s...	Minimum ...	Maximum ...	Family Object ID
CC_Dic_B01-CC_Dic_B01		Configurator Dictionary									
Unassigned Families											
PCA_Airbags	Teamcenter	Family		String		False	False	False			
PCA_Front_Airbag_Driver_Side		Feature									PCA_Airbags
PCA_PassengerAirbags_RtDoor		Feature									PCA_Airbags
PCA_front_Airbag_Lf_Side		Feature									PCA_Airbags
PCA_passengerAirbags_Lf_Door		Feature									PCA_Airbags
PCA_CAR_BODY	Teamcenter	Family		String		False	False	False			
PCA_Bracker_RR_Abs		Feature									PCA_CAR_BODY
PCA_Bracket_Tie_Bar		Feature									PCA_CAR_BODY
PCA_Deflecter_LHD		Feature									PCA_CAR_BODY
PCA_Front_Suspension_Frame		Feature									PCA_CAR_BODY
PCA_Panel_Upper_Extension		Feature									PCA_CAR_BODY
PCA_Plate_SeatBelt		Feature									PCA_CAR_BODY
PCA_Weather_Stripe_Assm		Feature									PCA_CAR_BODY
PCA_Car_Interiors	Teamcenter	Family		String		False	False	False			
PCA SB Front		Feature									PCA_Car_Interiors
PCA SB Rear		Feature									PCA_Car_Interiors
PCA_SEAT BELTS		Feature									PCA_Car_Interiors
PCA_Electrical_Fn_LX	Teamcenter	Family		String		False	False	False			
PCA_2022-F1		Feature									PCA_Electrical_Fn_LX
PCA_2022-F2		Feature									PCA_Electrical_Fn_LX
PCA_Acces_starting_Security		Feature									PCA_Electrical_Fn_LX
PCA_Body_inerior_exteriror		Feature									PCA_Electrical_Fn_LX
PCA_Changing_Energy_Storage		Feature									PCA_Electrical_Fn_LX
PCA_Chasis_Electrocnics		Feature									PCA_Electrical_Fn_LX
PCA_SafetyAvoidance		Feature									PCA_Electrical_Fn_LX

3. Execute the analysis utility in the default mode from the Teamcenter command prompt.

   variant_data_analysis -item=*B01* **-uidFile=***B01_Variability.txt*

   Where *B01* is the top item ID of the structure to be migrated.

4. Execute the analysis utility to generate the XML file and the CSV mapping file from the Teamcenter command prompt.

   variant_data_analysis -uidFile=*C:\Migration\B01_Variability.txt* **-xmlFilePath=**C:\Migration

```
** Analyzing Classic Variant Data **
** # of Valid Owning Items =  : 9 **
** Classic Variant Data Analysis Completed **
** Owning Item Stats file output created at
c:\Migration\B01_OwningItemStatsFile_44e8a83d.csv **
** Mapping file output created at
c:\Migration\B01_MappingFile_44e8a83d.csv **
** Creating XML file :
c:\Migration\B01_VariantAnalysisReport_44e8a83d.xml **
** XML successfully Created **
** Exiting successfully **
```

The **variant_data_analysis utility** generates the variability by default in an XML file. It helps you **understand how the classic data is going to be mapped** to Product Configurator. It also displays the number of owning items. Owning items are items that contain variability data. In addition, it creates a mapping file in CSV format for each product or master structure.

5. Extract variant conditions.

   Run the analysis utility to generate the XML file for variant conditions.

   variant_data_analysis -item=*B01* **-mode=***VariantCondition* **-xmlFilePath=***C:\Migration*

6. Edit the CSV mapping file to map classic options and values.

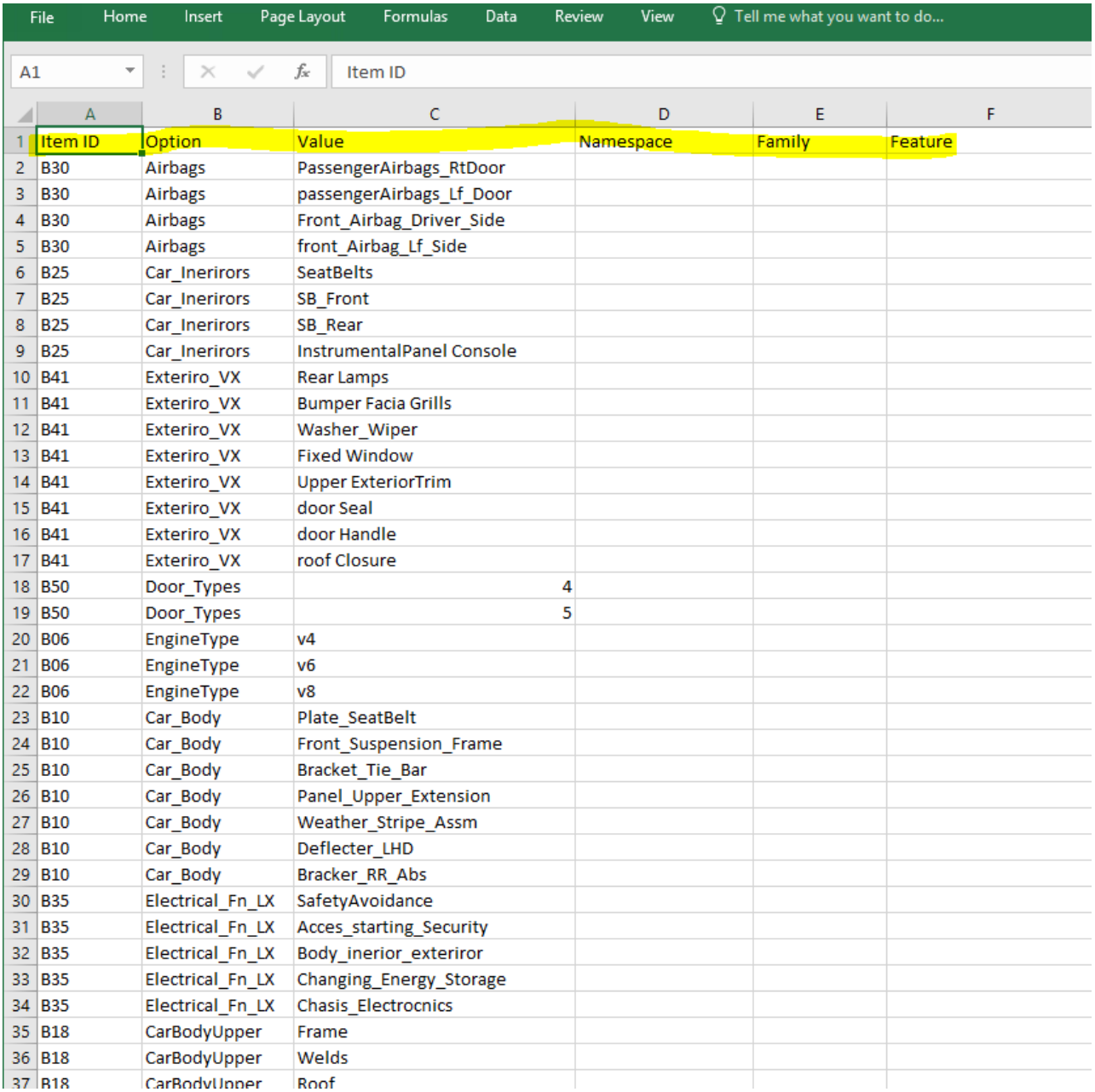
   The CSV mapping file has six columns. The first three columns are for classic variants. They are **Item ID**, **Option**, and **Value**, and these columns are prepopulated with classic namespace, option name, and value name.

   The CSV mapping file has entries for each item, that is, option and value combination (see example).

   As an example, assume that you have a structure with the top item ID as **B01**, an option such as **Color**, and values such as **Red**, **Blue**, and **Green** as part of the classic variant data. In such a case, the mapping CSV file has an entry for each of these three color values against the color option.

Item ID	Option	Value	Namespace	Family	Feature
B01	Color	Red			
B01	Color	Blue			
B01	Color	Green			

   The other three columns are for Product Configurator variability and they are empty. You must provide mapping by specifying the values from the existing namespace, families, and features in Product Configurator.



File Home Insert Page Layout Formulas Data Review View Tell me what you want to do...

