MODEL FORGE

CONNECTOR IMPORT TOOL

-USER GUIDE-

Table of Contents

INTRODUCTION **3**

COMPATIBILITY **3**

INSTALLATION **3**

PLUGIN USES AND OPERATIONS **3**

BASIC OPERATIONS AND COMMON FUNCTIONS 3

Opening Main Dialog *3*

Setting Tool Scope *4*

Defining Connectors in Excel *5*

USE CASE 1: IMPORT CONNECTORS WITH NESTED PORTS 6

USE CASE 2: IMPORT CONNECTORS WITHOUT NESTED PORTS 11

TROUBLESHOOTING AND DEBUGGING **13**

Table of Figures

Figure 1 Plugin Main Dialog 4

Figure 1 Plugin Scope Field and Controls 5

Figure 3 Adding or Removing Package To or From Plugin Scope 5

Figure 4 Example Connector Definition in Excel 6

Figure 5 Example System Hierarchy 7

Figure 6 Adding Package to Plugin Scope 8

Figure 6 Example Model with Plugin Scope Root Package Chosen 9

Figure 8 Window Prompt To Select Which Column Contains The Required Information 9

Figure 9 Plugin Representation of Column Mapping 9

Figure 10 Example Results of Import – Nested Ports 10

Figure 10 Example of Imported Element Containment – Nested Ports 11

Figure 10 Example Results of Import – Interconnects 12

Figure 10 Example of Imported Element Containment - Interconnects 12

Figure 14 Debug Log Button 13

Figure 15 Example Debug Log 13

Introduction

The Connector Import Tool was born out of a frustration with the lack of ability to easily import SysML connectors and associated elements (i.e. ports, item flows, etc.) into a SysML model from external sources. This is an open-source project so please add your valuable contributions at the link below!

https://github.com/Model-Forge/ConnectorImport

Compatibility

The Connector Import Tool is a MagicDraw product line (e.g. Cameo Systems Modeler) plugin that is currently only known to be compatible with the 2021x Refresh 2 version of the tool, but future testing may reveal the extent of any backwards compatibility.

Additionally, the plugin can, at this time, only import connector definitions from Microsoft Excel files, whose extension is '.xlsx'.

Installation

The plugin does not currently support import or installation via the Resource / Plugin Manager available in the MagicDraw / Cameo user interface and consequently is done entirely manually. To install the plugin perform the following:

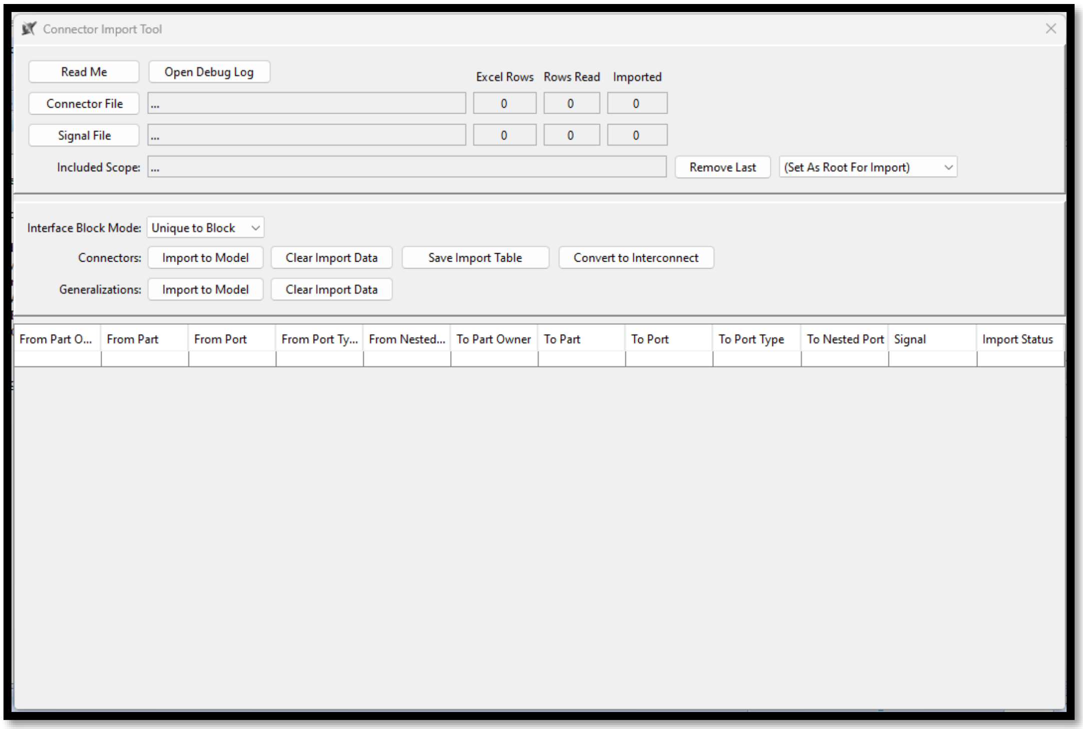
1. Download the plugin and unzip (if necessary) into a convenient directory.
2. Create a new folder/directory with the name ConnectorImportTool at the following location:
    a. \{MagicDraw Install Directory}\plugins
3. Select and copy (Crtl + C) the following files in the directory:
    a. ConnectorImportPlugin-X.X.X-FINAL.jar
    b. ConnectorImportTool Read Me.pdf
    c. plugin.xml
4. Paste (Ctrl+V) the copied files into the new folder created in step 2

Plugin Uses and Operations

Basic Operations and Common Functions

Opening Main Dialog

Select the "Tools" drop down menu from the MagicDraw application top row and click the "Connector Tool – Import Dialog" button to view the plugin main dialog.



