



Elektrobit

EB tresos® Studio for ACG8 user's guide

product release 8.7.1





Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 Erlangen, Germany
Phone: +49 9131 7701 0
Fax: +49 9131 7701 6333
Email: info.automotive@elektrobit.com

Technical support

<https://www.elektrobit.com/support>

Legal disclaimer

Confidential and proprietary information

ALL RIGHTS RESERVED. No part of this publication may be copied in any form, by photocopy, microfilm, retrieval system, or by any other means now known or hereafter invented without the prior written permission of Elektrobit Automotive GmbH.

All brand names, trademarks and registered trademarks are property of their rightful owners and are used only for description.

Copyright 2019, Elektrobit Automotive GmbH.



Table of Contents

Begin here	39
1. If you are upgrading from a previous version	39
2. If you are using EB tresos Studio for the first time	39
3. If you want to configure AUTOSAR modules or your own modules with EB tresos Studio	40
4. If you want to extend the EB tresos Studio functionality	40
5. If you need help/more information	41
1. About this documentation	42
1.1. Introduction	42
1.2. Required knowledge and system environment	42
1.2.1. Required system environment	43
1.3. Interpretation of version information	43
1.3.1. Product version number	43
1.3.2. Qualifiers for basic software	44
1.4. Typography and style conventions	45
1.5. Naming conventions	46
1.5.1. AUTOSAR XML schema	46
1.5.2. Configuration parameter names	47
1.6. EB tresos Studio naming conventions	47
1.6.1. EB tresos Studio command shell	47
2. Safe and correct use of EB tresos Studio	48
2.1. Intended usage of EB tresos Studio	48
2.2. Possible misuse of EB tresos Studio	48
2.3. Target group and required knowledge	48
2.4. Quality standards compliance	48
3. Background information	49
3.1. Overview	49
3.2. The AUTOSAR concept	49
3.3. What is EB tresos Studio?	50
3.3.1. The configuration and generation tool	50
3.3.2. Usage	51
3.3.3. Benefits	51
3.3.4. Features	52
3.4. File and directory structure	54
3.5. EB tresos Studio projects and workspaces	54
3.5.1. The Workspace	55
3.5.2. Projects	55
3.5.2.1. Configuration time of projects	56
3.5.3. File content types	57
3.5.3.1. Content types for XDM files	57



3.5.3.2. Content types for AUTOSAR module configuration files	58
3.5.3.3. Content types for AUTOSAR system description files	59
3.5.3.4. Content types for tresosDB files	60
3.5.3.5. Content types for DBC files	61
3.5.3.6. Content types for LDF files	61
3.5.3.7. Content types for Fibex files	62
4. Using EB tresos Studio for the first time	63
4.1. Starting the graphical user interface	63
4.2. Selecting the workspace location	63
4.3. The welcome page	65
4.4. Starting the Linux Command Line	66
5. The graphical user interface (GUI)	67
5.1. Using the Workbench	67
5.1.1. Customizing the workbench	69
5.1.1.1. Moving views to a different location	69
5.1.1.2. Closing views, and minimizing or maximizing view stacks	70
5.1.1.3. Opening a view	71
5.1.2. Using perspectives	72
5.1.2.1. Saving user-defined perspectives	73
5.1.2.2. Opening perspectives	74
5.1.2.3. Switching between perspectives	75
5.1.2.4. Closing perspectives	75
5.1.2.5. Reseting perspectives	76
5.1.2.6. Deleting perspectives	76
5.2. Menu bar and tool bar	77
5.2.1. File menu	78
5.2.2. Edit menu	80
5.2.3. Search menu	80
5.2.4. Project menu	81
5.2.5. Window menu	82
5.2.6. Help menu	83
5.3. Views	84
5.3.1. Project Explorer view	84
5.3.1.1. Project and ECU configuration states	86
5.3.1.2. Layering and grouping of module configurations	87
5.3.1.3. Changing the visible layers	88
5.3.1.4. Collapsing the Project Explorer tree	93
5.3.1.5. Synchronizing the Project Explorer with an editor	94
5.3.1.6. Indicating different project states with colors	95
5.3.2. The Editor view	96
5.3.3. Node Outline view	98
5.3.3.1. Changing the display settings of the Node Outline view	99



5.3.3.2. Copying the node name	100
5.3.3.3. Navigating to a node in the Editor view	100
5.3.3.4. Editing variant conditions	100
5.3.3.5. Deleting a variant node	100
5.3.3.6. Filtering the displayed items	101
5.3.3.7. Appearance of the displayed items	102
5.3.4. Element Outline views	103
5.3.4.1. Element Errors view	103
5.3.4.2. Element Information view	104
5.3.4.3. Element Description view	106
5.3.5. Error Log view	106
5.3.5.1. Changing the display settings of the Error Log view	108
5.3.5.2. Filtering Error Log messages by their severity level	109
5.3.5.3. Filtering Log messages by their Message Code	110
5.3.5.4. Restricting the amount of shown Error Log messages and the backup logfile size	110
5.3.5.5. Copying Error Log entries into text documents	111
5.3.6. Problems view	111
5.3.6.1. Changing the display settings of the Problems View	113
5.3.6.2. Export the messages of the Problems View	113
5.3.6.3. Filtering Problems view messages by their severity level	114
5.3.6.4. Limit the number of entries shown in the Problems View	114
5.3.7. Workflows view	115
5.3.7.1. Selecting a workflow	117
5.3.7.2. Reload workflows action	118
5.3.7.3. Performing steps of a workflow	119
5.3.8. Sidebar view	120
5.3.9. Properties view	121
5.3.9.1. Information tab	123
5.3.9.1.1. Copying Information tab values to the clipboard	124
5.3.9.2. Description tab	124
5.3.9.3. Comments tab	125
5.3.9.4. Problems tab	126
5.3.9.4.1. Copying error and warning messages to the clipboard	127
5.3.9.4.2. Adapting column width in Problems tab	127
5.3.9.5. PostBuildVariantConditions tab	128
5.3.9.5.1. Copying PostBuildVariantConditions tab values to the clipboard	128
5.3.9.6. CustomAttributes tab	128
5.3.9.6.1. Adding a custom attribute to a data node	129
5.3.9.6.2. Adding a value to a custom attribute	129
5.3.9.6.3. Removing a custom attribute from a data node	129
5.3.9.6.4. Editing a custom attribute value	129



5.3.9.6.5. Copying the content of the CustomAttributes tab to the clipboard	130
5.3.10. Results view	130
5.3.10.1. Filtering Error Log messages by their severity level	130
5.3.10.2. Export the messages of the Results view	131
5.4. Editors	131
5.4.1. The Generic Configuration Editor	131
5.4.1.1. Sections and parameters	131
5.4.1.2. References	136
5.4.1.3. Choices	138
5.4.1.4. Tables	139
5.4.1.4.1. Duplicating elements	142
5.4.1.5. Optional elements	143
5.5. Keyboard shortcuts	143
5.5.1. Shortcuts for the Workbench	144
5.5.2. Shortcuts for views	144
5.5.3. Shortcuts for editors and projects	144
5.5.4. Shortcuts for tables	144
6. Use cases	145
6.1. Overview	145
6.2. Changing the EB tresos Studio setup	146
6.2.1. Viewing setup information	146
6.2.1.1. Opening the About EB tresos dialog	146
6.2.1.2. Finding program startup parameters and system property settings	147
6.2.1.3. Opening the EB tresos Details dialog	148
6.2.1.4. Finding the location of the installation and the workspace	148
6.2.1.5. Finding all installed modules	148
6.2.1.6. Finding all installed plug-ins	149
6.2.2. Setting preferences	150
6.2.2.1. Setting general preferences	152
6.2.2.2. Setting AUTOSAR preferences	154
6.2.2.3. Setting Assistance dialog preferences	155
6.2.2.4. Setting Code generator preferences	156
6.2.2.5. Setting configuration preferences	157
6.2.2.6. Setting Labels preferences	159
6.2.2.7. Setting Load/reload preferences	160
6.2.2.8. Setting Developer features preferences	161
6.2.2.9. Setting Error Log preferences	162
6.2.2.10. Setting License preferences	163
6.2.2.11. Setting preferences for parameter comments	164
6.2.2.12. Setting colors and fonts	165
6.2.3. Working with multiple workspaces to store different projects and settings	168
6.2.3.1. Switching the workspace	168



6.2.3.2. Finding out which workspace directory is used	169
6.2.4. Placing an EB tresos Studio installation under version control	169
6.2.5. Adding new features and extensions to EB tresos Studio via external folders	169
6.3. Working with projects	170
6.3.1. Creating a new project	170
6.3.1.1. Step 1: Starting the Project Wizard	171
6.3.1.2. Step 2: Creating a new project	173
6.3.1.3. Step 3: Configuring project properties	174
6.3.1.4. Step 4: Defining module configurations	176
6.3.1.4.1. Adding module configurations to your project	177
6.3.1.4.2. Removing module configurations from your project	178
6.3.1.4.3. Enabling and disabling module configurations	180
6.3.1.4.4. Enabling and disabling code generation for module configurations	181
6.3.1.4.5. Naming Module Configurations	183
6.3.1.4.6. Choosing Pre-Configurations	184
6.3.1.4.6.1. Setting the Default Pre-Configuration	184
6.3.1.4.6.2. Changing the preconfiguration of one specific module configura- tion	185
6.3.1.4.7. Applying recommended configurations	187
6.3.1.4.7.1. Setting the Default Recommended Configuration	187
6.3.1.4.7.2. Changing the recommended configuration of one specific mod- ule configuration	188
6.3.1.4.8. Editing the configuration file settings	190
6.3.1.4.8.1. Using existing configuration files	191
6.3.1.4.9. Editing the code generation output path settings	191
6.3.1.4.10. Upgrading module configurations	192
6.3.1.5. Step 5: Importing data into the project (optional)	193
6.3.1.6. Step 6: Finishing set-up of the new project	194
6.3.2. Saving a project	194
6.3.3. Renaming a project	195
6.3.4. Copying a project	195
6.3.5. Deleting a project	196
6.3.6. Closing and opening projects	196
6.3.6.1. Closing projects	196
6.3.6.2. Opening a project	199
6.3.6.3. Loading a project	199
6.3.6.4. Reloading a project	199
6.3.6.4.1. Reloading projects automatically	199
6.3.6.4.2. Reloading projects manually	200
6.3.7. Viewing and changing project properties	200
6.3.7.1. Listing the supported models of the project	202
6.3.7.2. Editing the Code Generator settings	203



6.3.7.3. Editing the ECU Configuration settings	204
6.3.7.4. Viewing information about the project's modules	205
6.3.7.4.1. Adding module configurations to your project	206
6.3.7.4.2. Removing module configurations from your project	207
6.3.7.4.3. Enabling and disabling module configurations	209
6.3.7.4.4. Enabling and disabling code generation for module configurations	210
6.3.7.4.5. Naming Module Configurations	212
6.3.7.4.6. Choosing Pre-Configurations	213
6.3.7.4.6.1. Setting the Default Pre-Configuration	213
6.3.7.4.6.2. Changing the preconfiguration of one specific module configuration	214
6.3.7.4.7. Applying recommended configurations	216
6.3.7.4.7.1. Setting the Default Recommended Configuration	216
6.3.7.4.7.2. Changing the recommended configuration of one specific module configuration	217
6.3.7.4.8. Editing the configuration file settings	219
6.3.7.4.8.1. Using existing configuration files	220
6.3.7.4.9. Editing the code generation output path settings	220
6.3.7.4.10. Upgrading module configurations	221
6.3.7.5. Editing the System Configuration settings	222
6.4. Using team collaboration support	224
6.4.1. Overview	224
6.4.2. Working with CVS repositories	224
6.4.2.1. Registering your CVS repository location in EB tresos Studio	225
6.4.2.2. Sharing an EB tresos Studio project with other team members	227
6.4.2.3. Checking out a shared project from your CVS repository into your EB tresos Studio workspace	231
6.4.2.4. Updating, branching, and committing EB tresos Studio projects into your CVS repository	233
6.4.3. Working with SVN repositories	234
6.4.3.1. Registering your SVN repository location in EB tresos Studio	235
6.4.3.2. Sharing an EB tresos Studio project with other team members	236
6.4.3.3. Checking out a shared project from your SVN repository into your EB tresos Studio workspace	240
6.4.3.4. Updating, branching, and committing EB tresos Studio projects into your SVN repository	242
6.4.4. Synchronizing your project with your repository	245
6.4.4.1. Overview	245
6.4.4.2. Background information	245
6.4.4.2.1. The Team Synchronizing perspective	245
6.4.4.2.2. The Synchronize view	246
6.4.4.3. Opening the Team Synchronizing perspective	248



6.4.4.4. Viewing differences between your working copy and the base file in the repository	250
6.4.4.5. Synchronizing your working copy with changes in the repository	251
6.5. Diffing files and file versions	253
6.5.1. Overview	253
6.5.2. Diffing XDM files	253
6.5.2.1. Overview	254
6.5.2.2. Background information	254
6.5.2.3. Opening the compare editor	254
6.5.2.3.1. The Compare editor for XDM files	257
6.5.2.3.2. Tree table icons and overlay icons	259
6.5.2.4. Understanding the diffing results	260
6.5.2.5. Navigating from difference to difference	261
6.5.2.5.1. Filtering	261
6.5.2.5.2. Expanding and collapsing the tree	261
6.5.2.5.3. Keyboard shortcuts	262
6.5.2.6. Merging differences	262
6.5.2.6.1. Applying differences	263
6.5.2.6.1.1. Applying single differences	263
6.5.2.6.1.2. Applying subtrees	264
6.5.2.6.1.3. Applying all remaining differences	264
6.5.2.6.2. Rejecting changes	264
6.5.2.6.2.1. Rejecting single differences	265
6.5.2.6.2.2. Rejecting subtrees	265
6.5.2.6.3. Undoing/Redoing operations	266
6.5.2.7. Limitations	266
6.5.2.7.1. Ignored differences	266
6.5.2.7.2. Schema support	266
6.6. Editing module configurations	266
6.6.1. Adding module configurations to your project	269
6.6.2. Removing module configurations from your project	270
6.6.3. Enabling and disabling module configurations	272
6.6.4. Enabling and disabling code generation for module configurations	276
6.6.5. Naming Module Configurations	278
6.6.6. Choosing Pre-Configurations	279
6.6.6.1. Setting the Default Pre-Configuration	279
6.6.6.2. Changing the preconfiguration of one specific module configuration	280
6.6.7. Applying recommended configurations	282
6.6.7.1. Setting the Default Recommended Configuration	282
6.6.7.2. Changing the recommended configuration of one specific module configuration	283
6.6.8. Editing the configuration file settings	285