A1 | Item ID

	A	B	C	D	E	F
1	Item ID	Option	Value	Namespace	Family	Feature
2	B30	Airbags	PassengerAirbags_RtDoor			
3	B30	Airbags	passengerAirbags_Lf_Door			
4	B30	Airbags	Front_Airbag_Driver_Side			
5	B30	Airbags	front_Airbag_Lf_Side			
6	B25	Car_Inerirors	SeatBelts			
7	B25	Car_Inerirors	SB_Front			
8	B25	Car_Inerirors	SB_Rear			
9	B25	Car_Inerirors	InstrumentalPanel Console			
10	B41	Exteriro_VX	Rear Lamps			
11	B41	Exteriro_VX	Bumper Facia Grills			
12	B41	Exteriro_VX	Washer_Wiper			
13	B41	Exteriro_VX	Fixed Window			
14	B41	Exteriro_VX	Upper ExteriorTrim			
15	B41	Exteriro_VX	door Seal			
16	B41	Exteriro_VX	door Handle			
17	B41	Exteriro_VX	roof Closure			
18	B50	Door_Types	4			
19	B50	Door_Types	5			
20	B06	EngineType	v4			
21	B06	EngineType	v6			
22	B06	EngineType	v8			
23	B10	Car_Body	Plate_SeatBelt			
24	B10	Car_Body	Front_Suspension_Frame			
25	B10	Car_Body	Bracket_Tie_Bar			
26	B10	Car_Body	Panel_Upper_Extension			
27	B10	Car_Body	Weather_Stripe_Assm			
28	B10	Car_Body	Deflecter_LHD			
29	B10	Car_Body	Bracker_RR_Abs			
30	B35	Electrical_Fn_LX	SafetyAvoidance			
31	B35	Electrical_Fn_LX	Acces_starting_Security			
32	B35	Electrical_Fn_LX	Body_inerior_exteriror			
33	B35	Electrical_Fn_LX	Changing_Energy_Storage			
34	B35	Electrical_Fn_LX	Chasis_Electrocnics			
35	B18	CarBodyUpper	Frame			
36	B18	CarBodyUpper	Welds			
37	B18	CarBodyUpper	Roof			

In Product Configurator, each family has a namespace, and each feature has a family object ID associated with it. As per the **variability example in Product Configurator in step 2**, for row one in the CSV mapping file, specify values as follows:

Example before editing the file:

Item ID	Option	Value	Namespace	Family	Feature
B30	Airbags	PassengerAirbags_Rt_Door			
B30	Airbags	PassengerAirbags_Lf_Door			
B30	Airbags	Front_Airbag_Driver_Side			

Example after editing the file:

Item ID	Option	Value	Namespace	Family	Feature
B30	Airbags	PassengerAirbags_Rt_Door	Teamcenter	PCA_Airbags	PCA_PassengerAirbags_Rt_Door
B30	Airbags	PassengerAirbags_Lf_Door	Teamcenter	PCA_Airbags	PCA_PassengerAirbags_Lf_Door
B30	Airbags	Front_Airbag_Driver_Side	Teamcenter	PCA_Airbags	PCA_Front_Airbag_Driver_Side

Continue specifying values for other rows as appropriate in the CSV mapping file.

When you specify the CSV mapping file with the correct mapping values as an input while running the migration tool, family and features are retrieved from the global dictionary in Product Configurator and allocated to the configurator context. If you do not specify the CSV mapping file, options and values are created as new values.

> Note:
>
> If the item ID of the structure has a value such as **000022**, the CSV mapping file displays this as **22** by default. Be sure to open the mapping CSV file in a text editor and verify that the value is correct before specifying the file as an input while running the migration tool.

7. To retain classic variant data or options and values as is, do not specify any values in the **Namespace**, **Family**, or **Feature** columns of the mapping CSV file. In such a case, these are created as new values in Product Configurator.

8. To consolidate all options with multiple values in classic variant data, specify the appropriate namespace, common family name, and feature names as follows.

   Example for consolidating all options with multiple values:

Item ID	Option	Value	Namespace	Family	Feature
B55	Color	Red	Teamcenter	PCA_Color	PCA_Red
B55	Color	Blue	Teamcenter	PCA_Color	PCA_Blue
B56	Color	Red	Teamcenter	PCA_Color	PCA_Red
B56	Color	Blue	Teamcenter	PCA_Color	PCA_Blue

9. To map classic variant data for options that have string values such as **yes** and **no**, specify the appropriate namespace, family name, and feature name from Product Configurator as follows:

   Example for string values:

Item ID	Option	Value	Namespace	Family	Feature
B01	Sunroof	yes	Teamcenter	PCA_Sunroof_Required	PCA_Sunroof_Required
B01	Sunroof	no	Teamcenter	PCA_Sunroof_Required	_NO_SELECTION_

   In this case, the boolean family with the name **PCA_Sunroof** and the feature with the name **PCA_Sunroof** are allocated in Product Configurator.

   > Note:
   >
   > **_NO_SELECTION_** is a placeholder for migrating variant expressions with **no** option value.

10. To map classic variant data for options that have different string values such as **CD**, **DVD**, and **none**, specify the appropriate namespace, family name, and feature name from Product Configurator as follows:

    Example for different string values:

Item ID	Option	Value	Namespace	Family	Feature
B01	MusicSystem	CD	Teamcenter	PCA_MusicSystem	PCA_CD
B01	MusicSystem	DVD	Teamcenter	PCA_MusicSystem	PCA_DVD
B01	MusicSystem	none	Teamcenter	PCA_MusicSystem	_NO_SELECTION_

    In this case, the family with the name **PCA_MusicSystem** and features with the names **PCA_CD** and **PCA_DVD** are allocated in Product Configurator.

    > Note:
    >
    > **_NO_SELECTION_** is a placeholder for migrating variant expressions with the option value **none**.

11. Save the CSV mapping file after completing all the required entries for mapping.

12. Run the migration utility by specifying the CSV mapping file.

    variant_migration -u=*user_ID* **-p=***password*
    -g=*dba* **-xml_file=***C:\Migration\B01_VariantAnalysisReport_44e8a83d.xml*
    -mapping_file=*C:\Mapping\B01_MappingFile_44e8a83d.csv*

13. Open the **B01_CC_migrated** in Product Configurator.

Verify whether all options and values that are available in the configurator context are allocated as families and features with the prefix specified in the CSV mapping file.

In this example, the **PCA_** prefix is used for mapping values.

14. Verify whether fixed defaults, derived defaults, and rule checks have prefixes as specified in the CSV mapping file.

    Fixed defaults and derived defaults are available as part of configurator rules and rule checks are available as exclusion rules.

    In this example, the **PCA_** prefix is used for mapping values.

15. Migrate variant conditions.

    Run the migration utility to migrate variant conditions by specifying the CSV mapping file.

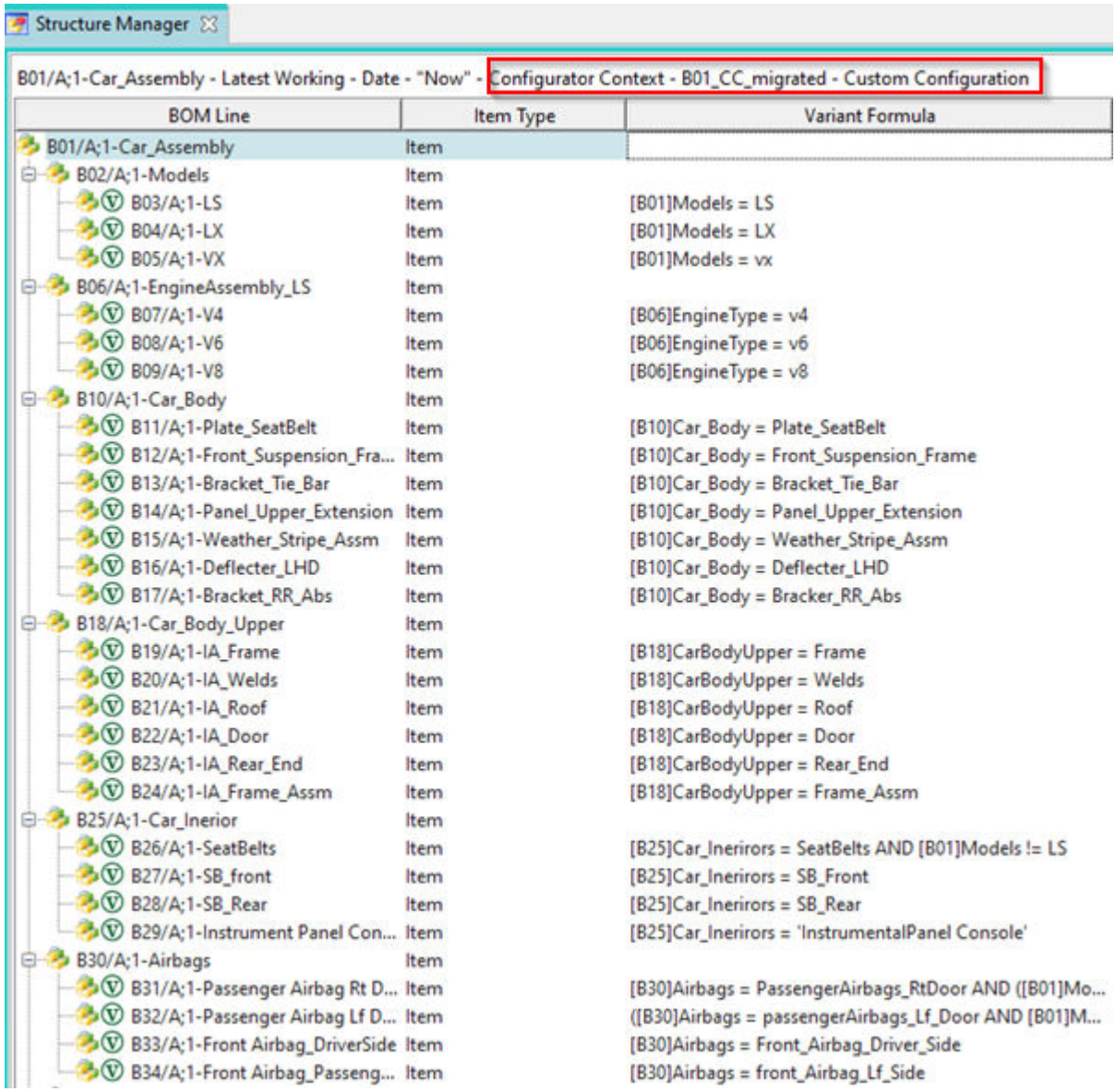
    variant_migration -u=*user_ID* **-p=***password* **-g=***dba*
    -xml_file=*C:\Migration\B01_IMC--1765912330_VariantConditionReport_365c20ec.xml*
    -createFormulaGrid -mapping_file=*C:\Migration\B01_MappingFile_44e8a83d.csv*
    -config_context=*B01_CC_migrated*

    > Note:
    >
    > Make sure that you use the same CSV mapping file for both variability and variant condition migration.