		Excel Rows	Rows Read	Imported
Connector File	...	0	0	0
Signal File	...	0	0	0

Included Scope: ... (Set As Root For Import)

Interface Block Mode: Unique to Block

Connectors:

Generalizations:

From Part O...	From Part	From Port	From Port Ty...	From Nested...	To Part Owner	To Part	To Port	To Port Type	To Nested Port	Signal	Import Status

Figure 1 Plugin Main Dialog

Note that closing the dialog by clicking the "X" in the upper right hand corner closes the main dialog, but does not clear any of the data in the plugin. Should the user need to reset the data in the plugin either press the "Clear Import Data" buttons or press the "Connector File" button to begin a new data import from an excel file. Closing the model project also does not clear out the import data, however closing the MagicDraw / Cameo application does.

Setting Tool Scope

The plugin is loaded when the MagicDraw application is booted and remains operating until the MagicDraw application is finally closed. When certain plugin functions are executed the user can observe progress via either a progress bar (available during connector import), statements written to the application notification window, or more detailed data written to a log file. See Troubleshooting and Debugging for more information.

The plugin requires the user to select the portion(s) of the SysML model that contain the system structure and related elements. This allows the user to maintain multiple system hierarchies or usages of various blocks and import connectors for use with a specific system hierarchy or structure. The user does this by selectively adding Packages in the model containment tree to the plugin's scope until all the packages containing the model elements associated with the connectors to be imported are listed in the plugin main dialog's "Included Scope" field (see Figure 2).

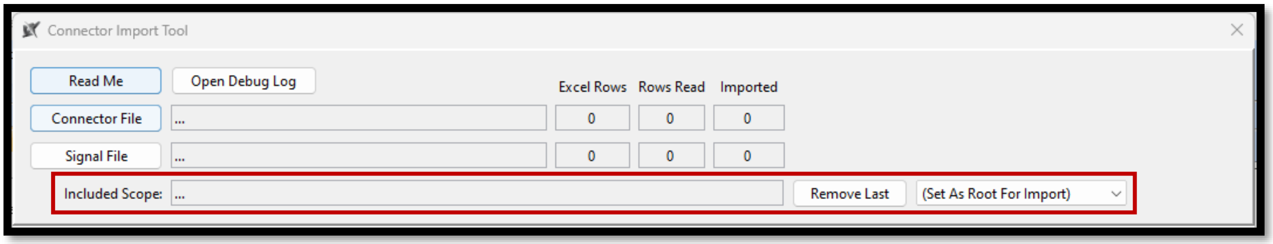


Figure 2 Plugin Scope Field and Controls

Packages are added to the plugin scope by selecting the desired package(s), one at a time, right click on the package, and press the "Connector Tool (Add Package To Scope)" button on the drop down menu. Similarly, if the user were to accidentally add a package to scope, select the package to be removed, right click, and press the "Connector Tool (Remove From Scope)" button on the resulting drop down menu (See Figure 3).