6.6.8.1. Using existing configuration files	286
6.6.9. Editing the code generation output path settings	287
6.6.10. Upgrading module configurations	287
6.7. Upgrading projects and configurations	288
6.7.1. Converting a project to a new AUTOSAR version	289
6.7.2. Upgrading module configurations to new module versions	290
6.8. Editing parameters of a module configuration	292
6.8.1. Overview	292
6.8.2. Background information	293
6.8.3. Opening the ECU Configuration Editor	294
6.8.4. Setting the configuration variant of the module	295
6.8.5. Editing variant conditions	296
6.8.6. Adapting references when you rename containers	298
6.8.7. Adding parameter comments	299
6.8.7.1. Adding and editing a parameter comment	299
6.8.7.2. Managing languages of parameter comments	301
6.8.7.3. Removing parameter comments	302
6.8.7.4. Generating a parameter comment report	303
6.8.8. Generating a configuration report which includes comments	303
6.8.8.1. Installing the application example <i>DemoReportGenerator</i>	303
6.8.8.2. Generating a report	304
6.8.8.3. Understanding the configuration report	304
6.8.8.4. Customizing the <i>DemoReportGenerator</i>	305
6.8.9. Editing several table rows simultaneously	305
6.8.10. Autoconfiguring routine jobs	308
6.8.10.1. Configuring unattended wizards	308
6.8.10.2. Managing multiple instances of unattended wizards	310
6.8.10.3. Running unattended wizards	312
6.8.10.3.1. Running a single unattended wizard	312
6.8.10.3.2. Running multiple unattended wizards	314
6.8.10.3.3. Result feedback	316
6.8.10.4. Configuring the Calculate Derivable Values wizard	317
6.8.10.5. Configuring the Calculate Handle IDs wizard	318
6.8.10.6. Configuring the Execute multiple tasks wizard	319
6.8.10.7. Configuring the Switch PostBuildVariant wizard	320
6.8.10.8. Configuring the Generate Code wizard	321
6.8.11. Searching for configuration parameters	322
6.8.11.1. Setting search options	324
6.8.11.1.1. Defining general search options	324
6.8.11.1.2. Defining which kind of text is searched	325
6.8.11.1.3. Limiting the kind of parameters that are searched	325
6.8.11.1.4. Defining the scope of the search	327



6.8.11.2. Displaying search results	327
6.9. Generating code for projects	329
6.9.1. Verifying a project	329
6.9.2. Generating a project	331
6.9.3. Building a project	334
6.10. Importing and exporting	336
6.10.1. Exchanging a complete EB tresos Studio project with other installations	336
6.10.1.1. Importing a project	337
6.10.1.2. Exporting a project	342
6.10.1.2.1. Exporting a project as copy	343
6.10.1.2.2. Exporting a project as archive	344
6.10.1.2.3. Importing a project	345
6.10.2. Importing and exporting configuration data	345
6.10.2.1. Step-by-step guide	346
6.10.2.1.1. Step 1: Using the Create, manage and run im- and exporters window.	346
6.10.2.1.2. Step 2: Creating an export or import configuration	349
6.10.2.1.3. Step 3: Saving changes to an import or export configuration	350
6.10.2.1.4. Step 4: Importing or exporting project configuration data	350
6.10.2.2. Importing and exporting configuration data from and to the AUTOSAR format	351
6.10.2.2.1. Path mapping in import mode	355
6.10.2.2.2. Strategies for importing and exporting configuration data	357
6.10.2.2.2.1. Replace	357
6.10.2.2.2.2. Replace specific Config Classes	357
6.10.2.2.2.3. Merge	358
6.10.2.2.2.4. Merge specific Config Classes	358
6.10.2.2.2.5. Export	359
6.10.2.2.2.6. Export specific Config Classes	359
6.10.2.2.2.7. Import/Export	359
6.10.2.2.3. Using path variables	359
6.10.2.2.4. Importing post build variants	361
6.10.2.2.4.1. Importing post build selectable variants	361
6.10.2.2.4.2. Importing post build loadable variants	362
6.10.2.3. Importing AUTOSAR system descriptions	362
6.10.2.3.1. Overview	362
6.10.2.3.2. Background information	362
6.10.2.3.3. Importing system information	363
6.10.2.3.4. References	369
6.10.2.3.5. Variants tab	369
6.10.2.4. Exporting AUTOSAR system description	370
6.10.2.4.1. Overview	370
6.10.2.4.2. Background information	370
6.10.2.4.3. Exporting system information	371



6.10.2.5. Importing from the OIL format	374
6.10.2.5.1. Overview	374
6.10.2.5.2. OIL Importer and file format	374
6.10.2.5.3. Importing OIL configurations	374
6.10.2.5.4. Importing OIL files via the command line	377
6.10.2.5.4.1. OIL Importer parameters for the command line	377
6.10.2.5.4.2. OIL import example	377
6.10.2.6. Importing CAN configurations from the DBC format	377
6.10.2.6.1. Overview	378
6.10.2.6.2. Background information	378
6.10.2.6.2.1. Limitations	379
6.10.2.6.3. Importing DBC configurations	379
6.10.2.6.4. References	383
6.10.2.6.4.1. Import customizations	383
6.10.2.6.4.2. The DBC parser	384
6.10.2.6.4.2.1. DBC grammar	384
6.10.2.6.4.3. I-Pdu Multiplexer Configuration	387
6.10.2.7. Importing from the LDF format	391
6.10.2.7.1. Overview	392
6.10.2.7.2. LDF Importer and file format	392
6.10.2.7.3. Importing LDF configurations	392
6.10.2.8. Importing from the Fibex format	396
6.10.2.8.1. Overview	396
6.10.2.8.2. Background information	396
6.10.2.8.2.1. Limitations	396
6.10.2.8.2.2. EndToEndProtection related information	397
6.10.2.8.2.2.1. END-TO-END-PROTECTION-SET	397
6.10.2.8.2.2.2. END-TO-END-PROTECTION	397
6.10.2.8.2.2.2.1. END-TO-END-PROTECTION-I-SIGNAL-I-PDU ..	398
6.10.2.8.2.2.3. I-SIGNAL-GROUP	398
6.10.2.8.2.2.4. SYSTEM-SIGNAL-GROUP	398
6.10.2.8.2.2.5. I-SIGNAL-TRIGGERING	399
6.10.2.8.2.2.6. I-SIGNAL-PORTs	399
6.10.2.8.3. Importing Fibex configurations	399
6.10.2.9. Importing configurations from the TDB format	403
6.10.2.9.1. Overview	403
6.10.2.9.2. Importing TDB configurations	403
6.10.2.10. Reusing importers and exporters across projects	406
6.10.2.10.1. Overview	406
6.10.2.10.2. Reusing workflow files across projects	406
6.10.2.10.3. Backward compatibility of importers	406
6.10.2.11. Importing ECU configurations from AUTOSAR system descriptions	407



6.10.2.11.1. Overview	407
6.10.2.11.2. Background information	407
6.11. Creating an ECU Extract	411
6.11.1. The unattended wizard Create an ECU Extract(EcuExtractCreator)	412
6.11.2. Fixed AUTOSAR short-name paths	413
6.11.3. SwComponentPrototype short-names within the ECU flat map	413
6.12. Working with the command line	416
6.12.1. General	416
6.12.1.1. Starting the commandline	416
6.12.1.2. The commandline syntax	416
6.12.1.3. System properties common to all commands	417
6.12.1.4. Specifying the workspace on the commandline	418
6.12.2. Commands for project-based code generation and configuration	419
6.12.2.1. Importing projects	419
6.12.2.1.1. Syntax	419
6.12.2.1.2. System properties	420
6.12.2.1.3. Parameters	420
6.12.2.1.4. Example	420
6.12.2.2. Deleting projects	420
6.12.2.2.1. Syntax	420
6.12.2.2.2. System properties	421
6.12.2.2.3. Parameters	421
6.12.2.2.4. Example	421
6.12.2.3. Renaming projects	421
6.12.2.3.1. Syntax	421
6.12.2.3.2. System properties	422
6.12.2.3.3. Parameters	422
6.12.2.3.4. Example	422
6.12.2.4. Executing importers configured for a project	422
6.12.2.4.1. Syntax	422
6.12.2.4.2. System properties	423
6.12.2.4.3. Parameters	423
6.12.2.4.4. Example	423
6.12.2.5. Listing importers configured for a project	423
6.12.2.5.1. Syntax	424
6.12.2.5.2. System properties	424
6.12.2.5.3. System properties	424
6.12.2.5.4. Parameters	424
6.12.2.5.5. Example	424
6.12.2.6. Verifying a project	424
6.12.2.6.1. Syntax	425
6.12.2.6.2. System properties	425



6.12.2.6.3. Parameters	425
6.12.2.6.4. Example	425
6.12.2.7. Generating code for a project	425
6.12.2.7.1. Syntax	425
6.12.2.7.2. System properties	426
6.12.2.7.3. Options	426
6.12.2.7.4. Parameters	427
6.12.2.7.5. Example	427
6.12.2.8. Listing custom-built generation modes	427
6.12.2.8.1. Syntax	427
6.12.2.8.2. System properties	428
6.12.2.8.3. Parameters	428
6.12.2.8.4. Example	428
6.12.2.9. Executing custom-built generation modes for projects	428
6.12.2.9.1. Syntax	429
6.12.2.9.2. System properties	429
6.12.2.9.3. Parameters	429
6.12.2.9.4. Example	430
6.12.2.10. Executing unattended wizards on projects	430
6.12.2.10.1. Syntax	430
6.12.2.10.2. System properties	431
6.12.2.10.3. Parameters	431
6.12.2.10.4. Example	431
6.12.2.10.5. Running the Calculate Default Values wizard on the command line.....	431
6.12.2.10.6. Running the Calculate Handle-Ids wizard on the command line	431
6.12.2.11. Listing all unattended wizards for a given project	432
6.12.2.11.1. Syntax	432
6.12.2.11.2. System properties	432
6.12.2.11.3. Parameters	432
6.12.2.11.4. Example	432
6.12.2.12. Upgrading module configurations for a project	433
6.12.2.12.1. Syntax	433
6.12.2.12.2. System properties	433
6.12.2.12.3. Parameters	434
6.12.2.12.4. Example	434
6.12.3. Commands for legacy code generation	434
6.12.3.1. Listing all modules available in legacy mode	435
6.12.3.1.1. Syntax	435
6.12.3.1.2. System properties	435
6.12.3.1.3. Example	435
6.12.3.2. Generating code	435
6.12.3.2.1. Syntax	435



6.12.3.2.2. System properties	436
6.12.3.2.3. Options	439
6.12.3.2.4. Parameters	441
6.12.3.2.5. Example	442
6.12.3.3. Verifying configurations	442
6.12.3.3.1. Syntax	442
6.12.3.3.2. System properties	442
6.12.3.3.3. Options	444
6.12.3.3.4. Parameters	446
6.12.3.3.5. Example	446
6.12.3.4. Listing all generation modes	446
6.12.3.4.1. Syntax	447
6.12.3.4.2. System properties	447
6.12.3.4.3. Example	447
6.12.3.5. Running generation modes	447
6.12.3.5.1. Syntax	447
6.12.3.5.2. System properties	448
6.12.3.5.3. Options	450
6.12.3.5.4. Parameters	452
6.12.3.5.5. Example	452
6.12.4. Commands for module development	452
6.12.4.1. Listing a data model	453
6.12.4.1.1. Syntax	453
6.12.4.1.2. System properties	453
6.12.4.1.3. Options	453
6.12.4.1.4. Parameters	454
6.12.4.1.5. Example	454
6.12.4.2. Creating vendor-specific module definitions (VSMD)	454
6.12.4.2.1. Syntax	455
6.12.4.2.2. System properties	455
6.12.4.2.3. Options	458
6.12.4.2.4. Parameters	459
6.12.4.2.5. Example	460
6.12.4.3. Checking vendor-specific against standard module definitions	460
6.12.4.3.1. Syntax	460
6.12.4.3.2. System properties	460
6.12.4.3.3. Parameters	461
6.12.4.3.4. Example	462
6.12.4.4. Listing all rules used for VSMD checking	463
6.12.4.4.1. Syntax	463
6.12.4.4.2. System properties	463
6.12.4.4.3. Example	463