16. Execute the **Finalize** command to finalize the migration.

    variant_migration -u=*user_ID* **-p=***password* **-g=***dba*
    -xml_file=*C:\Migration\B01_VariantAnalysisReport_ef06121.xml* **-finalize**

17. Send the top node to Structure Manager.



18. Open the structure in the **Variant Configuration** view.

    Verify by selecting values, validating, and applying the configuration.

19. Verify whether variant conditions have the same prefix as specified in the CSV mapping file.

    In this example, the **PCA_** prefix is used for mapping values.

7. Migrating for a single site

Migrate variant data for a single site

After **extracting the variant data**, you run the **variant_migration** utility from a Teamcenter command prompt to migrate classic variability and classic variant conditions to Product Configurator. You must first migrate classic variability and then the variant conditions.

While migrating variability, the **variant_migration** utility creates one configurator context object for each owning item or each set from the input **VariantAnalysisReport** XML file.

After the migration is complete, the classic options, option values, fixed defaults, derived defaults, rule checks, and saved variant rules are migrated to the Product Configurator data model. The classic variant data that was migrated is not deleted.

1. To migrate classic variability and classic variant conditions, run the **variant_migration** utility by using the appropriate arguments.

Argument	Description
-u=	Specifies the user ID. If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.
-p=	Specifies the password.
-pf=	Specifies the password file path. Use this instead of the [**-p=***password*] argument for the password file path.
-xml_file=	Specifies the file path to the variant data extracted using the **variant_data_analysis** utility. You must specify the file path to the most recent XML file created by the **variant_data_analysis** utility. The **variant_migration** utility does not consider any changes made after the XML file is generated for migration. You must migrate variability first with the **VariantAnalysisReport** XML file. After the successful completion of variability migration, continue variant condition migration with the **VariantConditionReport** XML file as an input to the utility.
-mapping_file=	**Maps classic variability to Product Configurator variability**.

Argument	Description
	This file is prepopulated with classic variability data. You must edit the CSV mapping file to specify values for families and features in Product Configurator that correspond to classic options and values. When you specify the mapping file with the correct mapping values, families and features are retrieved from the global dictionary and allocated to the configurator context. If you do not provide the mapping file, families and features are created as new values.
	Parameters for variability migration
-migrateFamiliesOnly	Migrates only classic options to Product Configurator families. Other rules and constraints are skipped from migration. You must specify the **VariantAnalysisReport** XML file when you use this argument.
	Parameters for variant condition migration
-config_context=	Specifies the configurator context ID that has variability from the product.
-createFormulaGrid	Creates formula grids for variant conditions. You must specify the **VariantConditionReport** XML file when you use this argument.

2. Look up the log files for errors if the utility displays error messages. The log file path is specified in the error message.

3. **Validate the data to ensure that the migration was done correctly**.

   After you are satisfied with the data migration, you can clear the classic variant data and start using Product Configurator to configure structures.

> Caution:
>
> While migrating, no other application or process should access the variability data.

Arguments for migrating variability

Task	Argument
Migrate only families	**variant_migration -migrateFamiliesOnly -xml_file=***Complete_File_Path_to_VariantAnalysisReport*
Migrate all options, option values, defaults, derived	**variant_migration -xml_file=***Complete_File_Path_to_VariantAnalysisReport* You must specify the **VariantAnalysisReport** XML file first when you use this argument.

Task	Argument
defaults, rule checks, and SVRs	You must migrate variability first using the **VariantAnalysisReport** XML file and then migrate variant conditions using the **VariantConditionReport** XML file.
Migrate variant conditions	**variant_migration -xml_file=***Complete_File_Path_to_VariantConditionReport* You must specify the **VariantConditionReport** XML file when you use this argument.
Migrate variant conditions by creating formula grids	**variant_migration -createFormulaGrid -xml_file=***Complete_File_Path_to_VariantConditionReport* You must specify the **VariantConditionReport** XML file when you use this argument.

For more information about these arguments, type **variant_migration -h** on the Teamcenter command prompt.

Migration example for a single site

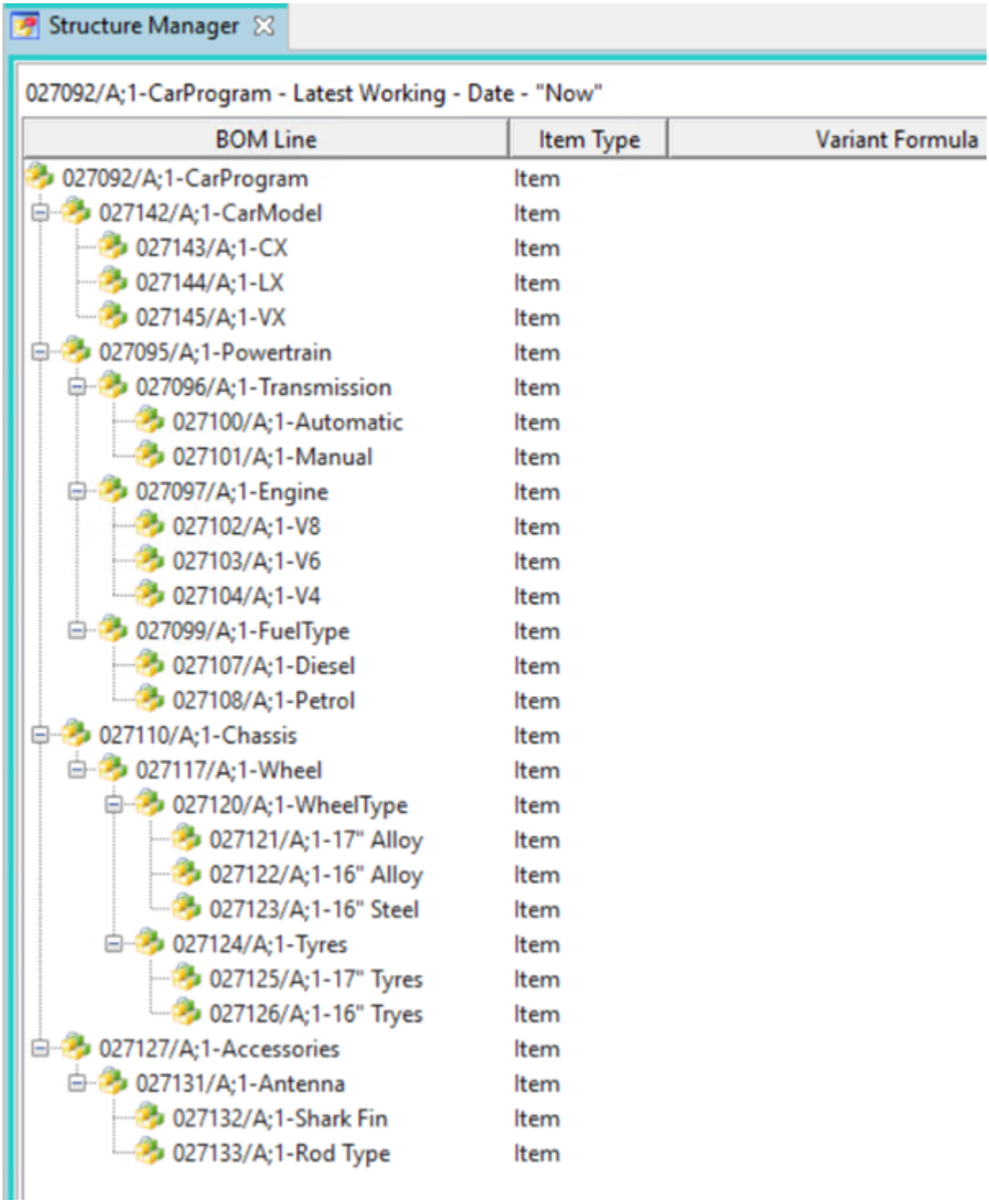
- Create a sample structure with variant data
- Extract the sample classic variant data
- Analyze the sample variability data
- Migrate the sample variability data
- Verify the migrated data