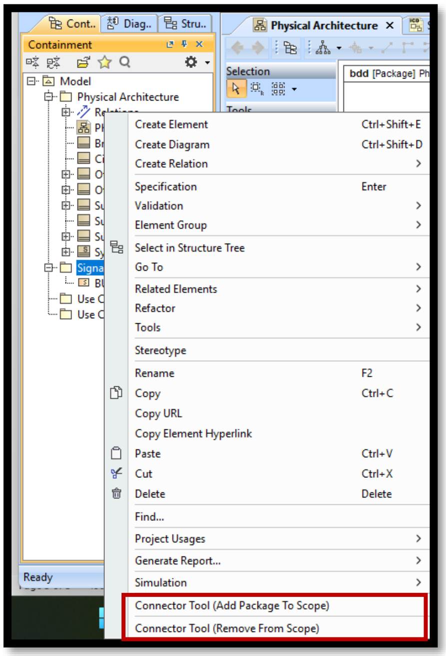


Figure 3 Adding or Removing Package To or From Plugin Scope

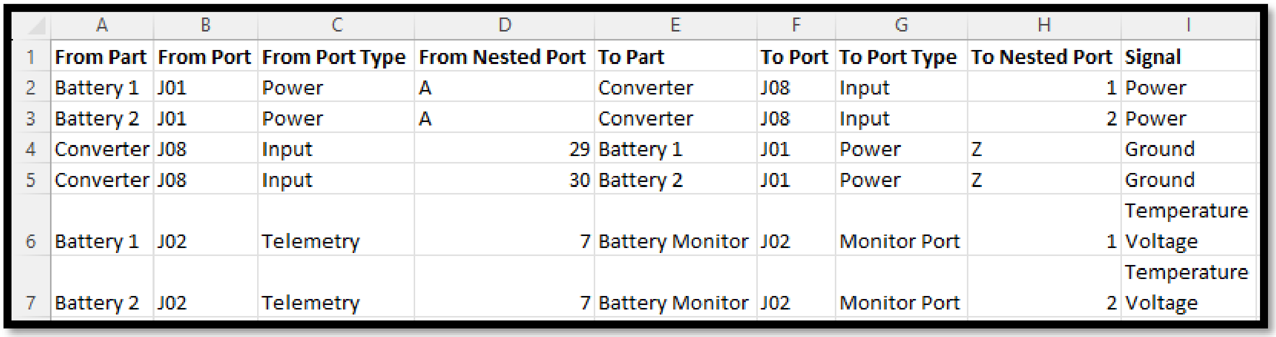
Lastly, the last package listed in the scope list can be removed by pressing the "Remove Last" button available on the main dialog (See Figure 2). The "Set As Root For Import" drop down menu (see Figure 2) allows the user to select a package from the included scope to use as the destination package for imported model elements that don't get assigned an owner as part of the element creation process (e.g. signals).

Defining Connectors in Excel

The plugin allows the user to import an excel file (.xslx format only) where each row represents a connector and an item flow realized by that connector. The necessary information, as shown below, should be contained in the columns (column headers not required and column names can be anything the user wants):

- From Part Name
- From Part Owner Name (optional – only required where part names are not unique)
- From Port Name
- From Port Type Name
- From Nested Port Name
- To Part Name
- To Part Owner Name (optional – only required where part names are not unique)
- To Port Name
- To Port Type Name
- To Nested Port Name

The columns do not need to be in any particular order and don't even need to be contiguous. The only restriction, is if the connector information is spread across multiple tabs in the excel file, the column locations must be the same across all tabs. For example, if in tab 1, the "From Port" is in column D, the "From Port" information in all other tabs must also be in column D. Below is an example of an excel file with the necessary information.



	A	B	C	D	E	F	G	H	I
1	**From Part**	**From Port**	**From Port Type**	**From Nested Port**	**To Part**	**To Port**	**To Port Type**	**To Nested Port**	**Signal**
2	Battery 1	J01	Power	A	Converter	J08	Input	1	Power
3	Battery 2	J01	Power	A	Converter	J08	Input	2	Power
4	Converter	J08	Input	29	Battery 1	J01	Power	Z	Ground
5	Converter	J08	Input	30	Battery 2	J01	Power	Z	Ground
6	Battery 1	J02	Telemetry	7	Battery Monitor	J02	Monitor Port	1	Temperature Voltage
7	Battery 2	J02	Telemetry	7	Battery Monitor	J02	Monitor Port	2	Temperature Voltage

Figure 4 Example Connector Definition in Excel

If the connectors to be imported DO NOT connect nested ports as shown in Figure 4, then simply omit the "From Nested Port" and "To Nested Port" and when importing the file in the plugin, tell the plugin those columns don't exist.

The Signal column is unique in that it can interpret multiple signal names in the same cell that are separated by carriage return (pressing Alt + Enter while typing in the cell) and import / realize a unique Item Flow for each on the connector defined in that row. In Figure 4, Row 4 would realize an Item Flow conveying the signal named Temperature and an Item Flow conveying a Signal named Voltage on the Connector defined in Row 4. The direction of the flow is inferred from the name in the "From Part" column as being the Item Flow source and the name in the "To Part" column as the Item Flow target.

Use Case 1: Import Connectors With Nested Ports

Use Case Description: As a modeler I want to automatically create connectors from the information stored in a Microsoft Excel file that defines SysML connectors where the connector ends are nested ports.

Pre-Conditions:

- Model project open in MagicDraw / Cameo Systems Modeler
- Basic structure (parts and part types) exists
- Microsoft Excel file (.xslx) as described in “Defining Connectors in Excel”

Narrative:

1. Given the basic hierarchy of the system (example below in Figure 5) exists and the excel file with the necessary connector definitions and signal flows exist (see Figure 4), the user adds the packages that contain the system hierarchy and any existing elements such as signals or interface blocks to be used to the plugin scope by right clicking on each package in the containment tree and selecting “Connector Tool (Add Package To Scope).”