6.12.4.5. Encrypting and signing modules	463
6.12.5. Commands for converting and merging files	463
6.12.5.1. Merging files and converting between different file formats	464
6.12.5.1.1. Syntax	464
6.12.5.1.2. System properties	464
6.12.5.1.3. Options	475
6.12.5.1.4. Parameters	475
6.12.5.1.5. Example	476
6.12.5.2. Using XDM files in different EB tresos Studio versions	477
6.12.5.2.1. Example	477
6.12.5.3. Upgrading the AUTOSAR compliance of XDM files	477
6.12.5.3.1. Syntax	477
6.12.5.3.2. System properties	478
6.12.5.3.3. Parameters	478
6.12.6. Executing multiple commands (batch mode)	479
6.12.6.1. Batch file syntax	479
6.12.6.2. Handling of system properties	480
6.13. Working with variants and Post-build loadable	480
6.13.1. Variant handling concept by AUTOSAR	481
6.13.1.1. PreBuild variation points	482
6.13.1.2. PostBuild variation points	483
6.13.1.2.1. PostBuild loadable concept	483
6.13.1.2.2. PostBuild selectable concept (PostBuildVariants)	484
6.13.1.2.3. Combined PostBuild loadable and selectable concept	484
6.13.1.3. Restrictions on variant handling in EB tresos Studio	485
6.13.1.4. Fast access tool bar	486
6.13.2. Prerequisites	486
6.13.2.1. Module supporting variant selection	486
6.13.2.2. Variant definitions	486
6.13.3. Use cases	488
6.13.3.1. Configuring variants for a project	488
6.13.3.2. Editing PostBuildVariants	489
6.13.3.3. Editing textual descriptions for PostBuildVariantCriterions	489
6.13.3.4. Editing variant conditions	490
6.13.3.5. Looking up variant conditions of a parameter	490
6.13.3.6. Resolving variants	490
6.13.3.6.1. Resolving variants example: commandline	490
6.13.3.6.2. Resolving variants example: Im- and Export wizard	491
6.13.3.7. Handling variants in legacy generate	491
6.13.4. Recommended workflow	492
6.13.4.1. Setting up a variant project	492
6.13.4.2. Working with post-build selectable variants	493



6.13.4.2.1. Prerequisites	493
6.13.4.2.2. Workflow	493
6.13.4.3. Working with post-build loadable variants	494
6.14. Post-build loadable configuration using multiple configuration containers	495
6.14.1. General post-build loadable concept	495
6.14.2. Using Multiple Configuration Containers for post-build loadables	496
6.14.3. Defining valid sets of <i>Multiple Configuration Containers</i>	496
6.14.3.1. Generating code	497
6.14.3.2. Importing a system description	497
6.14.3.3. Guided Configuration wizards	497
6.14.4. Value Constraints	497
6.14.4.1. Readonly	497
6.14.4.2. Dependencies between parameters	498
6.15. Creating a customer support package	498
6.15.1. Overview	498
6.15.2. Using the Create support package dialog	499
6.15.2.1. Opening the Create support package dialog	499
6.15.2.2. Creating the support package	501
7. Tool extensions	502
7.1. Overview	502
7.2. Using the Connection Editor	503
7.2.1. Overview	503
7.2.2. Background information	503
7.2.2.1. The hierarchical model	503
7.2.2.2. Delegatable service ports	504
7.2.2.3. Changes affect all prototypes of a composition	504
7.2.2.4. Connectors are displayed as children of the ports they connect	504
7.2.3. Prerequisites	505
7.2.4. Using the EB tresos Connection Editor	507
7.2.4.1. Getting started with the Connection Editor	507
7.2.4.1.1. Opening the Connection Editor	507
7.2.4.1.2. The sections of the Connection Editor	508
7.2.4.2. Connecting ports manually	509
7.2.4.2.1. Connecting incompatible ports	512
7.2.4.3. Creating a delegation connector	513
7.2.4.4. Adding a component prototype	514
7.2.4.5. Removing elements	516
7.2.4.6. Connecting ports automatically	518
7.2.4.7. Proceeding after editing connections	521
7.3. Using the System Model Viewer	522
7.3.1. Overview	522
7.3.2. Background information	523



7.3.2.1. Entity tree table icons	523
7.3.3. Opening the System Model Viewer	524
7.3.4. Displaying entities in the Entity tree table	524
7.3.5. Filtering the displayed entities	524
7.3.5.1. Filtering entities by type	524
7.3.5.2. Filtering entities by name	525
7.4. Using the Signal Mapping Editor	525
7.4.1. Overview	525
7.4.2. Background information	526
7.4.2.1. The hierarchical model	526
7.4.2.2. Displayed data mappings	526
7.4.2.3. Supported types of data mapping	527
7.4.3. Prerequisites	528
7.4.4. Using the EB tresos Signal Mapping Editor	529
7.4.4.1. Getting started with the Signal Mapping Editor	529
7.4.4.1.1. Opening the Signal Mapping Editor	529
7.4.4.1.2. The sections of the Signal Mapping Editor	530
7.4.4.2. Creating sender-receiver to signal mapping	532
7.4.4.3. Creating sender-receiver to signal group mapping	534
7.4.4.4. Creating client-server to signal mapping	537
7.4.4.5. Editing sender receiver to signal group mapping	540
7.4.4.6. Removing data mappings	542
7.4.4.7. Creating data mappings automatically	544
7.4.4.8. Proceeding after data mappings were created	547
7.5. Using the Auto-connect SWC unattended wizard	548
7.5.1. Configuring the Auto-connect SWC unattended wizard	549
7.5.1.1. Match table	550
7.5.1.2. Rule file	550
7.5.2. Validation mechanism	550
7.5.3. Executing the Auto-connect SWC unattended wizard	551
7.5.3.1. Executing the Auto-connect SWC unattended wizard on the command line	551
7.5.4. Logging mechanism	551
7.5.4.1. ConnectionEditor.arxml file	552
7.6. Using the Auto-map system signals unattended wizard	553
7.6.1. Configuring the Auto-map system signal unattended wizard	554
7.6.1.1. Match table	554
7.6.1.2. Rule file	555
7.6.2. Validation mechanism	555
7.6.3. Executing the Auto-map system signal unattended wizard	556
7.6.3.1. Executing the Auto-map system signal unattended wizard on the command line	556
7.6.4. Logging mechanism	556



7.6.4.1. ConnectionEditor.arxml file	557
7.7. Using the Edit PostBuildVariants wizards	557
7.7.1. The Predefined Variants page	559
7.7.2. The Variant Criterions page	560
7.8. Using the Variant Handling wizard	561
7.9. Using the Export/Import Split Configuration unattended wizards	562
7.9.1. Overview	562
7.9.2. Background information	562
7.9.2.1. Splittags	563
7.9.2.2. Export split arxml files	564
7.9.2.3. Import split arxml files	565
7.9.3. Configuring the Export Split Configuration wizard	565
7.9.3.1. Opening the Export Split Configuration wizard	565
7.9.3.2. Output folder path	566
7.9.3.3. AUTOSAR version	567
7.9.3.4. Store parameter state information	568
7.9.3.5. Export data filter	569
7.9.3.6. Expected files and content	570
7.9.4. Configuring the Import Split Configuration wizard	571
7.9.4.1. Opening the Import Split Configuration wizard	571
7.9.4.2. Input folder path	572
7.9.4.3. Files glob patterns	573
7.9.4.4. Select file action	574
7.9.4.5. Import strategy	575
7.9.4.6. Restore parameter state information	576
7.10. Using the Transform XML files using XSLT wizard	577
7.10.1. Overview	577
7.10.2. Configuring the Transform XML files using XSLT wizard	578
7.10.3. Running the Transform XML files using XSLT wizard	579
7.11. Using the Service Needs Calculator	580
7.11.1. Overview	580
7.11.2. Background information	580
7.11.2.1. The concept of service needs	581
7.11.2.2. Functional distribution	583
7.11.2.3. Service needs calculation life cycle	584
7.11.2.4. Limitations	586
7.11.2.5. Relation to the importer info value	586
7.11.2.6. Deleting elements	587
7.11.2.6.1. Deleting elements based on the importer info value	587
7.11.2.6.2. Removing Com elements	588
7.11.3. Configuring the Service Needs Calculator	589
7.11.4. Running the Service Needs Calculator	590



7.11.4.1. Results after running the Service Needs Calculator	591
8. References	594
8.1. Upgrading system models	594
8.1.1. Upgrading a system model from AUTOSAR 3.1.x to AUTOSAR 4.0.2	595
8.1.1.1. Topology	595
8.1.1.1.1. SYSTEM	595
8.1.1.1.1.1. Changed or moved parameters	595
8.1.1.1.2. ECU-INSTANCE	596
8.1.1.1.2.1. Reason for not upgraded source model parameter	596
8.1.1.1.3. CAN-CLUSTER	596
8.1.1.1.3.1. Changed or moved parameters	597
8.1.1.1.4. CAN-COMMUNICATION-CONTROLLER	597
8.1.1.1.5. CAN-COMMUNICATION-CONNECTOR	598
8.1.1.1.5.1. Changed or moved parameters	598
8.1.1.1.6. CAN-PHYSICAL-CHANNEL	598
8.1.1.1.6.1. Changed or moved parameters	599
8.1.1.1.7. FLEXRAY-CLUSTER	599
8.1.1.1.7.1. Changed or moved parameters	601
8.1.1.1.8. FLEXRAY-COMMUNICATION-CONTROLLER	601
8.1.1.1.9. FLEXRAY-COMMUNICATION-CONNECTOR	602
8.1.1.1.9.1. Changed or moved parameters	603
8.1.1.1.10. FLEXRAY-PHYSICAL-CHANNEL	603
8.1.1.1.10.1. Changed or moved parameters	603
8.1.1.1.11. LIN-CLUSTER	604
8.1.1.1.11.1. Changed or moved parameters	604
8.1.1.1.12. LIN-MASTER	604
8.1.1.1.13. LIN-SLAVE	605
8.1.1.1.13.1. Reasons for not configured target model parameters	605
8.1.1.1.14. LIN-ERROR-RESPONSE	605
8.1.1.1.15. LIN-COMMUNICATION-CONNECTOR	606
8.1.1.1.15.1. Reason for not configured target model parameters	606
8.1.1.1.15.2. Changed or moved parameters	606
8.1.1.1.16. LIN-PHYSICAL-CHANNEL	607
8.1.1.1.16.1. Changed or moved parameters	607
8.1.1.2. Communication	607
8.1.1.2.1. CAN-FRAME-TRIGGERING	607
8.1.1.2.2. FLEXRAY-FRAME-TRIGGERING	608
8.1.1.2.3. FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING	608
8.1.1.2.4. LIN-FRAME-TRIGGERING	609
8.1.1.2.4.1. Changed or moved parameters	609
8.1.1.2.5. CAN-FRAME	609
8.1.1.2.5.1. Changed or moved parameters	610



8.1.1.2.6. FLEXRAY-FRAME	610
8.1.1.2.6.1. Changed or moved parameters	610
8.1.1.2.7. LIN-UNCONDITIONAL-FRAME	611
8.1.1.2.7.1. Changed or moved parameters	611
8.1.1.2.8. LIN-SPORADIC-FRAME	611
8.1.1.2.8.1. Changed or moved parameters	612
8.1.1.2.9. LIN-EVENT-TRIGGERED-FRAME	612
8.1.1.2.9.1. Changed or moved parameters	613
8.1.1.2.10. LIN-SCHEDULE-TABLE	613
8.1.1.2.10.1. Changed or moved parameters	613
8.1.1.2.11. ASSIGN-FRAME-ID	613
8.1.1.2.12. UNASSIGN-FRAME-ID	614
8.1.1.2.13. ASSIGN-NAD	614
8.1.1.2.14. FREE-FORMAT	614
8.1.1.2.15. ASSIGN-FRAME-ID-RANGE	615
8.1.1.2.16. CONDITIONAL-CHANGE-NAD	615
8.1.1.2.17. DATA-DUMP-ENTRY	616
8.1.1.2.18. SAVE-CONFIGURATION-ENTRY	616
8.1.1.2.19. PDU-TRIGGERING	617
8.1.1.2.20. I-SIGNAL-TRIGGERING	617
8.1.1.2.20.1. Changed or moved parameters	618
8.1.1.2.21. DCM-I-PDU	618
8.1.1.2.21.1. Changed or moved parameters	618
8.1.1.2.22. I-SIGNAL-I-PDU	618
8.1.1.2.22.1. Changed or moved parameters	619
8.1.1.2.23. MULTIPLEXED-I-PDU	619
8.1.1.2.23.1. DYNAMIC-PART and STATIC-PART: differences between source model and the EB tresos Studio system model	619
8.1.1.2.23.2. Changed or moved parameters	620
8.1.1.2.24. DYNAMIC-PART	620
8.1.1.2.25. DYNAMIC-PART-ALTERNATIVE	620
8.1.1.2.26. STATIC-PART	621
8.1.1.2.27. SEGMENT-POSITION	621
8.1.1.2.28. N-PDU	622
8.1.1.2.28.1. Changed or moved parameters	622
8.1.1.2.29. NM-PDU	622
8.1.1.2.29.1. Changed or moved parameters	623
8.1.1.2.30. PDU-TO-FRAME-MAPPING	623
8.1.1.2.31. I-PDU-TIMING	623
8.1.1.2.31.1. Changed or moved parameters	623
8.1.1.2.32. I-SIGNAL-I-PDU-GROUP	624
8.1.1.2.33. I-SIGNAL-TO-I-PDU-MAPPING	624