Create a sample structure with variant data

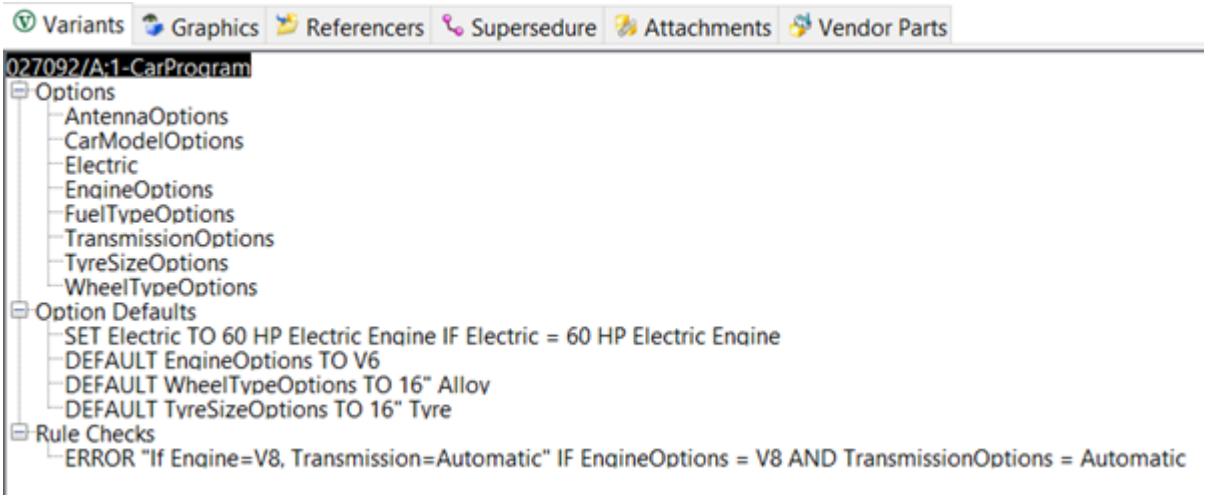
The following is an example of creating a sample structure with variant data that includes options, option values, option defaults, rule checks, variant conditions, and saved variant rules (SVRs).

Structures can also have external options and global options. These are not discussed in this migration example. In addition, stored option sets are not supported for migration.

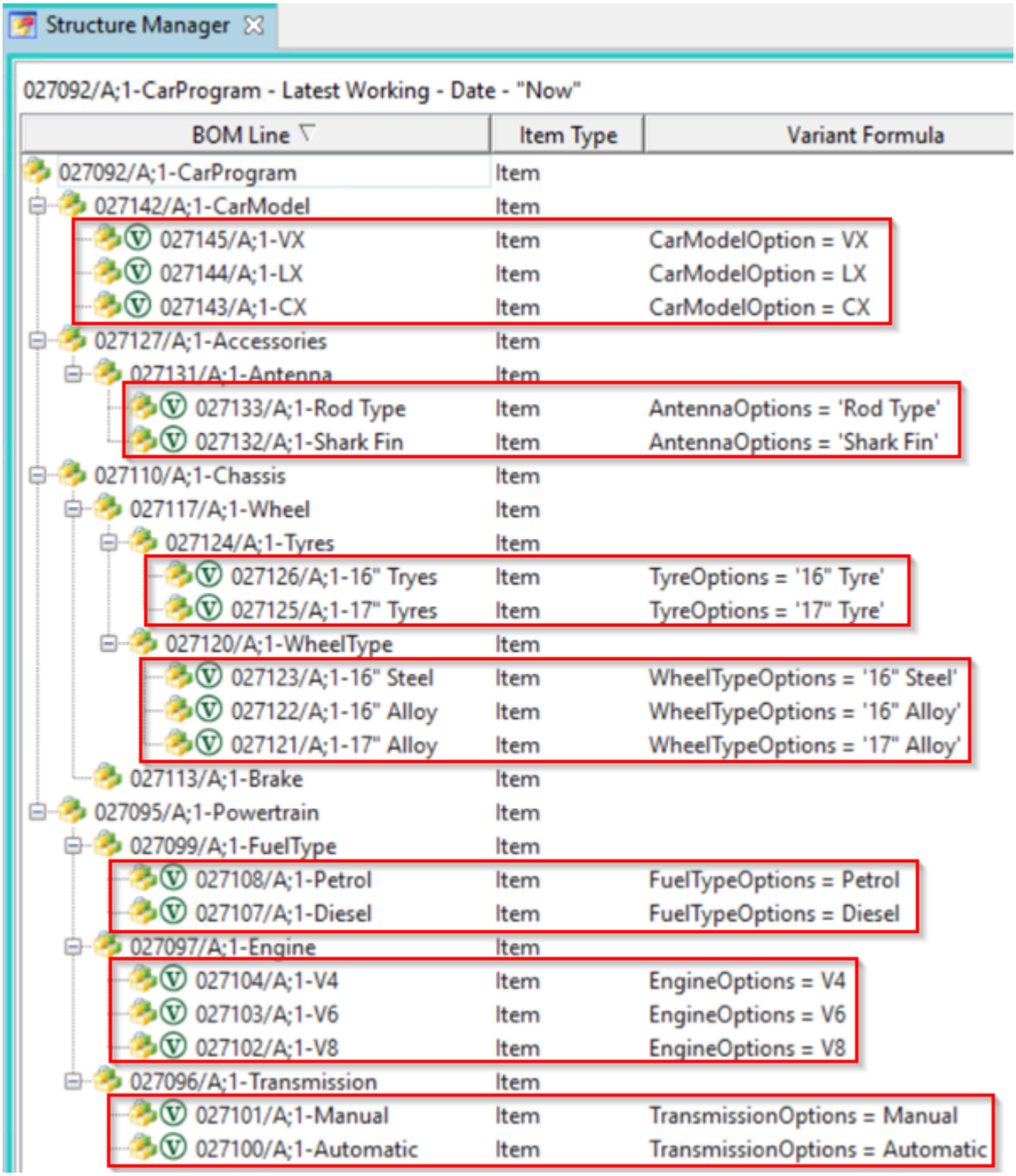
1. Create a structure as follows:



2. Select the root level item and click **Show/Hide Data Panel** on the main toolbar, and create the following options, option defaults, and rule checks in the **Variants** tab:

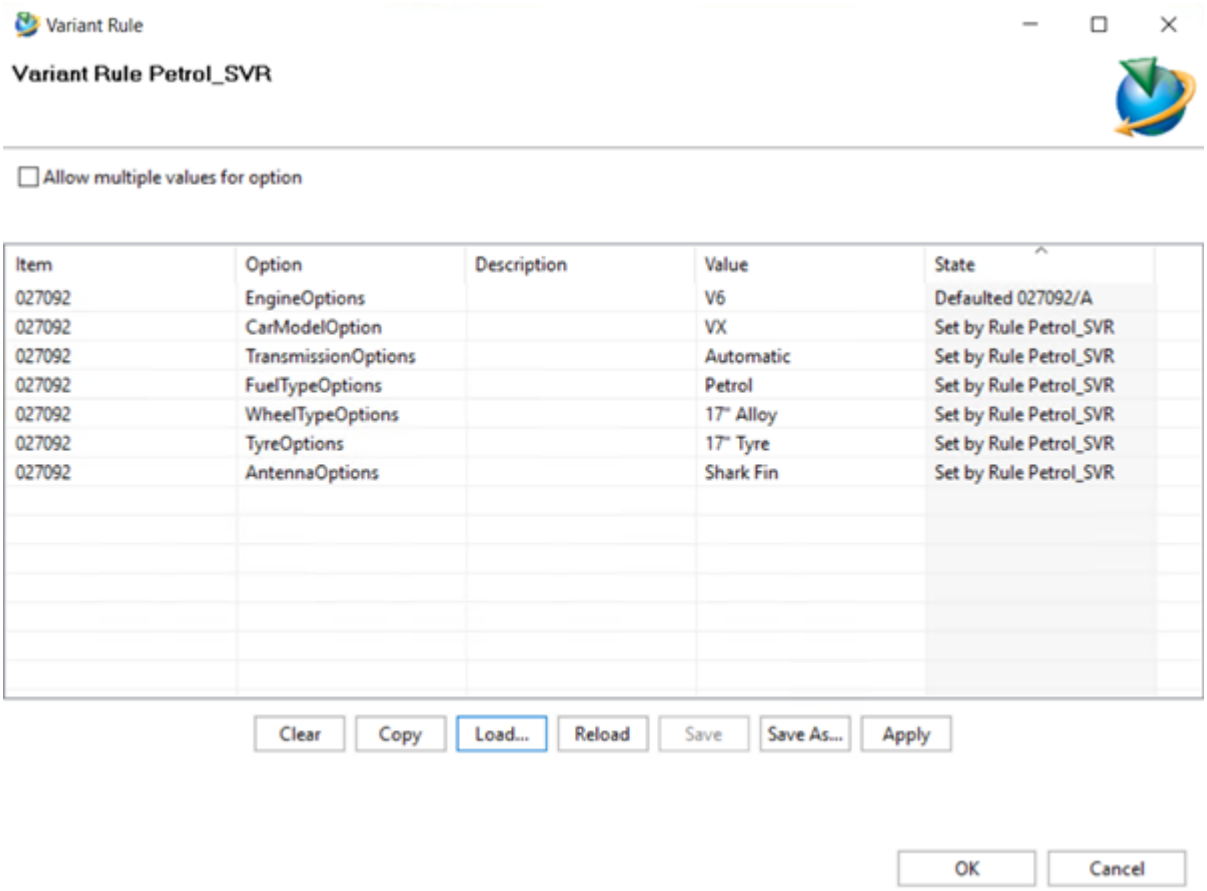


3. Click **Edit the variant condition** on the main toolbar, and create variant conditions as follows:

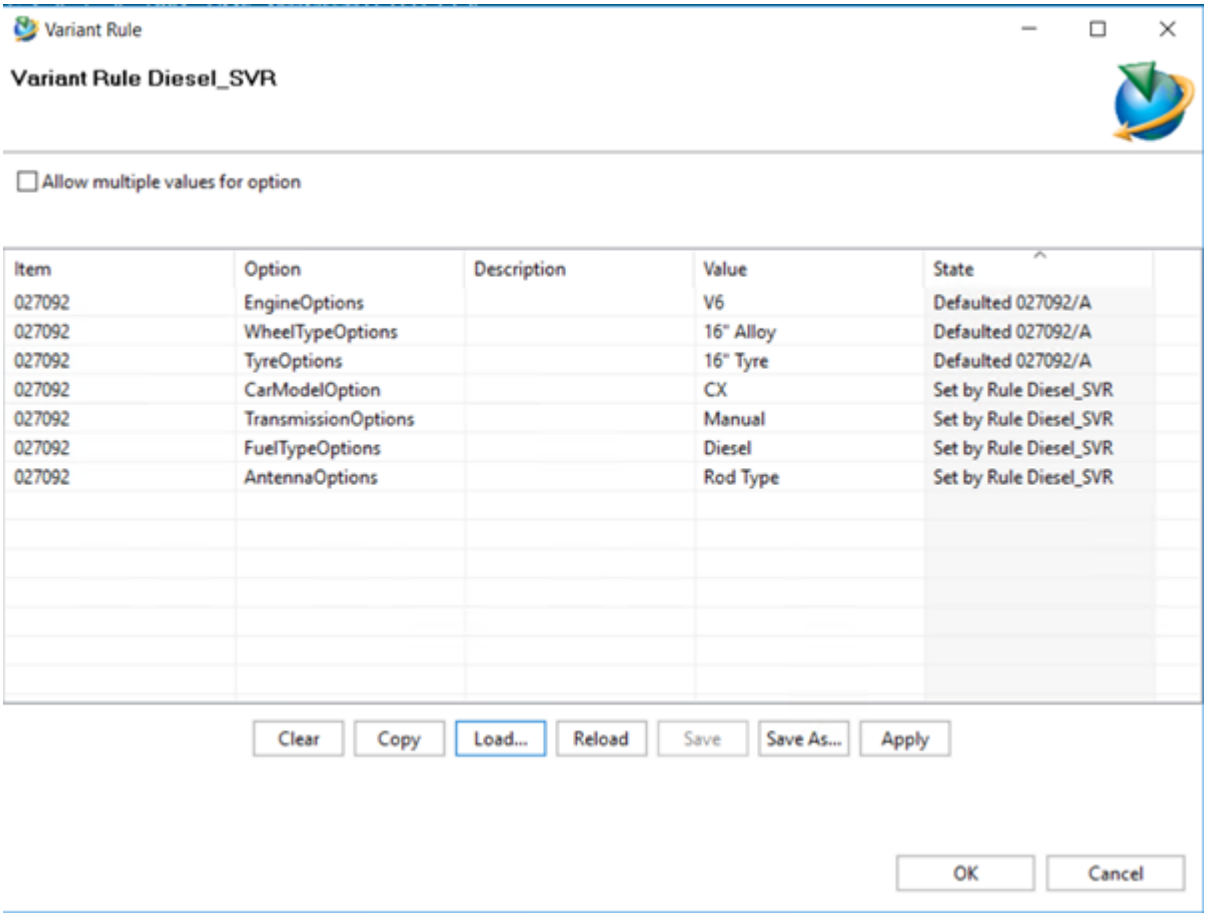


4. Click **Set option values for the selected module** on the main toolbar and create two SVRs as follows.

   Petrol SVR:



Diesel SVR:



Extract the sample classic variant data

You can use the **variant_data_analysis utility** to generate the UIDs of the owning items in a text file.

The first step is to generate the UID text file and use this file to extract the variant data.

1. To generate the UID text file for a product item, run this utility from a Teamcenter command prompt with the following commands:

   variant_data_analysis -item=*027092* **-uidFile=***C:\Users\Migration\027092.txt*

   In this example, the item ID of the car structure is *027092*.

   The generated UID text file has only one owning item as all the variability is created on the root item in the structure.

2. To extract both the variability and variant conditions, run this utility with the following commands:

   variant_data_analysis -item=*027092* **-uidFile=***027092.txt* **-xmlFilePath=***C:\Users\Migration* **-mode=***Variability_VariantCondition*

In this example, the mode is specified as *Variability_VariantCondition* to extract both the variability and variant conditions.

When you run this command, the system creates a CSV file and two XML files, namely, **VariantAnalysisReport** and **VariantConditionReport**.

Analyze the sample variability data

You can use the CSV file generated by the system to analyze the variability data.

For each owning item in the generated UID text file, the CSV file displays the corresponding options.

1. Open the CSV file from the **C:\Users\Migration** directory.

   In this example, this is the directory where the system generates the UID text file, CSV file, and XML files after running the **variant_data_analysis** utility by using the appropriate commands.

2. **Analyze the variability**.

Migrate the sample variability data

After extracting the variant data, run the **variant_migration utility** from a Teamcenter command prompt to migrate classic variability and classic variant conditions to Product Configurator.

1. From the Teamcenter command prompt, change to the directory from where you ran the **variant_data_analysis** utility to extract the variant data.

2. To migrate all options, option values, defaults, derived defaults, rule checks, and SVRs, run this utility with the following command:

   variant_migration -u=*user_ID* **-p=***password*
   -xml_file=*Complete_File_Path_to_VariantAnalysisReport*

   You must specify the **VariantAnalysisReport** XML file first when you use this argument.

   In this example, the XML file path is **C:\Users\Migration**. The XML file names are **VariantAnalysisReport_2020513_163028** and **VariantConditionReport_2020513_163028**.

3. To migrate variant conditions, run this utility with the following command:

   variant_migration -xml_file=*Complete_File_Path_to_VariantConditionReport*

   You must specify the **VariantConditionReport** XML file when you use this argument.

4. Look up the log files for errors if the utility displays error messages. The log file path is specified in the error message.