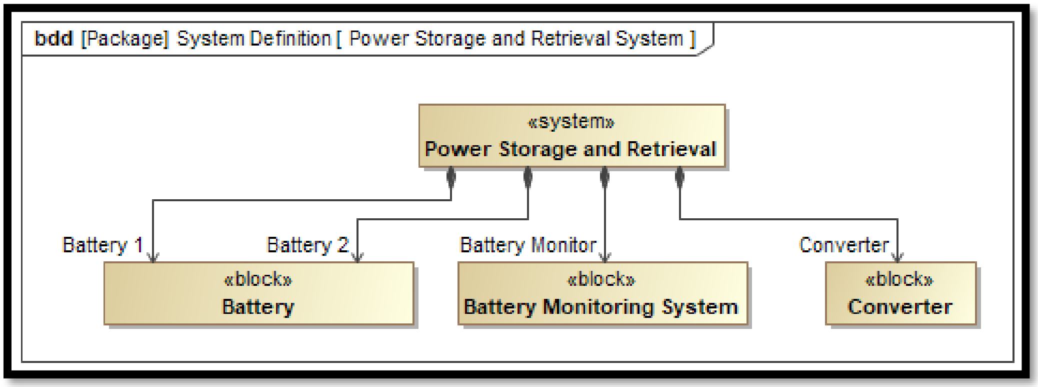

bdd [Package] System Definition [Power Storage and Retrieval System]
«system»
Power Storage and Retrieval
Battery 1
Battery 2
Battery Monitor
Converter
«block»
Battery
«block»
Battery Monitoring System
«block»
Converter

Figure 5 Example System Hierarchy

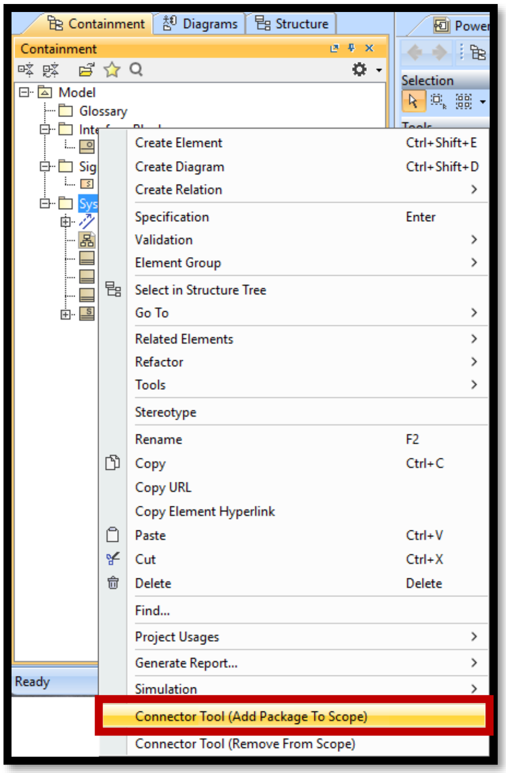
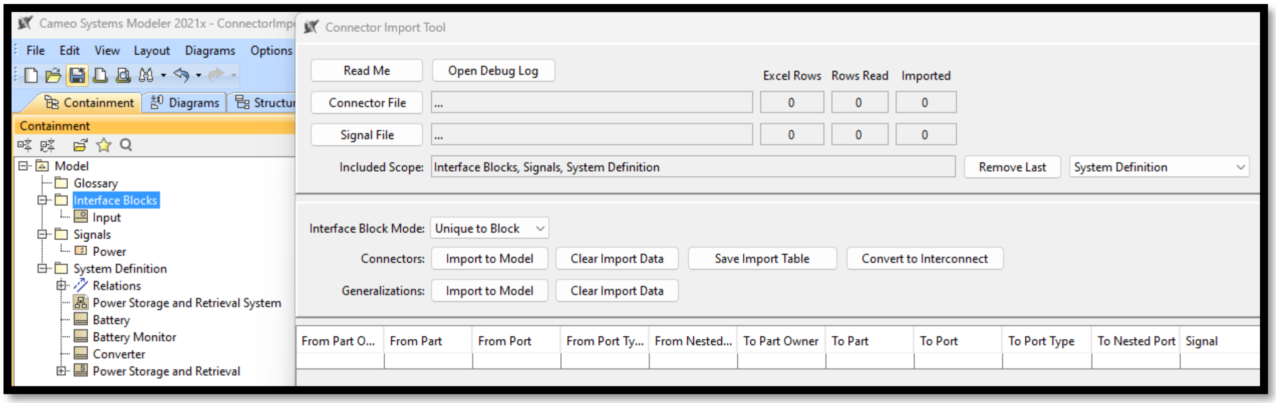


Figure 6 Adding Package to Plugin Scope

2. Once all the appropriate packages have been added to the plugin scope, the user navigates to the “Tools” drop down menu from the MagicDraw / Cameo application top row and click the “Connector Tool – Import Dialog” button to view the plugin main dialog. The user observes the correct packages were added to the “Included Scope” field (see Setting Tool Scope for removing packages from scope) and selects the desired folder where other elements such as Interface Blocks and Signals should be stored following creation by selecting the in-scope package from the drop down menu named “(Set As Root For Import)”. If no choice is made, the plugin creates the elements in the first package in the in-scope list. In the example provided in Figure 7 notice the scope includes packages with existing Interface Blocks and Signals that now may be used by the plugin to type proxy ports and be conveyed by item flows as defined in the excel file.