8.1.1.2.33.1. Changed or moved parameters	624
8.1.1.2.34. I-SIGNAL	625
8.1.1.2.34.1. Changed or moved parameters	625
8.1.1.2.35. I-SIGNAL-GROUP	626
8.1.1.2.35.1. Changed or moved parameters	626
8.1.1.2.36. SYSTEM-SIGNAL	627
8.1.1.2.36.1. Changed or moved parameters	627
8.1.1.2.37. I-SIGNAL-PORT	627
8.1.1.2.38. FRAME-PORT	628
8.1.1.2.39. I-PDU-PORT	628
8.1.1.2.40. TRANSMISSION-MODE-DECLARATION	628
8.1.1.2.40.1. Changed or moved parameters	629
8.1.1.2.41. TRANSMISSION-MODE-CONDITION	629
8.1.1.2.42. TRANSMISSION-MODE-FALSE-TIMING	629
8.1.1.2.43. EVENT-CONTROLLED-TIMING	629
8.1.1.2.44. CYCLIC-TIMING	630
8.1.1.2.45. TIME-RANGE-TYPE	630
8.1.1.2.46. ABSOLUTE-TOLERANCE	630
8.1.1.2.47. RELATIVE-TOLERANCE	630
8.1.1.3. Gateway	631
8.1.1.3.1. FRAME-MAPPING	631
8.1.1.3.2. I-PDU-MAPPING	631
8.1.1.3.3. TARGET-I-PDU-REF	631
8.1.1.3.4. PDU-MAPPING-DEFAULT-VALUE	631
8.1.1.3.5. DEFAULT-VALUE-ELEMENT	632
8.1.1.3.6. I-SIGNAL-MAPPING	632
8.1.1.4. CAN Transport Layer	632
8.1.1.4.1. CAN-TP-CONFIG	632
8.1.1.4.1.1. New parameters in the target model	632
8.1.1.4.1.2. Changed or moved parameters	633
8.1.1.4.2. CAN-TP-CONNECTION	633
8.1.1.4.2.1. New parameters in the target model	634
8.1.1.4.2.2. Changed or moved parameters	635
8.1.1.5. FlexRay Transport Layer	635
8.1.1.5.1. FLEXRAY-TP-CONFIG	635
8.1.1.5.1.1. New parameters in the target model	635
8.1.1.5.2. FLEXRAY-TP-CONNECTION	636
8.1.1.5.2.1. New parameters in the target model	636
8.1.1.5.3. FLEXRAY-TP-CONNECTION-CONTROL	637
8.1.1.5.3.1. New parameters in the target model	638
8.1.1.6. LIN Transport Layer	638
8.1.1.6.1. LIN-TP-CONFIG	638



8.1.1.6.1.1. New parameters in the target model	638
8.1.1.6.2. LIN-TP-CONNECTION	639
8.1.1.6.2.1. New parameters in the target model	639
8.1.1.7. Network Management	639
8.1.1.7.1. NM-CONFIG	639
8.1.1.7.2. NM-ECU	640
8.1.1.7.2.1. New parameters in the target model	640
8.1.1.7.3. CAN-NM-CLUSTER	641
8.1.1.7.3.1. New parameters in the target model	641
8.1.1.7.4. CAN-NM-NODE	642
8.1.1.7.4.1. New parameters in the target model	642
8.1.1.7.5. CAN-NM-CLUSTER-COUPLING	643
8.1.1.7.5.1. New parameters in the target model	643
8.1.1.7.6. FLEXRAY-NM-CLUSTER	643
8.1.1.7.6.1. New parameters in the target model	644
8.1.1.7.7. FLEXRAY-NM-NODE	644
8.1.1.7.7.1. New parameters in the target model	644
8.1.1.7.8. FLEXRAY-NM-CLUSTER-COUPLING	645
8.1.1.8. Software Component Description	645
8.1.1.8.1. COMPOSITION-SW-COMPONENT-TYPE	645
8.1.1.8.2. APPLICATION-SW-COMPONENT-TYPE	646
8.1.1.8.2.1. New parameters in the target model	646
8.1.1.8.3. SERVICE-SW-COMPONENT-TYPE	646
8.1.1.8.3.1. New parameters in the target model	647
8.1.1.8.4. COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE	647
8.1.1.8.4.1. New parameters in the target model	647
8.1.1.8.5. ECU-ABSTRACTION-SW-COMPONENT-TYPE	648
8.1.1.8.5.1. New parameters in the target model	648
8.1.1.8.6. PARAMETER-SW-COMPONENT-TYPE	648
8.1.1.8.6.1. New parameters in the target model	649
8.1.1.8.7. R-PORT-PROTOTYPE	649
8.1.1.8.8. P-PORT-PROTOTYPE	649
8.1.1.8.9. CLIENT-SERVER-INTERFACE	650
8.1.1.8.10. ASSEMBLY-SW-CONNECTOR	650
8.1.1.8.11. DELEGATION-SW-CONNECTOR	650
8.1.1.8.12. SERVICE-CONNECTOR-PROTOTYPE	651
8.1.1.8.13. PARAMETER-INTERFACE	651
8.1.1.8.14. MODE-SWITCH-INTERFACE	651
8.1.1.8.15. MODE-GROUP	652
8.1.1.8.16. SW-COMPONENT-PROTOTYPE	652
8.1.1.8.17. VARIABLE-DATA-PROTOTYPE	652



8.1.1.8.17.1. Reason for not upgraded source model parameter IS-QUEUED	653
8.1.1.8.18. ARGUMENT-DATA-PROTOTYPE	653
8.1.1.8.19. PARAMETER-DATA-PROTOTYPE	653
8.1.1.8.19.1. Configuration of target model parameter INIT-VALUE	653
8.1.1.8.20. CLIENT-SERVER-OPERATION	654
8.1.1.8.21. APPLICATION-ERROR	654
8.1.1.8.22. NONQUEUED-RECEIVER-COM-SPEC	654
8.1.1.8.22.1. Processing of source model parameter HANDLE-INVALID	655
8.1.1.8.23. QUEUED-RECEIVER-COM-SPEC	655
8.1.1.8.24. CLIENT-COM-SPEC	655
8.1.1.8.25. PARAMETER-REQUIRE-COM-SPEC	656
8.1.1.8.25.1. Upgrade of source model INIT-VALUE	656
8.1.1.8.26. QUEUED-SENDER-COM-SPEC	656
8.1.1.8.27. NONQUEUED-SENDER-COM-SPEC	657
8.1.1.8.28. SERVER-COM-SPEC	657
8.1.1.8.29. MODE-SWITCH-SENDER-COM-SPEC	657
8.1.1.8.30. PARAMETER-PROVIDE-COM-SPEC	658
8.1.1.8.30.1. Upgrade of source model INIT-VALUE	658
8.1.1.8.31. TRANSMISSION-ACKNOWLEDGE	658
8.1.1.8.32. DATA-FILTER	658
8.1.1.8.32.1. Configuration of target model parameter DATA-FILTER-TYPE	659
8.1.1.8.33. VARIABLE-ACCESS	659
8.1.1.8.34. PARAMETER-ACCESS	660
8.1.1.8.34.1. SHORT-NAME of target model ACCESSED-VARIABLE	661
8.1.1.8.35. RUNNABLE-ENTITY	661
8.1.1.8.35.1. Migration of source model DATA-RECEIVE-POINT elements...	662
8.1.1.8.35.2. Creation of target model ASYNCHRONOUS-SERVER-CALL-RESULT-POINT elements	662
8.1.1.8.36. WAIT-POINT	662
8.1.1.8.37. SWC-INTERNAL-BEHAVIOR	663
8.1.1.8.37.1. Creation of target model PARAMETER-DATA-PROTO-TYPE / INIT-VALUE for INIT-VALUES / LOCAL-PARAMETER-INIT-VALUE-ASSIGNMENT	664
8.1.1.8.37.2. Upgrade of source model INTER-RUNNABLE-VARIABLES / INTER-RUNNABLE-VARIABLE	664
8.1.1.8.37.3. Upgrade of source model SERVICE-NEEDSS	664
8.1.1.8.37.4. Creation of target model DATA-TYPE-MAPPING-REFS / DATA-TYPE-MAPPING-REF	664
8.1.1.8.38. PORT-API-OPTION	664
8.1.1.8.38.1. Upgrade of PORT-ARG-VALUES	665



8.1.1.8.39. DATA-SEND-COMPLETED-EVENT	665
8.1.1.8.40. DATA-RECEIVED-EVENT	665
8.1.1.8.41. DATA-RECEIVE-ERROR-EVENT	666
8.1.1.8.42. ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT	666
8.1.1.8.42.1. Upgrade of source model EVENT-SOURCE-REF	666
8.1.1.8.43. TIMING-EVENT	666
8.1.1.8.44. OPERATION-INVOKED-EVENT	667
8.1.1.8.45. SWC-MODE-SWITCH-EVENT	667
8.1.1.8.46. MODE-SWITCHED-ACK-EVENT	667
8.1.1.8.47. ASYNCHRONOUS-SERVER-CALL-POINT	668
8.1.1.8.47.1. Upgrade of source model OPERATION-IREFS/OPER- ATION-IREF	668
8.1.1.8.48. SYNCHRONOUS-SERVER-CALL-POINT	668
8.1.1.8.48.1. Upgrade of source model OPERATION-IREFS/OPER- ATION-IREF	668
8.1.1.8.49. NUMERICAL-VALUE-SPECIFICATION	668
8.1.1.8.49.1. Upgrade of source model VALUE	669
8.1.1.8.50. TEXT-VALUE-SPECIFICATION	669
8.1.1.8.51. ARRAY-VALUE-SPECIFICATION	669
8.1.1.8.52. RECORD-VALUE-SPECIFICATION	670
8.1.1.8.53. CONSTANT-SPECIFICATION	670
8.1.1.8.54. CONSTANT-REFERENCE	670
8.1.1.8.55. UNIT	670
8.1.1.8.55.1. No upgrade of source model DISPLAY-NAME and PHYSI- CAL-DIMENSION-REF	671
8.1.1.8.56. UNIT-GROUP	671
8.1.1.8.57. SW-BASE-TYPE	671
8.1.1.8.57.1. Upgrade of source model BASE-TYPE-ENCODING	671
8.1.1.8.57.2. Upgrade of source model CATEGORY	672
8.1.1.8.57.3. No upgrade of source model MAX-BASE-TYPE-SIZE and BASE-TYPE-REF	672
8.1.1.8.57.4. No upgrade of source model BYTE-ORDER	672
8.1.1.8.58. SW-DATA-DEF-PROPS	672
8.1.1.8.58.1. No upgrade of source model parameters DATA-CONSTR- REF, DISPLAY-FORMAT, SW-CALPRM-AXIS-SET, SW-CLASS-AT- TR-IMPL-REF, SW-CLASS-REF, SW-CODE-SYNTAX-REF, SW-DA- TA-DEPENDENCY , SW-HOST-VARIABLE, SW-POINTER, SW-RECORD- LAYOUT-REF, SW-TEXT-PROPS, SW-VALUE-BLOCK-SIZE, SW- VARIABLE-ACCESS-IMPL-POLICY	673
8.1.1.8.58.2. No configuration of target model parameters SW-ALIGN- MENT, SW-COMPARISON-VARIABLES, SW-DATA-DEPENDENCY, SW- HOST-VARIABLE, SW-INTENDED-RESOLUTION, SW-INTERPO-	



LATION-METHOD, SW-IS-VIRTUAL, SW-POINTER-TARGET-PROPS, SW-REFRESH-TIMING, SW-TEXT-PROPS, VALUE-AXIS-DATA-TYPE- REF	673
8.1.1.8.58.3. Configuration of SW-DATA-DEF-PROPS-CONDITIONAL/SW- IMPL-POLICY	673
8.1.1.8.58.4. Creation of SW-DATA-DEF-PROPS elements in the target mod- el	673
8.1.1.8.59. IMPLEMENTATION-DATA-TYPE	674
8.1.1.8.59.1. Creation of AUTOSAR 4.0 platform data types	674
8.1.1.8.59.2. Configuration of target model IMPLEMENTATION-DA- TA-TYPE/CATEGORY platform data types	674
8.1.1.8.59.3. Configuration of target model DATA-CONSTR	674
8.1.1.8.59.4. Assignment of AUTOSAR 4.0 platform data type to target mod- el IMPLEMENTATION-DATA-TYPE	674
8.1.1.8.59.5. ARRAY-TYPE	675
8.1.1.8.59.6. RECORD-TYPE	675
8.1.1.8.60. COMPU-METHOD	675
8.1.1.8.61. COMPU-SCALE	675
8.1.1.8.62. EXCLUSIVE-AREA	676
8.1.1.8.63. COM-MGR-USER-NEEDS	676
8.1.1.8.63.1. No upgrade of source model SWC-COM-MGR-USER-NEEDS / CALLBACK-PORTS and SWC-COM-MGR-USER-NEEDS/SERVICE-CALL- PORTS	676
8.1.1.8.64. DIAGNOSTIC-COMMUNICATION-MANAGER-NEEDS	677
8.1.1.8.64.1. No upgrade of source model SWC-DIAGNOSTIC-COMMU- NATION-NEEDS/CALLBACK-PORTS and SWC-DIAGNOSTIC-COMMU- NATION-NEEDS/SERVICE-CALL-PORT	677
8.1.1.8.65. DIAGNOSTIC-EVENT-NEEDS	677
8.1.1.8.65.1. No upgrade of source model SWC-DIAGNOSTIC-EVENT- NEEDS/CALLBACK-PORTS and SWC-DIAGNOSTIC-EVENT-NEEDS / SERVICE-CALL-PORTS	677
8.1.1.8.65.2. No configuration of target model parameters CONSIDER-PTO- STATUS, DIAG-EVENT-DEBOUNCE-ALGORITHM, DIAG-REQUIREMENT, DTC-KIND, DTC-NUMBER, INHIBITING-FID-REF	677
8.1.1.8.66. ECU-STATE-MGR-USER-NEEDS	677
8.1.1.8.66.1. No upgrade of source model SWC-ECU-STATE-MGR-USER- NEEDS/SERVICE-CALL-PORTS	678
8.1.1.8.66.2. No configuration of target model parameters CONSIDER-PTO- STATUS, DIAG-EVENT-DEBOUNCE-ALGORITHM, DIAG-REQUIREMENT, DTC-KIND, DTC-NUMBER, INHIBITING-FID-REF	678
8.1.1.8.67. FUNCTION-INHIBITION-NEEDS	678