Verify the migrated data

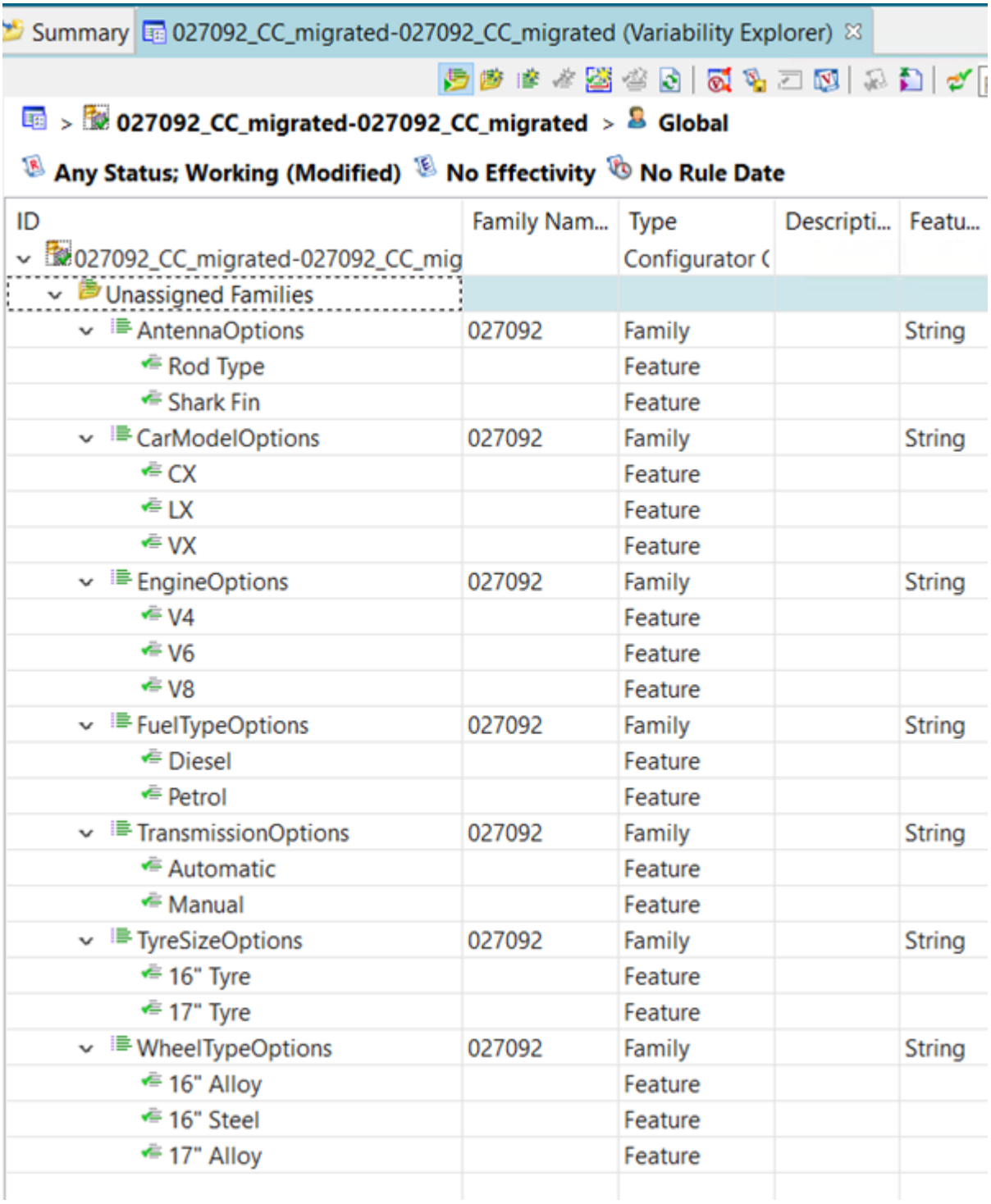
In this example, as all the variability was defined at the root level of the structure, the system creates only one configurator context.

To verify if the migration is successful, perform the following tasks:

1. To open the migrated configurator context, in My Teamcenter, search for the *product_item_ID*_**CC***. The system opens the migrated configurator context in a new view. Double-click it to open it in Product Configurator.

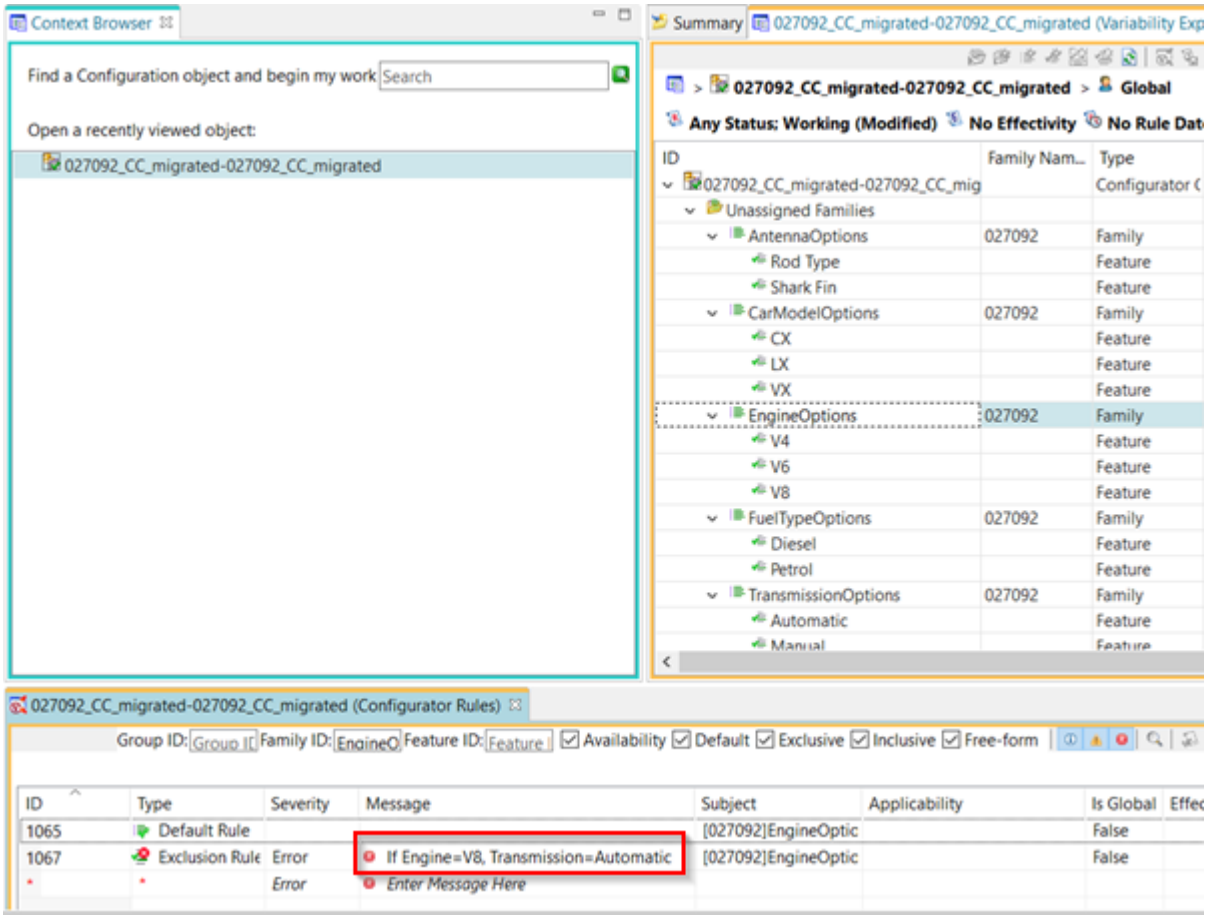
2. Click **Click to change the rule date on the configuration**, choose **Set Rule Date**, and click **No Date**.

3. To verify if all the options and allowed values are migrated, expand all the families and features in the configurator context.



ID	Family Nam...	Type	Descripti...	Featu...
027092_CC_migrated-027092_CC_mig		Configurator (		
Unassigned Families				
AntennaOptions	027092	Family		String
Rod Type		Feature		
Shark Fin		Feature		
CarModelOptions	027092	Family		String
CX		Feature		
LX		Feature		
VX		Feature		
EngineOptions	027092	Family		String
V4		Feature		
V6		Feature		
V8		Feature		
FuelTypeOptions	027092	Family		String
Diesel		Feature		
Petrol		Feature		
TransmissionOptions	027092	Family		String
Automatic		Feature		
Manual		Feature		
TyreSizeOptions	027092	Family		String
16" Tyre		Feature		
17" Tyre		Feature		
WheelTypeOptions	027092	Family		String
16" Alloy		Feature		
16" Steel		Feature		
17" Alloy		Feature		

4. To verify if the rule check is migrated, in the configurator context, click **Open Configurator Rules** view and select the **EngineOptions**. In this example, a rule check was created for the engine component **If Engine=V8, Transmission=Automatic**.



5. Copy the configurator context from the **Context Browser** view, open the top item of the structure in My Teamcenter, and paste the configurator context to the structure at the item level.

6. Right-click the top item of the structure in My Teamcenter and choose **Send To → Structure Manager**.

7. Click **Open Saved Variant Rules Dialog** on the main toolbar. In the **Load Variant Rules** dialog, specify an asterisk (*****) in the **Name** field and click **Search**.

   In this example, the system displays the two SVRs that were migrated.

In this example, on completion, all options and allowed values, option defaults, rule checks, SVRs, and variant conditions are migrated. This example did not have external options and global options. They are also migrated. Stored option sets, however, are not migrated.

8. Migrating for multisites

Migrate variant data for all sites

You can migrate classic variant data, including variant options, defaults, derived defaults, rule checks, and saved variant rules (SVRs), from Structure Manager to Product Configurator.

Migrating variant data for multisites is done in stages. The stages include analyzing variability, migrating options to families, migrating rules, and migrating variant conditions, and these stages have to be completed necessarily in this (specific) order. After each stage of the migration process, you must sync the data from the master site to the replica sites and vice versa before proceeding to the next stage. After completing all the stages, you can finalize the variant migration by attaching the configuration context to the structure. The finalized structure can be used with Product Configurator variants.

The **variant_data_analysis** and the **variant_migration** utilities create **.bat** (Windows) or **.sh** (UNIX and Linux) files at each stage of the migration process except at the initial stage when you run the analysis utility to create the UID file. Each of these batch files can be used to sync families across sites, sync rules and constraints across sites, migrate variant conditions, sync the structures after migrating variant conditions, or to sync the configurator context across sites. Syncing involves sending the data from the master site to the replica sites or vice versa by using the batch files that contain the **data_share** command.

The name of the batch **.bat** file starts with the top item ID of the structure to be migrated, followed by the site name, and the content of the batch file. If you use a UNIX or Linux system, refer to **.sh** files.