		Excel Rows	Rows Read	Imported
Connector File	...	0	0	0
Signal File	...	0	0	0

Included Scope: Interface Blocks, Signals, System Definition

Interface Block Mode: Unique to Block

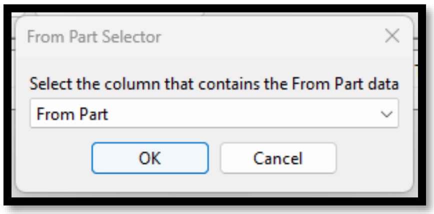
Connectors:

Generalizations:

From Part O...	From Part	From Port	From Port Ty...	From Nested...	To Part Owner	To Part	To Port	To Port Type	To Nested Port	Signal

Figure 7 Example Model with Plugin Scope Root Package Chosen

3. In the plugin main dialog, the user clicks the "Connector File" button to bring up the file chooser dialog and selects the excel file that provides the connector definition information. The "Sheet Selector" dialog opens where the user chooses which sheet contains the connector definitions or if all sheets are to be used, chooses the "(All Sheets)" option. Subsequent dialog windows give the user the ability to tell the plugin which column contains each of the different types of information defining the connector and item flows.

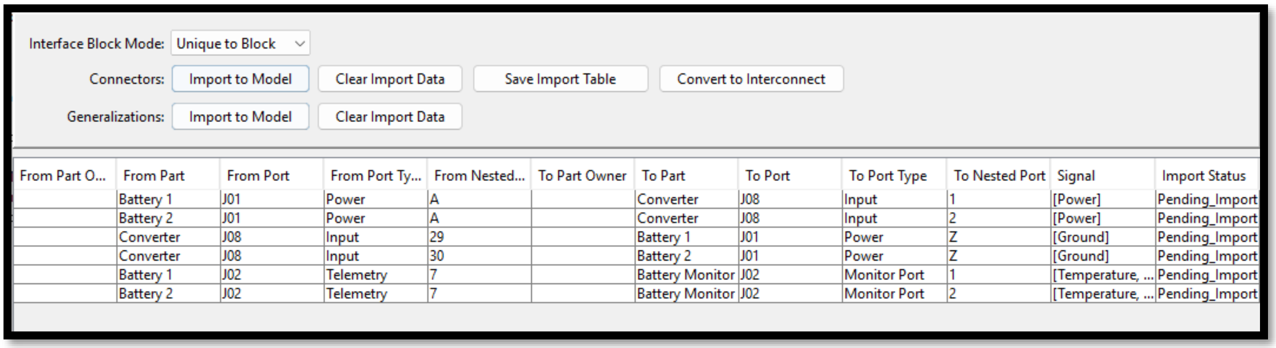


Select the column that contains the From Part data

From Part

Figure 8 Window Prompt To Select Which Column Contains The Required Information

4. After mapping each column to the required information, the user can observe how the plugin has understood the column mapping. If the user selected a column incorrectly the user will need to reperform the column mapping by reselecting the connector file and going through the process again.



Interface Block Mode: Unique to Block

Connectors:

Generalizations:

From Part O...	From Part	From Port	From Port Ty...	From Nested...	To Part Owner	To Part	To Port	To Port Type	To Nested Port	Signal	Import Status
	Battery 1	J01	Power	A		Converter	J08	Input	1	[Power]	Pending_Import
	Battery 2	J01	Power	A		Converter	J08	Input	2	[Power]	Pending_Import
	Converter	J08	Input	29		Battery 1	J01	Power	Z	[Ground]	Pending_Import
	Converter	J08	Input	30		Battery 2	J01	Power	Z	[Ground]	Pending_Import
	Battery 1	J02	Telemetry	7		Battery Monitor	J02	Monitor Port	1	[Temperature, ...	Pending_Import
	Battery 2	J02	Telemetry	7		Battery Monitor	J02	Monitor Port	2	[Temperature, ...	Pending_Import

Figure 9 Plugin Representation of Column Mapping

5. The user must choose whether each Interface Block should be created once and used to type each instance of the appropriate Proxy Port or to create a new Interface Block of that name for each Block that owns Proxy Ports typed by it. The latter should be chosen when multiple Proxy Ports owned by multiple Blocks are typed Interface Blocks of the same name, but who may have different Flow Properties or nested Proxy Ports. The user makes the choice by selecting either "Unique to Block" or "Common Types" in the "Interface Block Mode" drop down menu seen in Figure 9.
6. When satisfied with the status of the data mapping and other import choices, the user generates the Connectors, Interface Blocks, Item Flows, and Signals in the model by pressing the "Import to Model" button in the "Connectors" row as seen in Figure 9. A progress bar displays the progress of the import and if any connectors failed to import, the user is asked if they would like to save a spreadsheet containing the list of connectors that failed with a description of each failure somewhere. Otherwise, the progress bar is closed and the imported model elements can be observed in the model as seen in Figure 10.