8.1.1.8.67.1. No upgrade of source model SWC-FUNCTION-INHIBITION-NEEDS/SERVICE-CALL-PORTS	678
8.1.1.8.68. NV-BLOCK-NEEDS	678
8.1.1.8.68.1. No upgrade of source model SWC-NV-BLOCK-NEEDS/CALL-BACK-PORTS, SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF, SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF, SWC-NV-BLOCK-NEEDS/SERVICE-CALL-PORTS	679
8.1.1.8.68.2. No configuration of target model parameters CALC-RAM-BLOCK-CRC, CHECK-STATIC-BLOCK-ID, N-ROM-BLOCKS, STORE-AT-SHUTDOWN, WRITE-VERIFICATION	679
8.1.1.8.68.3. No configuration of target model parameter RELIABILITY	679
8.1.1.8.68.4. Additional configuration of ASSIGNED-DATAS/ROLE-BASED-DATA-ASSIGNMENT for SWC-NV-BLOCK-NEEDS elements	679
8.1.1.8.69. OBD-CONTROL-SERVICE-NEEDS	680
8.1.1.8.69.1. No upgrade of source model SWC-OBD-CONTROL-SERVICE-NEEDS/CALLBACK-PORT	680
8.1.1.8.69.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL	680
8.1.1.8.70. OBD-INFO-SERVICE-NEEDS	680
8.1.1.8.70.1. No upgrade of source model SWC-OBD-INFO-SERVICE-NEEDS/CALLBACK-PORT	680
8.1.1.8.70.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL	681
8.1.1.8.71. OBD-PID-SERVICE-NEEDS	681
8.1.1.8.71.1. No upgrade of source model SWC-OBD-PID-SERVICE-NEEDS/CALLBACK-PORT	681
8.1.1.8.71.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL	681
8.1.1.8.72. OBD-RATIO-SERVICE-NEEDS	681
8.1.1.8.72.1. No upgrade of source model SWC-OBD-RATIO-SERVICE-NEEDS/SERVICE-CALL-PORT	682
8.1.1.8.72.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL	682
8.1.1.8.73. SUPERVISED-ENTITY-NEEDS	682
8.1.1.8.73.1. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL	682
8.1.1.8.74. PER-INSTANCE-MEMORY	683
8.1.1.8.74.1. Configuration of target model parameter INIT-VALUE	683
8.1.1.8.75. PER-INSTANCE-MEMORY-SIZE	683
8.1.1.8.76. SWC-IMPLEMENTATION	683
8.1.1.8.76.1. No upgrade of source model parameters COM-PILERS/COMPILER, IMPLEMENTATION-DEPENDENCIES/DEPENDEN-	



CY-ON-FILE, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-LIBRARY, LINKERS/LINKER, PROCESSOR-REFS/PROCESSOR-REF, RESOURCE-CONSUMPTION	684
8.1.1.8.76.2. Upgrade of source model parameter PROGRAMMING-LANGUAGE	684
8.1.1.8.76.3. Upgrade of source model parameter VENDOR-ID	684
8.1.1.8.76.4. Creation of target model RESOURCE-CONSUMPTION	684
8.1.1.8.77. CODE	684
8.1.1.8.77.1. Configuration of source model parameter TYPE	685
8.1.1.8.78. MODE-DECLARATION	685
8.1.1.8.79. MODE-DECLARATION-GROUP	685
8.1.1.8.80. DISABLED-MODE-IREF	685
8.1.1.8.81. MODE-SWITCHED-ACK	686
8.1.1.8.82. MODE-ACCESS-POINT	686
8.1.1.8.82.1. Configuration MODE-ACCESS-POINT out of source model R-PORT-PROTOTYPE elements	686
8.1.1.8.82.2. Configuration MODE-ACCESS-POINT out of source model MODE-SWITCH-POINT elements	686
8.1.1.8.83. MODE-SWITCH-POINT	687
8.1.1.8.84. SW-ADDR-METHOD	687
8.1.1.8.85. MEMORY-SECTION	687
8.1.1.8.85.1. Configuration of source model parameters SW-ADDR-METHOD-REFS/SW-ADDR-METHOD-REF	687
8.1.1.8.86. SYSTEM-MAPPING	687
8.1.1.8.86.1. No upgrade of source model parameters ECU-RESOURCE-MAPPINGS, MAPPING-CONSTRAINTS, RESOURCE-ESTIMATIONS and SIGNAL-PATH-CONSTRAINTS	688
8.1.1.8.87. SWC-TO-ECU-MAPPING	688
8.1.1.8.87.1. No configuration of target model parameters PARTITION-REF and PROCESSING-UNIT-REF	688
8.1.1.8.88. SWC-TO-IMPL-MAPPING	689
8.1.1.8.89. SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING	689
8.1.1.8.90. SENDER-RECEIVER-TO-SIGNAL-MAPPING	689
8.1.1.8.91. CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING	690
8.1.1.8.92. SENDER-REC-RECORD-TYPE-MAPPING	691
8.1.1.8.93. SENDER-REC-RECORD-ELEMENT-MAPPING	691
8.1.1.8.94. SENDER-REC-ARRAY-TYPE-MAPPING	691
8.1.1.8.95. SENDER-REC-ARRAY-ELEMENT-MAPPING	692
8.1.1.8.96. END-TO-END-PROTECTION-SET	692
8.1.1.8.97. END-TO-END-PROTECTION	692
8.1.1.8.98. END-TO-END-PROTECTION-I-SIGNAL-I-PDU	693
8.1.1.8.99. END-TO-END-PROTECTION-VARIABLE-PROTOTYPE	693



8.1.2. Upgrading a system model from AUTOSAR 3.2.x to AUTOSAR 4.0.3	694
8.1.2.1. Topology	694
8.1.2.1.1. SYSTEM	694
8.1.2.1.1.1. Changed or moved parameters	695
8.1.2.1.2. ECU-INSTANCE	695
8.1.2.1.2.1. Reason for not upgraded source model parameter	696
8.1.2.1.3. CAN-CLUSTER	696
8.1.2.1.3.1. Changed or moved parameters	696
8.1.2.1.4. CAN-COMMUNICATION-CONTROLLER	697
8.1.2.1.4.1. Changed or moved parameters	698
8.1.2.1.5. CAN-COMMUNICATION-CONNECTOR	698
8.1.2.1.5.1. Changed or moved parameters	698
8.1.2.1.6. CAN-PHYSICAL-CHANNEL	699
8.1.2.1.6.1. Changed or moved parameters	699
8.1.2.1.7. FLEXRAY-CLUSTER	699
8.1.2.1.7.1. Changed or moved parameters	701
8.1.2.1.7.2. Reason for not upgraded source model parameter	702
8.1.2.1.8. FLEXRAY-COMMUNICATION-CONTROLLER	702
8.1.2.1.8.1. Changed or moved parameters	703
8.1.2.1.8.2. Reason for not upgraded source model parameter	704
8.1.2.1.9. FLEXRAY-FIFO-CONFIGURATION	704
8.1.2.1.9.1. Changed or moved parameters	704
8.1.2.1.10. FLEXRAY-FIFO-RANGE	704
8.1.2.1.11. FLEXRAY-COMMUNICATION-CONNECTOR	705
8.1.2.1.11.1. Changed or moved parameters	705
8.1.2.1.12. FLEXRAY-PHYSICAL-CHANNEL	705
8.1.2.1.12.1. Changed or moved parameters	706
8.1.2.1.13. LIN-CLUSTER	706
8.1.2.1.14. LIN-MASTER	706
8.1.2.1.15. LIN-SLAVE	707
8.1.2.1.16. LIN-ERROR-RESPONSE	707
8.1.2.1.17. LIN-COMMUNICATION-CONNECTOR	707
8.1.2.1.17.1. Changed or moved parameters	707
8.1.2.1.18. LIN-PHYSICAL-CHANNEL	708
8.1.2.1.18.1. Changed or moved parameters	708
8.1.2.2. Communication	708
8.1.2.2.1. CAN-FRAME-TRIGGERING	708
8.1.2.2.2. FLEXRAY-FRAME-TRIGGERING	709
8.1.2.2.3. FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING	709
8.1.2.2.4. LIN-FRAME-TRIGGERING	709
8.1.2.2.4.1. Changed or moved parameters	710
8.1.2.2.5. CAN-FRAME	710



8.1.2.2.6. FLEXRAY-FRAME	710
8.1.2.2.7. LIN-UNCONDITIONAL-FRAME	710
8.1.2.2.8. LIN-SPORADIC-FRAME	710
8.1.2.2.9. LIN-EVENT-TRIGGERED-FRAME	710
8.1.2.2.10. LIN-SCHEDULE-TABLE	710
8.1.2.2.11. ASSIGN-FRAME-ID	711
8.1.2.2.12. UNASSIGN-FRAME-ID	711
8.1.2.2.13. ASSIGN-NAD	711
8.1.2.2.14. FREE-FORMAT	711
8.1.2.2.15. PDU-TRIGGERING	711
8.1.2.2.16. I-SIGNAL-TRIGGERING	711
8.1.2.2.17. DCM-I-PDU	711
8.1.2.2.18. I-SIGNAL-I-PDU	711
8.1.2.2.19. MULTIPLEXED-I-PDU	711
8.1.2.2.19.1. Changed or moved parameters	712
8.1.2.2.20. DYNAMIC-PART	712
8.1.2.2.21. DYNAMIC-PART-ALTERNATIVE	712
8.1.2.2.22. STATIC-PART	713
8.1.2.2.23. SEGMENT-POSITION	713
8.1.2.2.24. N-PDU	713
8.1.2.2.24.1. Changed or moved parameters	714
8.1.2.2.25. NM-PDU	714
8.1.2.2.25.1. Changed or moved parameters	714
8.1.2.2.26. USER-DEFINED-PDU	714
8.1.2.2.27. USER-DEFINED-I-PDU	715
8.1.2.2.27.1. Changed or moved parameters	715
8.1.2.2.28. PDU-TO-FRAME-MAPPING	715
8.1.2.2.29. I-PDU-TIMING	715
8.1.2.2.30. I-SIGNAL-I-PDU-GROUP	715
8.1.2.2.31. PDUR-I-PDU-GROUP	716
8.1.2.2.32. I-SIGNAL-TO-I-PDU-MAPPING	716
8.1.2.2.33. I-SIGNAL	716
8.1.2.2.34. I-SIGNAL-GROUP	716
8.1.2.2.35. SYSTEM-SIGNAL	716
8.1.2.2.36. I-SIGNAL-PORT	716
8.1.2.2.37. FRAME-PORT	717
8.1.2.2.38. I-PDU-PORT	717
8.1.2.2.39. TRANSMISSION-MODE-DECLARATION	717
8.1.2.2.39.1. Changed or moved parameters	717
8.1.2.2.40. TRANSMISSION-MODE-CONDITION	717
8.1.2.2.41. TRANSMISSION-MODE-FALSE-TIMING	717
8.1.2.2.42. EVENT-CONTROLLED-TIMING	717



8.1.2.2.43. CYCLIC-TIMING	718
8.1.2.2.44. TIME-RANGE-TYPE	718
8.1.2.2.45. ABSOLUTE-TOLERANCE	718
8.1.2.2.46. RELATIVE-TOLERANCE	718
8.1.2.3. Gateway	718
8.1.2.3.1. FRAME-MAPPING	718
8.1.2.3.2. I-PDU-MAPPING	718
8.1.2.3.3. TARGET-I-PDU-REF	719
8.1.2.3.4. PDU-MAPPING-DEFAULT-VALUE	719
8.1.2.3.5. DEFAULT-VALUE-ELEMENT	719
8.1.2.3.6. I-SIGNAL-MAPPING	719
8.1.2.4. CAN Transport Layer	719
8.1.2.4.1. CAN-TP-CONFIG	719
8.1.2.4.1.1. New parameters in the target model	719
8.1.2.4.1.2. Moved parameters in the target model	719
8.1.2.4.2. CAN-TP-NODE	720
8.1.2.4.2.1. New parameters in the target model	720
8.1.2.4.3. CAN-TP-ADDRESS	721
8.1.2.4.4. CAN-TP-CHANNEL	721
8.1.2.4.5. CAN-TP-CONNECTION	721
8.1.2.4.5.1. New parameters in the target model	722
8.1.2.4.5.2. Changed or moved parameters	722
8.1.2.4.6. Second CAN-TP-CONNECTION for FULL-DUPLEX-MODE	722
8.1.2.5. FlexRay ISO Transport Layer	723
8.1.2.5.1. FLEXRAY-TP-CONFIG	723
8.1.2.5.1.1. New parameters in the target model	723
8.1.2.5.1.2. Moved parameters in the target model	724
8.1.2.5.2. FLEXRAY-TP-ECU	724
8.1.2.5.3. FLEXRAY-TP-NODE	725
8.1.2.5.4. TP-ADDRESS	725
8.1.2.5.5. FLEXRAY-TP-CONNECTION	725
8.1.2.5.6. FLEXRAY-TP-CONNECTION-CONTROL	726
8.1.2.5.7. FLEXRAY-TP-PDU-POOL	726
8.1.2.6. LIN Transport Layer	727
8.1.2.6.1. LIN-TP-CONFIG	727
8.1.2.6.1.1. Moved parameters in the target model	727
8.1.2.6.1.2. LIN-TP-NODE	727
8.1.2.6.1.3. TP-ADDRESS	728
8.1.2.6.1.4. LIN-TP-CONNECTION	728
8.1.2.7. Network Management	728
8.1.2.7.1. NM-CONFIG	728
8.1.2.7.2. NM-ECU	729