> Example:
>
> If you generate the variability XML, the file name is **000463_site2_XML_Generation_Command_4430e24e.bat**.
>
> Where **000463** is the top item ID of the structure to be migrated, **site2** the site name where the batch file has to be executed, **XML_Generation_Command** indicates the content or the purpose of the batch file, and **4430e24e** is the system log suffix.

If the batch file contains the **-p=** argument, you must edit the batch file to specify the password. Some batch files require that you specify the XML file path.

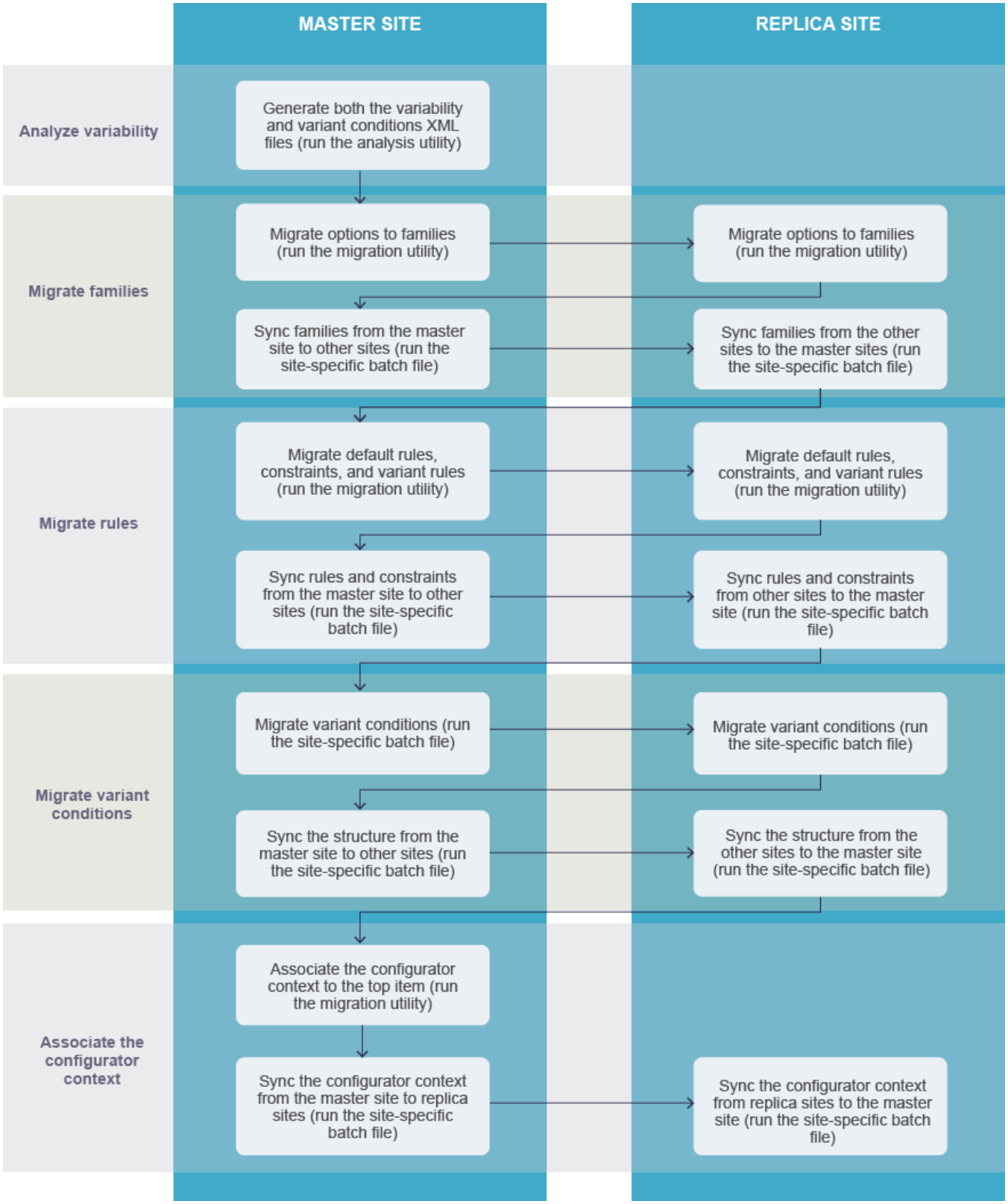
The following batch files are created by the utilities during the migration process. You can view the batch files and then run the site-specific batch file to complete the task.

Migration stage	Migration task	View the batch files or run the site-specific batch file
Analyze variability	Generate both the variability and variant conditions XML files by running the analysis utility.	• *itemID*_**site2_XML_Generation_Command.bat** • *itemID*_**site1_Variant_Condition_Migration.bat**

Migration stage	Migration task	View the batch files or run the site-specific batch file
		• *itemID*_**site2_Variant_Condition_Migration.bat**
Migrate families	Migrate options to families by running the migration utility.	• *itemID*_**site1_CC_data_share.bat file** • *itemID*_**site2_CC_data_share.bat file**
	Sync families from the master site to other sites and vice versa by running the site-specific batch file.	1. Edit the *itemID*_**site1_CC_data_share.bat** file to specify the required parameters and run it. 2. Edit the *itemID*_**site2_CC_data_share.bat** file to specify the required parameters and run it.
Migrate rules	Migrate default rules, constraints, and variant rules by running the migration utility.	• *itemID*_**site1_CC_data_share.bat file** • *itemID*_**site2_CC_data_share.bat file**
	Sync rules and constraints from the master site to other sites and vice versa by running the site-specific batch file.	1. Edit the *itemID*_**site1_CC_data_share.bat** file to specify the required parameters and run it. 2. Edit the *itemID*_**site2_CC_data_share.bat** file to specify the required parameters and run it.
Migrate variant conditions	Migrate the variant conditions by running the site-specific batch file.	1. Edit the *itemID*_**VariantConditionMigration_site1** file to specify the required parameters and run it. The utility creates the *itemID*_**site1_BVR_data_share.bat** file. 2. Edit the *itemID*_**VariantConditionMigration_site2** file to specify the required parameters and run it. The utility creates the *itemID*_**site2_BVR_data_share.bat** file.
	Sync the structure from the master site to other sites by running the site-specific batch file.	1. Edit the *itemID*_**site1_BVR_data_share.bat** file to specify the required parameters and run it. 2. Edit the *itemID*_**site2_BVR_data_share.bat** file

Migration stage	Migration task	View the batch files or run the site-specific batch file
		to specify the required parameters and run it.
Associate the configurator context	Associate the configurator context to the top item by running the migration utility using the **finalize** command.	• *itemID*_**site1_finalize_item_data_share.bat** • *itemID*_**site2_finalize_item_data_share.bat**
	Sync the configurator context from the master site to replica sites and vice versa by running the site-specific batch file.	1. Edit the *itemID*_**site1_finalize_item_data_share.bat** file to specify the required parameters and run it. 2. Edit the *itemID*_**site2_finalize_item_data_share.bat** file to specify the required parameters and run it.

The process for migrating variant data across sites is as follows:



Arguments for migrating variant data

Task	Argument
Migrate only families	**variant_migration -u=***UserID* **-p=***password* **-g=***group* **-migrateFamiliesOnly -xml_file=***Complete_File_Path_to_VariantAnalysisReport*
Migrate all options, option values, defaults, derived defaults, rule checks, and SVRs	**variant_migration -u=***UserID* **-p=***password* **-g=***group* **-xml_file=***Complete_File_Path_to_VariantAnalysisReport* You must specify the **VariantAnalysisReport** XML file first when you use this argument. You must migrate variability first using the **VariantAnalysisReport** XML file and then migrate variant conditions using the **VariantConditionReport** XML file.
Migrate the external options and allocate them to the top configurator context	When you run the **variant_data_analysis** utility, it reports the number of external options. It also creates the *topItemID***_VC_Ext_Opt_UIDs_***system_log_suffix***.txt** file. This file can be used to migrate the external options and allocate it to the top configurator context. Example: **variant_migration -u=***UserID* **-p=***password* **-g=***group* **-optionsFile=***VC_Ext_Opt_UIDs_Text_File_created_by_Variant_Data_Analysis_Utility*
Migrate variant conditions	**variant_migration -u=***UserID* **-p=***password* **-g=***group* **-xml_file=***Complete_File_Path_to_VariantConditionReport* You must specify the **VariantConditionReport** XML file when you use this argument.
Migrate variant conditions by creating formula grids	**variant_migration -u=***UserID* **-p=***password* **-g=***group* **-createFormulaGrid -xml_file=***Complete_File_Path_to_VariantConditionReport* You must specify the **VariantConditionReport** XML file when you use this argument.
Finalize variant migration by attaching the configuration context	You can finalize variant migration by attaching the configuration context to the owning item so that the structure can be used with Product Configurator variants. When you run the **finalize** command, it creates a batch file. You must sync this batch file with other sites. For more details, see **Migration example for multisites**. Example:

Task	Argument
	variant_migration -u=*userID* **-p=***password* **-g=***group* **-finalize -xml_file=***C:\temp\00463_VariantAnalysisReport_4430e24e.xml*

For more information about these arguments, type **variant_migration -h** in the Teamcenter command prompt.