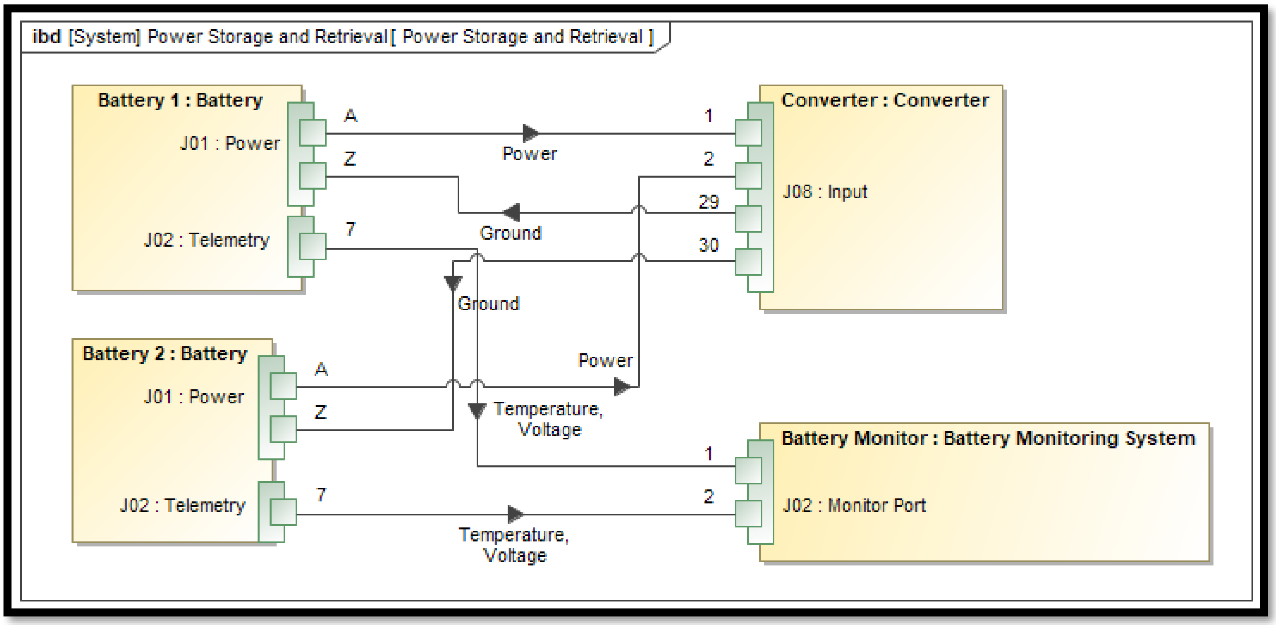


Figure 10 Example Results of Import – Nested Ports

Post-Conditions: The model elements have been created and are contained in the desired elements. Where SysML does not dictate where the elements are to be contained, the plugin has created packages to hold those elements. See the below figure as an example.

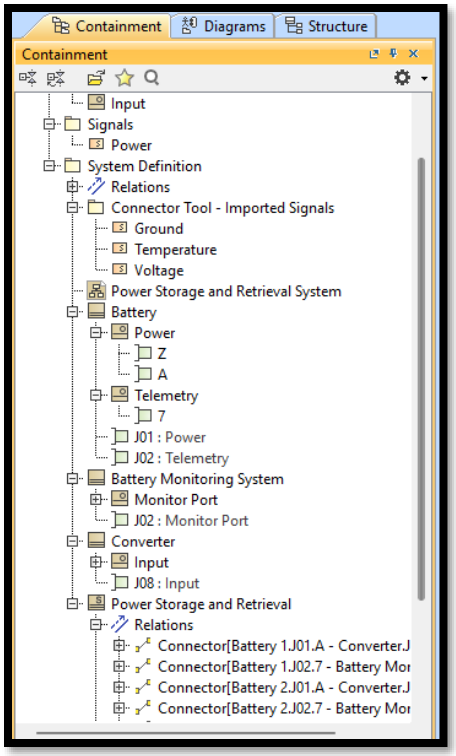

Containment
Diagrams
Structure
Containment
Input
Signals
Power
System Definition
Relations
Connector Tool - Imported Signals
Ground
Temperature
Voltage
Power Storage and Retrieval System
Battery
Power
Z
A
Telemetry
7
J01 : Power
J02 : Telemetry
Battery Monitoring System
Monitor Port
J02 : Monitor Port
Converter
Input
J08 : Input
Power Storage and Retrieval
Relations
Connector[Battery 1.J01.A - Converter.J
Connector[Battery 1.J02.7 - Battery Mor
Connector[Battery 2.J01.A - Converter.J
Connector[Battery 2.J02.7 - Battery Mor

Figure 11 Example of Imported Element Containment – Nested Ports

Use Case 2: Import Connectors Without Nested Ports

Use Case Description: As a modeler I want to automatically create connectors from the information stored in a Microsoft Excel file that defines SysML connectors where the connector ends are proxy ports that are not nested within other proxy ports.