8.1.2.7.2.1. New parameters in the target model	729
8.1.2.7.3. CAN-NM-CLUSTER	729
8.1.2.7.3.1. New parameters in the target model	730
8.1.2.7.4. CAN-NM-NODE	730
8.1.2.7.4.1. New parameters in the target model	730
8.1.2.7.5. CAN-NM-CLUSTER-COUPLING	731
8.1.2.7.5.1. New parameters in the target model	731
8.1.2.7.6. FLEXRAY-NM-CLUSTER	731
8.1.2.7.6.1. New parameters in the target model	732
8.1.2.7.7. FLEXRAY-NM-NODE	732
8.1.2.7.7.1. New parameters in the target model	732
8.1.2.7.8. FLEXRAY-NM-CLUSTER-COUPLING	733
8.1.2.8. Software Component Description	733
8.1.2.8.1. COMPOSITION-SW-COMPONENT-TYPE	733
8.1.2.8.2. APPLICATION-SW-COMPONENT-TYPE	733
8.1.2.8.2.1. New parameters in the target model	733
8.1.2.8.3. SERVICE-SW-COMPONENT-TYPE	733
8.1.2.8.3.1. New parameters in the target model	734
8.1.2.8.4. COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE	734
8.1.2.8.4.1. New parameters in the target model	734
8.1.2.8.5. ECU-ABSTRACTION-SW-COMPONENT-TYPE	735
8.1.2.8.5.1. New parameters in the target model	735
8.1.2.8.6. PARAMETER-SW-COMPONENT-TYPE	735
8.1.2.8.6.1. New parameters in the target model	736
8.1.2.8.7. R-PORT-PROTOTYPE	736
8.1.2.8.8. P-PORT-PROTOTYPE	736
8.1.2.8.9. CLIENT-SERVER-INTERFACE	736
8.1.2.8.10. DELEGATION-SW-CONNECTOR	736
8.1.2.8.11. ASSEMBLY-SW-CONNECTOR	736
8.1.2.8.12. SERVICE-CONNECTOR-PROTOTYPE	736
8.1.2.8.13. PARAMETER-INTERFACE	736
8.1.2.8.14. MODE-SWITCH-INTERFACE	737
8.1.2.8.15. MODE-GROUP	737
8.1.2.8.16. SW-COMPONENT-PROTOTYPE	737
8.1.2.8.17. VARIABLE-DATA-PROTOTYPE	737
8.1.2.8.18. ARGUMENT-DATA-PROTOTYPE	737
8.1.2.8.19. PARAMETER-DATA-PROTOTYPE	737
8.1.2.8.20. CLIENT-SERVER-OPERATION	737
8.1.2.8.21. APPLICATION-ERROR	738
8.1.2.8.22. NONQUEUED-RECEIVER-COM-SPEC	738
8.1.2.8.22.1. Processing of source model parameter HANDLE-INVALID	738
8.1.2.8.23. QUEUED-RECEIVER-COM-SPEC	739



8.1.2.8.24. CLIENT-COM-SPEC	739
8.1.2.8.25. PARAMETER-REQUIRE-COM-SPEC	739
8.1.2.8.26. QUEUED-SENDER-COM-SPEC	739
8.1.2.8.27. NONQUEUED-SENDER-COM-SPEC	740
8.1.2.8.28. SERVER-COM-SPEC	740
8.1.2.8.29. MODE-SWITCH-SENDER-COM-SPEC	740
8.1.2.8.30. PARAMETER-PROVIDE-COM-SPEC	741
8.1.2.8.31. TRANSMISSION-ACKNOWLEDGE	741
8.1.2.8.32. DATA-FILTER	741
8.1.2.8.32.1. Configuration of target model parameter DATA-FILTER-TYPE	741
8.1.2.8.33. VARIABLE-ACCESS	742
8.1.2.8.34. PARAMETER-ACCESS	742
8.1.2.8.35. RUNNABLE-ENTITY	742
8.1.2.8.35.1. Migration of source model DATA-RECEIVE-POINT elements....	743
8.1.2.8.35.2. Creation of target model ASYNCHRONOUS-SERVER-CALL-RESULT-POINT elements	743
8.1.2.8.36. WAIT-POINT	743
8.1.2.8.37. SWC-INTERNAL-BEHAVIOR	743
8.1.2.8.38. PORT-API-OPTION	744
8.1.2.8.39. DATA-SEND-COMPLETED-EVENT	744
8.1.2.8.40. DATA-RECEIVED-EVENT	744
8.1.2.8.41. DATA-RECEIVE-ERROR-EVENT	744
8.1.2.8.42. ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT	744
8.1.2.8.43. TIMING-EVENT	744
8.1.2.8.44. OPERATION-INVOKED-EVENT	744
8.1.2.8.45. SWC-MODE-SWITCH-EVENT	744
8.1.2.8.46. MODE-SWITCHED-ACK-EVENT	745
8.1.2.8.47. ASYNCHRONOUS-SERVER-CALL-POINT	745
8.1.2.8.48. SYNCHRONOUS-SERVER-CALL-POINT	745
8.1.2.8.49. NUMERICAL-VALUE-SPECIFICATION	745
8.1.2.8.50. TEXT-VALUE-SPECIFICATION	745
8.1.2.8.51. ARRAY-VALUE-SPECIFICATION	745
8.1.2.8.52. RECORD-VALUE-SPECIFICATION	745
8.1.2.8.53. CONSTANT-SPECIFICATION	745
8.1.2.8.54. CONSTANT-REFERENCE	746
8.1.2.8.55. UNIT	746
8.1.2.8.55.1. No upgrade of source model DISPLAY-NAME	746
8.1.2.8.56. UNIT-GROUP	746
8.1.2.8.57. PHYSICAL-DIMENSION	746
8.1.2.8.58. SW-BASE-TYPE	747
8.1.2.8.58.1. Upgrade of source model BASE-TYPE-ENCODING	747



8.1.2.8.58.2. Upgrade of source model CATEGORY	747
8.1.2.8.58.3. No upgrade of source model BASE-TYPE-REF	748
8.1.2.8.58.4. No upgrade of source model BYTE-ORDER	748
8.1.2.8.59. SW-DATA-DEF-PROPS	748
8.1.2.8.59.1. No upgrade of source model parameters SW-CALPRM-AXIS-SET, SW-CLASS-ATTR-IMPL-REF, SW-CLASS-REF, SW-CODE-SYN-TAX-REF, SW-DATA-DEPENDENCY , SW-HOST-VARIABLE, SW-POINTER, SW-RECORD-LAYOUT-REF, SW-TEXT-PROPS, SW-VARIABLE-ACCESS-IMPL-POLICY	749
8.1.2.8.59.2. No configuration of target model parameters SW-ALIGN-MENT, SW-COMPARISON-VARIABLES, SW-DATA-DEPENDENCY, SW-HOST-VARIABLE, SW-INTENDED-RESOLUTION, SW-INTERPOLATION-METHOD, SW-IS-VIRTUAL, SW-REFRESH-TIMING, SW-TEXT-PROPS, VALUE-AXIS-DATA-TYPE-REF	749
8.1.2.8.59.3. Configuration of SW-DATA-DEF-PROPS-CONDITIONAL/SW-IMPL-POLICY	749
8.1.2.8.59.4. Creation of SW-DATA-DEF-PROPS elements in the target model	749
8.1.2.8.60. DATA-CONSTR	749
8.1.2.8.61. DATA-CONSTR-RULE	750
8.1.2.8.62. PHYS-CONSTRS	750
8.1.2.8.62.1. No configuration of target model parameters MAX-DIFF, MAX-GRADIENT, MONOTONY, SCALE-CONSTRS	750
8.1.2.8.62.2. Configuration of target model parameter UNIT-REF	750
8.1.2.8.63. INTERNAL-CONSTRS	751
8.1.2.8.63.1. No configuration of target model parameters MAX-DIFF, MAX-GRADIENT, MONOTONY, SCALE-CONSTRS	751
8.1.2.8.64. IMPLEMENTATION-DATA-TYPE	751
8.1.2.8.65. COMPU-METHOD	751
8.1.2.8.66. COMPU-SCALE	751
8.1.2.8.67. EXCLUSIVE-AREA	752
8.1.2.8.68. COM-MGR-USER-NEEDS	752
8.1.2.8.69. DIAGNOSTIC-COMMUNICATION-MANAGER-NEEDS	752
8.1.2.8.70. DIAGNOSTIC-EVENT-NEEDS	752
8.1.2.8.71. ECU-STATE-MGR-USER-NEEDS	752
8.1.2.8.72. FUNCTION-INHIBITION-NEEDS	752
8.1.2.8.73. NV-BLOCK-NEEDS	752
8.1.2.8.74. OBD-CONTROL-SERVICE-NEEDS	753
8.1.2.8.75. OBD-PID-SERVICE-NEEDS	753
8.1.2.8.76. OBD-RATIO-SERVICE-NEEDS	753
8.1.2.8.77. SUPERVISED-ENTITY-NEEDS	753
8.1.2.8.78. PER-INSTANCE-MEMORY	753



8.1.2.8.78.1. Configuration of target model parameter INIT-VALUE	753
8.1.2.8.78.2. Creation of target model parameter SW-DATA-DEF-PROPS	753
8.1.2.8.79. PER-INSTANCE-MEMORY-SIZE	754
8.1.2.8.80. SWC-IMPLEMENTATION	754
8.1.2.8.81. CODE	754
8.1.2.8.82. MODE-DECLARATION	754
8.1.2.8.83. MODE-DECLARATION-GROUP	754
8.1.2.8.84. DISABLED-MODE-IREF	754
8.1.2.8.85. MODE-SWITCHED-ACK	754
8.1.2.8.86. MODE-ACCESS-POINT	754
8.1.2.8.87. MODE-SWITCH-POINT	755
8.1.2.8.88. SW-ADDR-METHOD	755
8.1.2.8.89. MEMORY-SECTION	755
8.1.2.8.90. SYSTEM-MAPPING	755
8.1.2.8.90.1. No upgrade of source model parameters ECU-RESOURCE-MAPPINGS, MAPPING-CONSTRAINTS, RESOURCE-ESTIMATIONS and SIGNAL-PATH-CONSTRAINTS	755
8.1.2.8.91. SWC-TO-ECU-MAPPING	755
8.1.2.8.92. SWC-TO-IMPL-MAPPING	756
8.1.2.8.93. SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING	756
8.1.2.8.94. SENDER-RECEIVER-TO-SIGNAL-MAPPING	756
8.1.2.8.95. CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING	756
8.1.2.8.96. SENDER-REC-RECORD-TYPE-MAPPING	756
8.1.2.8.97. SENDER-REC-RECORD-ELEMENT-MAPPING	756
8.1.2.8.98. SENDER-REC-ARRAY-TYPE-MAPPING	756
8.1.2.8.99. SENDER-REC-ARRAY-ELEMENT-MAPPING	756
8.1.2.8.100. PNC-MAPPING	757
8.1.2.8.101. END-TO-END-PROTECTION-SET	757
8.1.2.8.102. END-TO-END-PROTECTION	757
8.1.2.8.103. END-TO-END-PROTECTION-I-SIGNAL-I-PDU	757
8.1.2.8.104. END-TO-END-PROTECTION-VARIABLE-PROTOTYPE	757
8.1.3. Upgrading a system model from AUTOSAR 3.2.2 to AUTOSAR 4.0.3	757
8.1.3.1. Topology	758
8.1.3.1.1. CAN-CLUSTER	758
8.1.3.1.1.1. Changed or moved parameters	758
8.1.3.1.2. CAN-COMMUNICATION-CONNECTOR	759
8.1.3.1.2.1. Changed or moved parameters	759
8.1.3.1.3. FLEXRAY-CLUSTER	760
8.1.3.1.3.1. Changed or moved parameters	761
8.1.3.1.3.2. Reason for not upgraded source model parameter	762
8.1.3.1.4. FLEXRAY-COMMUNICATION-CONNECTOR	762
8.1.3.1.4.1. Changed or moved parameters	763



8.1.3.1.5. LIN-CLUSTER	763
8.1.3.1.5.1. Changed or moved parameters	763
8.1.3.1.6. LIN-COMMUNICATION-CONNECTOR	764
8.1.3.1.6.1. Changed or moved parameters	764
8.1.3.1.7. LIN-SLAVE-CONFIG	764
8.1.3.1.7.1. New parameters in the target model	765
8.1.3.2. FlexRay ISO Transport Layer	765
8.1.3.2.1. FLEXRAY-TP-ECU	765
8.1.3.3. Network Management	766
8.1.3.3.1. NM-COORDINATOR	766
8.1.3.4. Software Component Description	766
8.1.3.4.1. APPLICATION-SW-COMPONENT-TYPE	766
8.1.3.4.1.1. New parameters in the target model	767
8.1.3.4.2. SERVICE-SW-COMPONENT-TYPE	767
8.1.3.4.2.1. New parameters in the target model	767
8.1.3.4.3. COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE	767
8.1.3.4.3.1. New parameters in the target model	768
8.1.3.4.4. ECU-ABSTRACTION-SW-COMPONENT-TYPE	768
8.1.3.4.4.1. New parameters in the target model	768
8.1.3.4.5. NV-BLOCK-SW-COMPONENT-TYPE	769
8.1.3.4.5.1. New parameters in the target model	769
8.1.3.4.6. NV-BLOCK-DESCRIPTOR	769
8.1.3.4.6.1. No configuration of target model parameters CONS- TANT-VALUE-MAPPING-REFS, DATA-TYPE-MAPPING-REFS, INSTAN- TIATION-DATA-DEF-PROPS	770
8.1.3.4.6.2. New parameters in the target model	770
8.1.3.4.7. NV-BLOCK-DATA-MAPPING	770
8.1.3.4.8. ROLE-BASED-PORT-ASSIGNMENT	770
8.1.3.4.9. P-PORT-PROTOTYPE	771
8.1.3.4.10. NONQUEUED-RECEIVER-COM-SPEC	771
8.1.3.4.10.1. Processing of source model parameter HANDLE-INVALID	772
8.1.3.4.11. QUEUED-RECEIVER-COM-SPEC	772
8.1.3.4.12. NV-PROVIDE-COM-SPEC	773
8.1.3.4.13. RUNNABLE-ENTITY	773
8.1.3.4.13.1. Migration of source model DATA-RECEIVE-POINT elements....	774
8.1.3.4.13.2. Creation of target model ASYNCHRONOUS-SERVER-CALL- RESULT-POINT elements	774
8.1.3.4.14. R-PORT-PROTOTYPE	775
8.1.3.4.15. NV-REQUIRE-COM-SPEC	775
8.1.3.4.16. NV-DATA-INTERFACE	775
8.1.3.4.17. COMPU-SCALE	776
8.1.3.4.18. NV-BLOCK-NEEDS	776