Migration example for multisites

The following is an example of migrating a structure with two subassemblies. One of the subassemblies is an owning item (*sa_owining_itemID*) from *Site1* and the other is a replica item (*sa_replica_itemID*) from *Site2*. The parent structure is an owning item (*pa_owning_itemID*) from *Site1*.

The **tmp** environment variable determines the folder created to store all the migration XML files and batch files. For more information about the migration process and the batch file naming convention, see **Migrate variant data for all sites**.

The process for migrating structures for multisites is as follows:

1. **Generate the variability XML file and the variant conditions XML file**
2. **Migrate options to families and update them across sites by syncing configurator contexts**
3. **Migrate rules and constraints and update them across sites by syncing configurator contexts**
4. **Migrate variant conditions and update them across sites by syncing BVRs**
5. **Associate the configurator context to structures using the finalize command and sync them across sites**

Generate the variability XML file and the variant conditions XML file

Step	Site1	Site2
1. Run the variant_data_analysis utility	**variant_data_analysis -itemID=***pa_owning_itemID* **-uidFile=***itemID.txt* Where **-itemID=** is the top item ID of the structure to be migrated. The utility displays the number of owning item IDs from the variability analysis, number of owning items in variant conditions, and the number of external options found in variant conditions.	

Step	Site1	Site2
	It creates the required UID files.	
2. Generate the variability XML file	**variant_data_analysis** **-u=***userID* **-p=***password* **-g=***group* **-uidFile=***itemID.txt* **-xmlFilePath=***C:\temp* The utility generates the following files: • *pa_owning_itemID*_**VariantAnalysisReport_4430e24e.xml** variability XML file • *pa_owning_itemID*_**site2_XML_Generation_Command_4430e24e.bat** You can use this batch file to create the variability XML file for site2.	
3) Execute the batch file on site2		To create the variability XML file, edit the **XML_Generation_Command** batch file generated on site1 to specify the password and the XML file path and run it on site2. Example: **variant_data_analysis** **-u=***userID* **-p=***password* **-g=***group* **-uidFile=***C:\temp\pa_owning_itemID_site2_OwningItemUIDs_4430e24e.txt* **-xmlFilePath=***C:\temp*
4) Generate the variant conditions XML files	**variant_data_analysis** **-u=***userID* **-p=***password* **-g=***group* **-item=***pa_owning_itemID* **-mode=***VariantCondition* **-xmlFilePath=***C:\temp* The utility creates a variant condition report file for site1 and another report for site2.	

Step	Site1	Site2
	• *pa_owning_itemID*_**VariantConditionReport_site1_9f802ee2.xml** Where *pa_owning_itemID* is the top item ID of the parent structure, **site1** is the site name, and **9f802ee2** is the system log suffix. • *sa_replica_itemID*_**VariantConditionReport_site2_9f802ee2.xml** It also creates two site-specific batch files. You can use this batch file to migrate variant conditions for that specific site. • *pa_owning_itemID*_**VariantConditionMigration_site1_Variant_-Condition_Migration_acc4ac25.bat** • *pa_owning_itemID*_**VariantConditionMigration_site2_Variant_-Condition_Migration_acc4ac25.bat** Edit the batch files to specify the password.	

Migrate options to families and update them across sites by syncing configurator contexts

Step	Site1	Site2
1) Migrate options to families for site1	**variant_migration -u=***userID* **-p=***password* **-g=***group* **-migrateFamiliesOnly -xml_file=***Complete_Path_to_the_Variant_Analysis_Report_XML_File* The utility creates the *pa_owning_itemID*	

Step	Site1	Site2
	_site1_CC_data_share_a9e8746b.bat file. Use this batch file to sync data to replica sites.	
2) Migrate options to families for site2		1. Run the analysis utility. **variant_data_analysis -item=***sa_replica_itemID* **-xmlFilePath=***C:\Temp* It creates the *sa_replica_itemID*_**VariantAnalysisReport_b2f88c14.xml** variant analysis report file. 2. Run the migration utility to migrate families only. **variant_migration-u=***userID* **-p=***password* **-g=***group* **-migrateFamiliesOnly -xml_file=***Complete_Path_to_the_Variant_Analysis_Report_XML_File* The utility creates the *sa_replica_itemID*_**site2_CC_data_share_b2f88c14.bat** file. Use this batch file to sync data to the master site.
3) Update families across sites by syncing configurator contexts from site1 to site2	Edit the *pa_owning_itemID*_**site1_CC_data_share_b2f88c14.bat** file to specify the password and run it. The **data_share** utility syncs the configurator context.	
4) Update families across sites by syncing configurator contexts from site2 to site1		Edit the *sa_replica_itemID*_**site2_CC_data_share_a9e8746b.bat** file to specify the password and run it.

Step	Site1	Site2
		The **data_share** utility syncs the configurator context.

Migrate rules and constraints and update them across sites by syncing configurator contexts

Step	Site1	Site2
1) Migrate default rules, constraints, and variant rules for site1	**variant_migration -u=***userID* **-p=***password* **-g=***group* **-xml_file=***Complete_Path_to_the_Variant_Analysis_Report_XML_File* The utility creates the *pa_owning_itemID* **_site1_CC_data_share_b994d280.bat** Use this batch file to sync data to replica sites.	
2) Migrate default rules, constraints, and variant rules for site2		**variant_migration -u=***userID* **-p=***password* **-g=***group* **-xml_file=***Complete_Path_to_the_Variant_Analysis_Report_XML_File* The utility creates the *sa_replica_itemID* **_site2_CC_data_share_b994d280.bat** file. Use this batch file to sync data to the master site.
3) Update default rules, constraints, and variant rules from site1 to site2 by syncing the configurator contexts	Edit the *pa_owning_itemID* **_site1_CC_data_share_b994d280.bat** file to specify the password and run it. The **data_share** utility syncs the configurator context.	
4) Update default rules, constraints, and variant rules from site2 to site1 by syncing the configurator contexts		Edit the *sa_replica_itemID***_site2_CC_data_share_b994d280.bat** file to specify the password and run it. The **data_share** utility syncs the configurator context.

Migrate variant conditions and update them across sites by syncing BVRs

Step	Site1	Site2
1) Migrate variant conditions on site1	**Run the *pa_owning_itemID*_VariantConditionMigration_site1_Variant_Condition_Migration_acc4ac25.bat file created at site1.** If the top item ID has thousands of BVRs that are local to that site, they are listed in the *pa_owning_itemID* **_BVR_UIDS_file_b944600.txt** file. If the list of BVRs is very large, you can split them into multiple files using a third party split utility. When you split files, you must add all the file names as commands to the **BVR_data_share** batch file. It creates the *pa_owning_itemID*_**site1_BVR_data_share_b944600.bat** file. Use this batch file to sync BVRs to replica sites.	
2) Migrate variant conditions on site2		**Run the *pa_owning_itemID*_VariantConditionMigration_site2_Variant_Condition_Migration_acc4ac25.bat file created at site1.** It creates the *sa_replica_itemID* **_site2_BVR_data_share_b944600.bat** file. Use this batch file to sync BVRs to the master site.
3) Update variant conditions from site1 to site2 by syncing the BVRs	Edit the **C:\Temp\00463_site1_BVR_data_share_b944600.bat** file to specify the password and run it. The BVR IDs are sent to site2.	
4) Update variant conditions from site2 to site1 by syncing the BVRs		Edit the **C:\Temp\00471_site2_BVR_data_share_b944600.bat** file to specify the password and run it.

Step	Site1	Site2
		The BVR IDs are sent to site1.

Associate the configurator context to structures using the finalize command and sync them across sites

Step	Site1	Site2
1) Associate the configurator context to the owing item structure for site1 using the finalize command	**variant_migration -u=***userID* **-p=***password* **-g=***group* **-finalize -xml_file=***Complete_Path_to_the_Variant_Analysis_Report_XML_File* This is the variant analysis report that was created for site1 when you **generated the variability XML file**. It creates the *itemID* **_site1_finalize_item_data_share_b904f6ae.bat** file. Use this batch file to sync configurator contexts to replica sites.	
2) Associate the configurator context to the replica structure for site2 using the finalize command		**variant_migration -u=***userID* **-p=***password* **-g=***group* **-finalize -xml_file=***Complete_Path_to_the_Variant_Analysis_Report_XML_File* This is the variant analysis report that was created for site2 when you **generated the variability XML file**. It creates the *itemID* **_site2_finalize_item_data_share_b904f6ae.bat** file. Use this batch file to sync configurator contexts to the master site.

Step	Site1	Site2
3) Sync the associated configurator context from site1 to site2	To sync the data, run the *itemID* **_site1_finalize_item_data_share_b904f6ae.bat** batch file.	
4) Sync the associated configurator context from site2 to site1		To sync the data, run the *itemID* **_site2_finalize_item_data_share_b904f6ae.bat** batch file.