Pre-Conditions:

- Model project open in MagicDraw / Cameo Systems Modeler
- Basic structure (parts and part types) exists
- Microsoft Excel file (.xslx) as described in “Defining Connectors in Excel”

Narrative: See Use Case 1: Import Connectors With Nested Ports with the following exceptions:

- Step 3: The user may ignore the “From Nested Port” and “To Nested Port” columns during the column mapping process.
- Step 6: Prior to pressing the “Import to Model” button, the user presses the “Convert to Interconnect” button and observes that the “From Nested Port” and “To Nested Port” columns in the plugin data map table are empty. Any duplicate connector definitions will have been collapsed down into a single connector row.

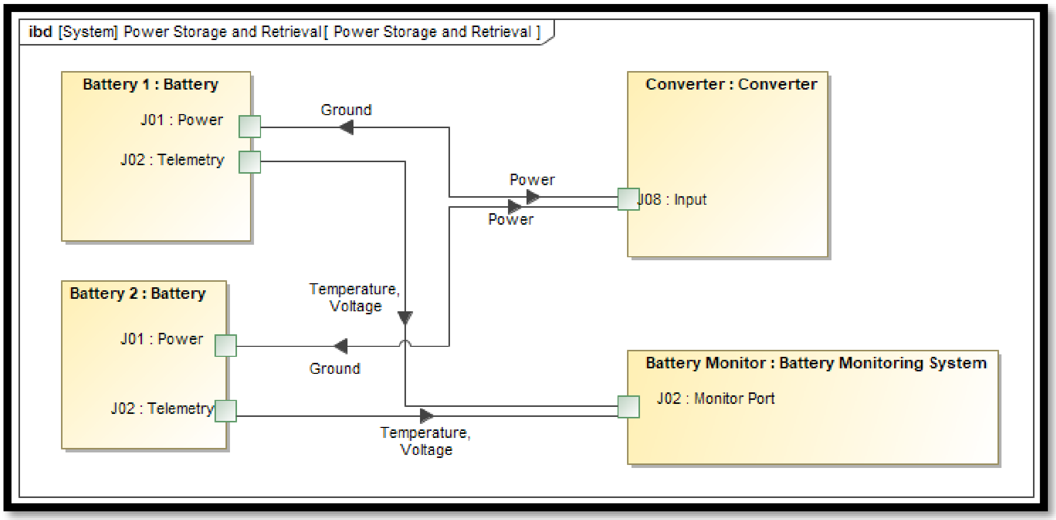
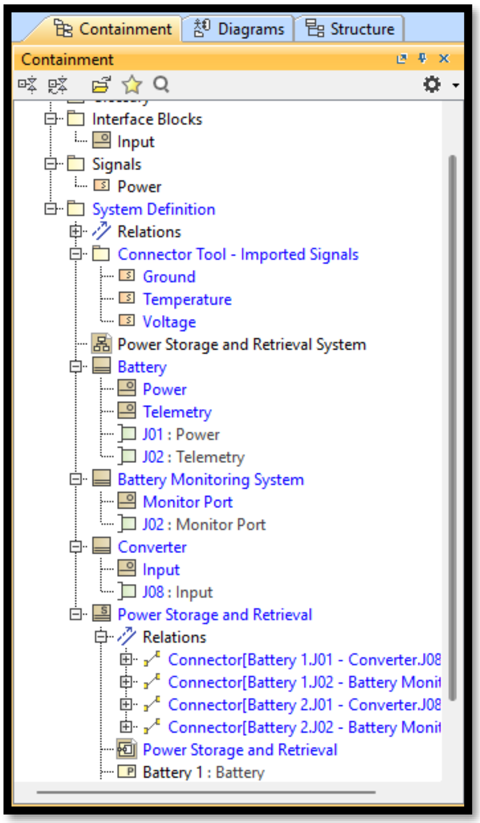


Figure 12 Example Results of Import – Interconnects

Post-Conditions: The model elements have been created and are contained in the desired elements. Where SysML does not dictate where the elements are to be contained, the plugin has created packages to hold those elements. See the below figure as an example.