8.1.3.4.18.1. No upgrade of source model SWC-NV-BLOCK-NEEDS / CALL-BACK-PORTS, SWC-NV-BLOCK-NEEDS/NV-DATA-PORT-PROTOTYPE-REFS, SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF, SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF, SWC-NV-BLOCK-NEEDS/SERVICE-CALL-PORTS	777
8.1.3.4.18.2. No upgrade of source model NV-BLOCK-NEEDS/RAM-BLOCK-STATUS-CONTROL	777
8.1.3.4.18.3. No configuration of target model parameters CALC-RAM-BLOCK-CRC, CHECK-STATIC-BLOCK-ID, N-ROM-BLOCKS, WRITE-VERIFICATION	777
8.1.3.4.18.4. No configuration of target model parameter RELIABILITY	777
8.1.3.4.18.5. Additional configuration of ASSIGNED-DATAS/ROLE-BASED-DATA-ASSIGNMENT for SWC-NV-BLOCK-NEEDS elements	777
8.1.3.4.19. END-TO-END-PROTECTION	778
8.1.3.5. BSW Module Description	778
8.1.3.5.1. BSW-MODULE-DESCRIPTION	778
8.1.3.5.1.1. New parameters in the target model	779
8.1.3.5.2. BSW-MODULE-ENTRY	779
8.1.3.5.2.1. Configuration of target model parameter EXECUTION-CONTEXT	780
8.1.3.5.2.2. Configuration of target model parameter IS-REENTRANT	780
8.1.3.5.2.3. Configuration of target model parameter SW-SERVICE-IM-PL-POLICY	780
8.1.3.5.3. SW-SERVICE-ARG	780
8.1.3.5.4. RETURN-TYPE	780
8.1.3.5.5. BSW-MODULE-DEPENDENCY	781
8.1.3.5.5.1. Configuration of SERVICE-NEEDS parameters in BSW-MODULE-DEPENDENCY/SERVICE-ITEMS	781
8.1.3.5.6. BSW-INTERNAL-BEHAVIOR	781
8.1.3.5.6.1. No configuration of BSW-BACKGROUND-EVENT, BSW-EXTERNAL-TRIGGER-OCCURRED-EVENT, BSW-MODE-SWITCH-EVENT, BSW-MODE-SWITCHED-ACK-EVENT in target model	782
8.1.3.5.6.2. Configuration of source model parameter MODULE-REF	782
8.1.3.5.7. BSW-INTERRUPT-ENTITY	782
8.1.3.5.8. BSW-SCHEDULABLE-ENTITY	783
8.1.3.5.9. BSW-CALLED-ENTITY	784
8.1.3.5.10. BSW-TIMING-EVENT	785
8.1.3.5.11. BSW-INTERNAL-TRIGGER-OCCURRED-EVENT	785
8.1.3.5.11.1. Creation of target model element BSW-INTERNAL-BEHAVIOR/INTERNAL-TRIGGERING-POINTS/BSW-INTERNAL-TRIGGERING-POINT	786
8.1.3.5.12. BSW-IMPLEMENTATION	786

8.1.3.5.12.1. No upgrade of source model parameters COM-PILERS/COMPILER, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-FILE, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-LIBRARY, LINKERS/LINKER, PROCESSOR-REFS/PROCESSOR-REF, RESOURCE-CONSUMPTION, PRECONFIGURED-CONFIGURATION-REF, RECOMMENDED-CONFIGURATION-REF, REQUIRED-HW-REFS/REQUIRED-HW-REF, VENDOR-SPECIFIC-MODULE-DEF-REF	787
8.1.3.5.12.2. Creation of target model RESOURCE-CONSUMPTION	787
8.1.3.5.12.3. Configuration of target model SWC-BSW-MAPPING-REF	787
8.1.3.5.12.4. Upgrade of source model parameter PROGRAMMING-LANGUAGE	787
8.1.3.5.12.5. Upgrade of source model parameter VENDOR-ID	787
8.1.3.5.13. SWC-BSW-MAPPING	788
8.1.4. AUTOSAR 4.0 platform data types	788
Bibliography	790
A. Third party license	791
Index	797

Begin here

1. If you are upgrading from a previous version

- ▶ What's new in this EB tresos Studio version?

Refer to the document EB tresos Studio new and noteworthy, located in your EB tresos Studio <INSTALL_PATH>/doc directory.

- ▶ Which known problems, fixed problems, incompatibilities to previous releases, limitations and restrictions have been reported for the current EB tresos Studio version?

Refer to the EB tresos Studio release notes, located in your EB tresos Studio <INSTALL_PATH>/doc directory.

2. If you are using EB tresos Studio for the first time

If you are a first time user of EB tresos Studio, you may want to get familiar with some of the concepts behind AUTOSAR at [Section 3.2, “The AUTOSAR concept”](#).

- ▶ What is EB tresos Studio?

The best way to find out more about EB tresos Studio is to read through [Chapter 3, “Background information”](#). In this chapter you find basic information about file formats and basic concepts within EB tresos Studio. You will need this knowledge to understand the rest of the user's guide.

- ▶ How do I start working with EB tresos Studio?

The chapter [Chapter 4, “Using EB tresos Studio for the first time”](#) covers the very first start of EB tresos Studio.

- ▶ What are all these GUI elements for?

To get to know the EB tresos Studio GUI, see [Chapter 5, “The graphical user interface \(GUI\)”](#).

- ▶ Can I customize the EB tresos Studio GUI for my needs?

Yes, you can. Information on how to do that is available at [Section 6.2, “Changing the EB tresos Studio setup”](#).

3. If you want to configure AUTOSAR modules or your own modules with EB tresos Studio

[Chapter 6, “Use cases”](#) is the place where you find instructions for:

- ▶ Starting a new project ([Section 6.3, “Working with projects”](#)).
- ▶ Using EB tresos Studio to edit an ECU configuration ([Section 6.8, “Editing parameters of a module configuration”](#)).
- ▶ Using the importing functions to import DBC, LDF, Fibex, or AUTOSAR files ([Section 6.10, “Importing and exporting”](#)).
- ▶ Generating source code out of your ECU configuration ([Section 6.9, “Generating code for projects”](#)).
- ▶ Using the EB tresos Studio tool on the command line ([Section 6.12, “Working with the command line”](#)).

[Chapter 7, “Tool extensions”](#) is the place where you find instructions for specific editors:

- ▶ Creating connections between ports and adding component prototypes to composition component types ([Section 7.2, “Using the Connection Editor”](#)).
- ▶ Viewing the whole system model ([Section 7.3, “Using the System Model Viewer”](#)).
- ▶ Adding data mappings to data elements ([Section 7.4, “Using the Signal Mapping Editor”](#)).

4. If you want to extend the EB tresos Studio functionality

Refer to the EB tresos Studio developer's guide if you want to extend the EB tresos Studio functionalities and find out how to:

- ▶ work with Eclipse,
- ▶ develop your own modules,
- ▶ work with configuration models,
- ▶ create your own code generator, or
- ▶ extend the EB tresos Studio user interface.

The EB tresos Studio developer's guide is delivered as a separate document, which is located at `$TRESOS_BASE\doc\2.0_EB_tresos_Studio`. `$TRESOS_BASE` is the directory into which you have installed EB tresos Studio.

5. If you need help/more information

- ▶ Technical support

Receive technical support via email or phone from the support hotline.

- ▶ [Chapter 1, "About this documentation"](#)

Find out about:

- ▶ Required knowledge, tools, and system environment
- ▶ Typographical and style conventions used throughout this documentation. Defines usage of special fonts in the documentation.
- ▶ Naming conventions used in this documentation. Defines usage of special names and small/capital lettering in the documentation.

- ▶ [Index "Index"](#)

You cannot find what you are looking for? Try the [Index "Index"](#) list (alphabetically sorted).

- ▶ EB tresos glossary

You do not understand what a word or abbreviations means? Find out in the EB tresos glossary.

- ▶ ["Bibliography"](#)

Would you like to read more detailed information? Find bibliographic references and further reading suggestions in the ["Bibliography"](#).



1. About this documentation

1.1. Introduction

Welcome to the EB tresos Studio documentation.

This chapter About this documentation provides you with

- ▶ Required knowledge, tools, and system environment
- ▶ Typographical and style conventions used throughout this documentation. Defines usage of special fonts in the documentation.
- ▶ Naming conventions used in this documentation. Defines usage of special names and small/capital lettering in the documentation.

NOTE

Note that there is a separate documentation for the EB tresos AutoCore if you also purchased and installed the EB tresos AutoCore modules.



1.2. Required knowledge and system environment

1.2.1. Required knowledge

If you want to configure an AUTOSAR stack with EB tresos Studio, you need to know the following:

- ▶ the EB tresos Studio user's guide
- ▶ the documentation of any other products from the EB tresos product line that you purchased along with EB tresos Studio
- ▶ the documentation of any MCAL products that you purchased along with EB tresos Studio

If you want to extend the EB tresos Studio functionality, you need to know the following:

- ▶ the EB tresos Studio developer's guide
- ▶ Java programming language



- ▶ basic knowledge about developing Eclipse plug-ins

1.2.1. Required system environment

Item	Requirement
Operating system	Microsoft Windows 7, 64-bit Microsoft Windows 10 (tested with Microsoft Windows 10 Enterprise 2016 (Version 10.0.14393 Build 14393)) or Linux Ubuntu, 64-bit (tested with Ubuntu 16.04 LTS)
Processor	Dual-core (minimal) Quad-core (recommended)
RAM	2 GB (minimal) 8 GB (recommended)
Connectors	USB port, if dongled licenses are used
Network	Network connection, if network licenses are used

Table 1.1. Minimal and recommended system requirements

1.3. Interpretation of version information

1.3.1. Product version number

Each product within the product line is assigned an individual *product version number*.

The product version number scheme is made of three parts:

1. The major number

An increment of the *major* version number indicates that the release contains major new features and changes compared to the previous major version.

2. The minor number



An increment of the *minor* version number indicates that the release contains minor new features and changes compared to the previous minor version.

3. The patch number

An increment of the *patch* version number indicates that the release fixes known problems.

The numbers are separated by dots in the following naming scheme: <major>. <minor>. <patch>.

Examples for spelled out product version numbers are:

- ▶ EB tresos AutoCore OS 4.0
- ▶ EB tresos AutoCore OS 4.1
- ▶ EB tresos AutoCore Generic 4.3
- ▶ EB tresos AutoCore Generic 4.4

You can find the product version number for example on the documentation cover and the release notes cover.

Within GUIs, e.g. in EB tresos Studio, you can find the product version number on the splash screen and the **EB tresos details** dialog next to the keyword *Studio*.

There may be a *qualifier* additionally to the product version number. This *qualifier* is provided in brackets and displayed after the product version number.

1.3.2. Qualifiers for basic software

Basic software of the EB tresos product line are the products: EB tresos AutoCore Generic and EB tresos AutoCore OS.

The qualifier for basic software is provided in the quality statement (Q statement), which is part of each delivery. In the quality statement the application area is documented based on the quality level of the delivery. The different quality levels and their associated quality criteria are documented in the EB tresos AutoCore Generic Quality Level documentation.

The Quality Level documentation can be found in \$TRESOS_BASE/doc/4.0_EB_tresos_AutoCore_Generic/AutoCore_Generic_Quality_Level_documentation.pdf where \$TRESOS_BASE refers to the location of your EB tresos Studio installation.

The quality statement is provided in parallel to the installation file of the basic software and on the command server (EB Command) in the naming scheme: EB_tresos_AutoCore-quality_statement-<RelName>-<Target>-B<BuildNr>(_<Suffix>).doc, e.g. EB_tresos_AutoCore-quality_statement-ACG-6.4-WIN32X86-B64208.doc



1.4. Typography and style conventions

Throughout the documentation you see that words and phrases are displayed in bold or italic font, or in Mono-space font. To find out what these conventions mean, consult the following table. All default text is written in Arial Regular font without any markup.

Convention	Item is used	Example
Arial italics	to define new terms	The <i>basic building blocks</i> of a configuration are module configurations.
Arial italics	to emphasize	If your project's release version is mixed, all content types are available. It is thus called <i>mixed version</i> .
Arial italics	to indicate that a term is explained in the glossary	...exchanges <i>protocol data units (PDUs)</i> with its peer instance of other ECUs.
Arial boldface	for menus and submenus	Choose the Options menu.
Arial boldface	for buttons	Select OK .
Arial boldface	for keyboard keys	Press the Enter key
Arial boldface	for keyboard combination of keys	Press Ctrl+Alt+Delete
Arial boldface	for commands	Convert the XDM file to the newer version by using the legacy convert command.
Monospace font (Courier)	for file and folder names, also for chapter names	Put your script in the <code>function_name/abc-folder</code>
Monospace font (Courier)	for code	<code>for (i=0; i<5; i++) { /* now use i */ }</code>
Monospace font (Courier)	for function names, methods, or routines	The <code>cos</code> function finds the cosine of each array element. Syntax line example is <code>MLGetVar ML_var_name</code>
Monospace font (Courier)	for user input/indicates variable text	Enter a three-digit prefix in the menu line.
Square brackets []	for optional parameters; for command syntax with optional parameters	<code>insertBefore [<opt>]</code>
Curly brackets {}	for mandatory parameters; for command syntax with mandatory parameters (in curly brackets)	<code>insertBefore {<file>}</code>
Three dots ...	for further parameters	<code>insertBefore [<opt>...]</code>
A vertical bar	to separate parameters in a list from which one parameters must be cho-	<code>allowinvalidmarkup {on off}</code>



Convention	Item is used	Example	
	seen or used; for command syntax, indicates a choice of parameters		
Warning	to show information vital for the success of your configuration	WARNING 	This is a warning This is what a warning looks like.
Notice	to give additional important information on the subject	NOTE 	This is a notice This is what a notice looks like.
Tip	to provide helpful hints and tips	TIP 	This is a tip This is what a tip looks like.