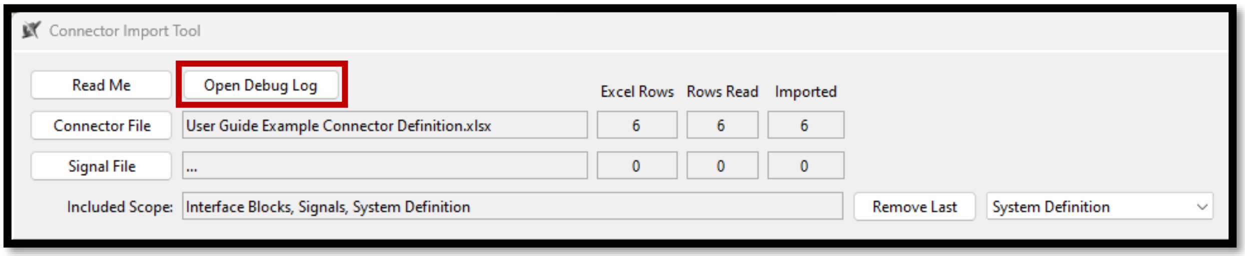


```
Interface Blocks
  Input
Signals
  Power
System Definition
  Relations
  Connector Tool - Imported Signals
    Ground
    Temperature
    Voltage
  Power Storage and Retrieval System
  Battery
    Power
    Telemetry
    J01 : Power
    J02 : Telemetry
  Battery Monitoring System
    Monitor Port
    J02 : Monitor Port
  Converter
    Input
    J08 : Input
  Power Storage and Retrieval
    Relations
      Connector[Battery 1.J01 - Converter.J08
      Connector[Battery 1.J02 - Battery Monit
      Connector[Battery 2.J01 - Converter.J08
      Connector[Battery 2.J02 - Battery Monit
    Power Storage and Retrieval
    Battery 1 : Battery
```

Figure 13 Example of Imported Element Containment - Interconnects

Troubleshooting and Debugging

To aid the user in determining why an individual connector was not imported, the plugin includes a log file accessible from the main dialog via the “Open Debug Log” button. Upon opening the debug log, the user is shown the complete log of actions printed to the log from the moment MagicDraw / Cameo has been opened with the plugin installed. Should the user want to retain a copy of the log the user should save the open debug log file to another directory.

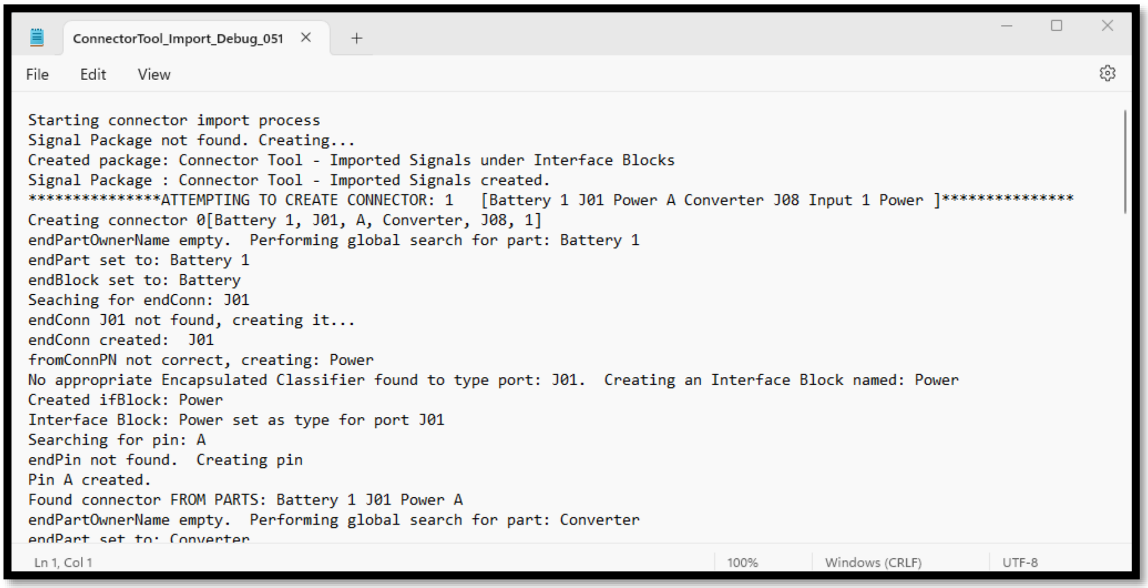


		Excel Rows	Rows Read	Imported
Connector File	User Guide Example Connector Definition.xlsx	6	6	6
Signal File	...	0	0	0

Included Scope: Interface Blocks, Signals, System Definition

System Definition

Figure 14 Debug Log Button



```
Starting connector import process
Signal Package not found. Creating...
Created package: Connector Tool - Imported Signals under Interface Blocks
Signal Package : Connector Tool - Imported Signals created.
***************ATTEMPTING TO CREATE CONNECTOR: 1   [Battery 1 J01 Power A Converter J08 Input 1 Power ]***************
Creating connector 0[Battery 1, J01, A, Converter, J08, 1]
endPartOwnerName empty.  Performing global search for part: Battery 1
endPart set to: Battery 1
endBlock set to: Battery
Seaching for endConn: J01
endConn J01 not found, creating it...
endConn created:  J01
fromConnPN not correct, creating: Power
No appropriate Encapsulated Classifier found to type port: J01.  Creating an Interface Block named: Power
Created ifBlock: Power
Interface Block: Power set as type for port J01
Searching for pin: A
endPin not found.  Creating pin
Pin A created.
Found connector FROM PARTS: Battery 1 J01 Power A
endPartOwnerName empty.  Performing global search for part: Converter
endPart set to: Converter
```

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

Figure 15 Example Debug Log