1.5. Naming conventions

The naming conventions listed below will enable you to understand meanings of words and word groups that may otherwise be confusing without explanation. This chapter is useful as a reference point if you are unsure what a specific spelling (e.g. capital letters in parameters, a word in boldface that appears on a screen) means. Browse through to see the conventions and return if you need an explanation for a phenomenon you do not understand.

1.5.1. AUTOSAR XML schema

Parameters spelled in capital letters (e.g. RECOMMENDED-CONFIGURATION-REF and BSW-MODULE-DESCRIPTION) refer to the AUTOSAR XML schema.

XML tags are presented as specified in AUTOSAR Model Persistence Rules for XML V2.1.2 R3.-0 Rev 0001, specifically in chapter 3.6 XML names [7]:

- ▶ All XML elements, XML attributes, XML groups and XML types used in the AUTOSAR XML schema are written in upper case letters.
- ▶ In order to increase the readability of the XML names, hyphens are inserted in the XML names which separate the parts of the names.



1.5.2. Configuration parameter names

All configuration parameter names used in EB tresos Studio could be used for code generation and therefore potentially conflict with names used in the EB tresos AutoCore.

WARNING To avoid naming conflicts, do not use any of the following names as configuration parameter name:



- ▶ any C keywords
- ▶ name of AUTOSAR modules, e.g. Rte, Os, Com.

1.6. EB tresos Studio naming conventions

1.6.1. EB tresos Studio command shell

System commands and properties in the EB tresos Studio command shell are spelled with mixed lower- and upper-case letters, e.g. -DmergeConfigs.

Where the AUTOSAR XML schema uses hyphens (–) to separate parameter parts, EB tresos Studio uses the camelback case to separate different parts of the parameters.



Example 1.1. Example instructions that use the AUTOSAR XML schema and the EB tresos Studio spelling convention

Define the SHORT-NAME of the parameter ECU-PARAMETER. Example: –
DecuParamDefName=<name>



2. Safe and correct use of EB tresos Studio

2.1. Intended usage of EB tresos Studio

EB tresos Studio is intended to be used for configuring and generating EB tresos OsekCore and ECU basic software compliant to the AUTOSAR standard (see www.autosar.org).

2.2. Possible misuse of EB tresos Studio

- ▶ If you use the product in non-automotive projects or in automotive projects that are not based on OSEK or AUTOSAR technology (see www.autosar.org), the product and its technology may not conform to the requirements of your application. Elektrobit Automotive GmbH is not liable for such misuse.

2.3. Target group and required knowledge

- ▶ Automotive software engineers
- ▶ Programming skills and experience in programming AUTOSAR- and OSEK/VDX-compliant ECUs

2.4. Quality standards compliance

EB tresos Studio has been developed following processes that have been assessed and awarded ISO 9001:2008. These processes are based on Automotive SPICE. Should it be necessary to use any part of EB tresos Studio for a safety relevant project, contact EB first.



3. Background information

3.1. Overview

The chapter **Background information** is for first-time users of EB tresos Studio and those, who would like

- ▶ to update their knowledge of the basic concepts of AUTOSAR and
- ▶ to find out some of the advantages EB tresos Studio has to offer.

In each of the following chapters you will also find sections called **Background information**. These provide a conceptual context for the instructions and references that follow in later sections.

While **Background information** chapters may help advanced users of EB tresos Studio to figure out why some things are the way they are, they are primarily aimed at newcomers to AUTOSAR and EB tresos Studio and those, not very familiar with the graphical user interface.

3.2. The AUTOSAR concept

AUTOSAR (*Automotive Open System Architecture*) is a partnership of

- ▶ automotive,
- ▶ electronics,
- ▶ semiconductor,
- ▶ hard- and software

companies that work toward an open industry standard for automotive embedded architectures.

Software adhering to this industry standard allows you

- ▶ to manage the complicated interaction within automotive Electrical/Electronic (E/E) architectures,
- ▶ to make product upgrades and updates easier and more flexible,
- ▶ to re-use software within a product line and across product lines,
- ▶ to improve E/E systems' quality and reliability,
- ▶ and to find software errors in the early phases of design. Electronic control units (ECU) are the platforms on which the functions of the application (AUTOSAR application software components) are executed.



AUTOSAR functionally separates an ECU's application from the ECU's basic functionalities. An ECU's basic functionalities are

- ▶ the operating system,
- ▶ communication systems,
- ▶ and access to peripherals.

These basic functionalities are implemented in what AUTOSAR calls Basic Software (BSW). All data exchange between application components and the BSW are routed via an abstraction layer. This abstraction layer is the Run-Time Environment (Rte), which hides the BSW layers from the application components. The Rte implements the Virtual Function Bus (VFB), which is the abstract communication layer via which the application components actually exchange signals/data. Thus application components exchange data without knowing the path this data is going to take within or between ECUs.

The BSW includes drivers to access peripherals and communication, and the operating system. Application engineers no longer need to or indeed may access the BSW's resources directly but route their call via defined interfaces in the Rte. In theory this also enables to move functions from one ECU to another. In practice this is difficult because of signal transmission times and delays.

3.3. What is EB tresos Studio?

3.3.1. The configuration and generation tool

Since AUTOSAR relies on generating code according to user requirements, a configuration tool such as EB tresos Studio is needed to configure standard software components and to generate specific ECUs.

The tool follows the AUTOSAR methodology, but can be used for much more than just basic software modules of this standard.

The open concept of its code generator engines, parameter verification language, data model access and GUI allows you to seamlessly integrate EB tresos Studio into existing tool-chains and workflows.

It helps you to deploy standard software components within your organization in an efficient and easy-to-use manner.

The main concepts of EB tresos Studio are to follow the AUTOSAR methodology, provide open and documented interfaces, help to avoid user-errors and create an easy-to-use tool environment.

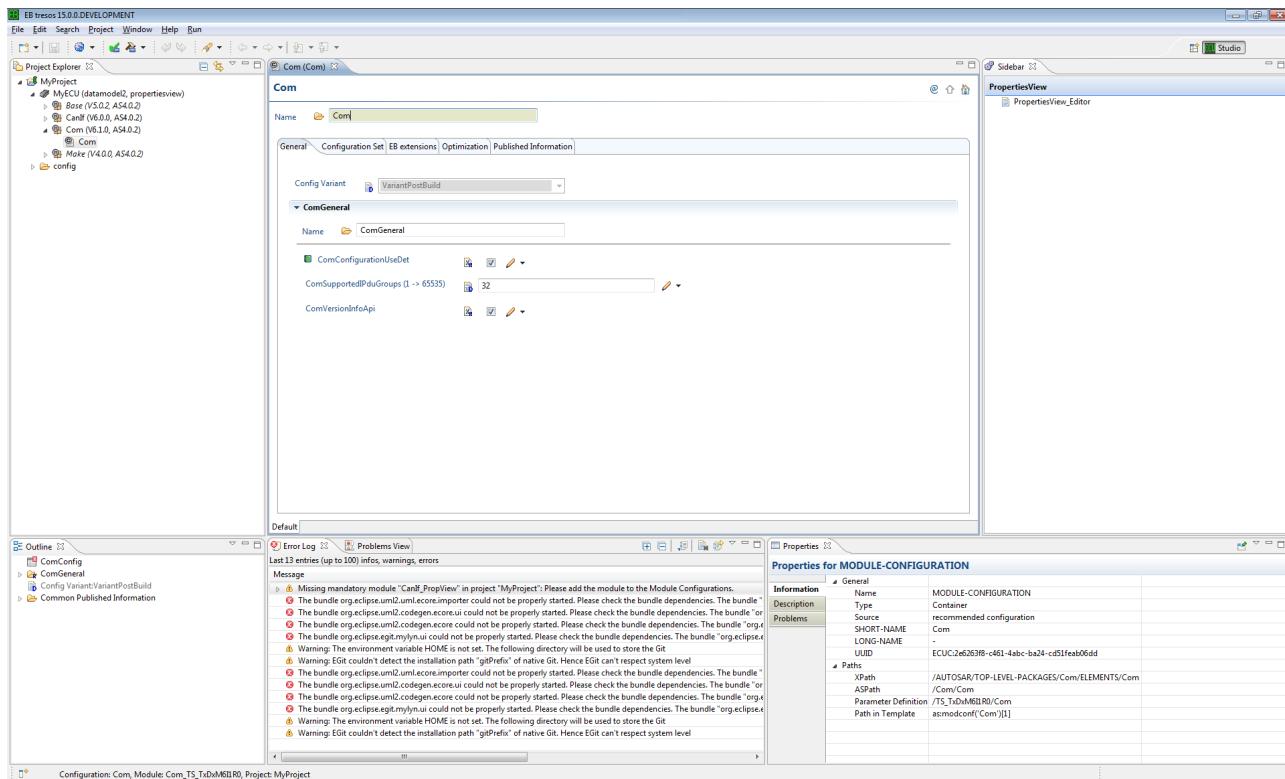


Figure 3.1. The main window of EB tresos Studio

3.3.2. Usage

Typical use scenarios for EB tresos Studio are:

- ▶ OEMs use EB tresos Studio for pre-integration of ECU standard-core software and deployment among their suppliers.
- ▶ Tier-one companies use EB tresos Studio for specification and code generation of basic software for their production ECU projects.
- ▶ Semiconductor vendors have chosen EB tresos Studio as the tool for the deployment of microcontroller drivers among their customers. Of course, EB uses EB tresos Studio as the configuration editor for all their EB tresos AutoCore products.

3.3.3. Benefits

You can immediately experience the benefits of EB tresos Studio's AUTOSAR methodology, when you work with the software tool:



- ▶ User-friendly configuration of basic software modules
- ▶ Competent implementation of the AUTOSAR methodology
- ▶ Quick familiarization
- ▶ Open and documented interfaces
- ▶ Extendibility and interoperability

3.3.4. Features

EB tresos Studio offers a wealth of features, which simplify your job of configuring software modules for an *ECU*.

Furthermore, the tool offers interfaces and methods for tool-chain integration and customization:

- ▶ Generic configuration editor according to AUTOSAR with many add-on features and improvements.
- ▶ Easy navigation within the ECU configuration data structure via
 - ▶ tabbed editors,
 - ▶ node views,
 - ▶ and clickable parameter references.
- ▶ Eclipse-based graphical user-interface, customizable via Java plug-ins.
- ▶ Validation of configuration parameters and immediate user feedback of detected issues.
- ▶ Importers for legacy formats such as
 - ▶ *FIBEX*,
 - ▶ *DBC*,
 - ▶ and *LDF*
- ▶ Integration of company-specific software modules due to an open Java API and code generator engines
- ▶ Interoperability with system architecture tools such as dSPACE SystemDesk.

In detail, these features include:

Eclipse based framework

Using Eclipse 3.x as the foundation technology ensures major extensibility and support for a fast growing set of software development related tooling.

Configuration capabilities

EB tresos Studio provides support for configuration editors that allow to edit the configuration of an embedded software module.

Generic configurator

EB tresos Studio provides a generic configuration editor that uses the definition of configuration parameters and automatically displays a dialog-based graphical user interface (GUI).



Custom configurators

EB tresos Studio can handle multiple editors per module. Editors can be specifically implemented in Java for a specific module.

Code generator capabilities

EB tresos Studio provides support for code generators that allow to transform a configuration into source code that can be compiled for an embedded target platform.

Commandline interface

EB tresos Studio provides a commandline interface for code-generation to run in batch process.

Multiple generators

Each module may provide its own or even multiple generators.

Template-based generator

EB tresos Studio provides a code-template based generic generator engine.

Project-based configuration

EB tresos Studio is capable of managing multiple configuration projects where each project may represent the configuration for a whole ECU.

Multiple modules

A project is capable to host multiple modules that share a common configuration database.

Multiple configurations

A module can be configured multiple times inside one project.¹

Cross module dependencies

Cross module-dependencies within the configuration of one ECU can be expressed via XPath-expressions and are evaluated constantly during configuration.

GUI hints

A parameter definition of a module can contain GUI hints such as an enable/disable expression, a HTML-description or layout hints to allow the generic configurator provide a more useable interface.

List in the configuration

Lists of elements in the configuration are shown as a table with multiple sortable columns.

Navigatable references

From references in the configuration that point to loaded parameters the GUI can be navigated easily to the referenced element.

Importer/Exporter

EB tresos Studio provides a generic importer/exporter interface that allows to add importers and exporters for foreign formats such as OIL, DBC, DIL, Fibex and annotates imported data so that a re-import is possible.

¹Not every module may support this option.

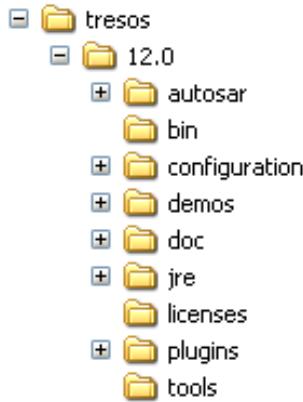


ECU Configuration

EB tresos Studio allows to configure, import and export ECU Configuration modules compatible to the AUTOSAR release versions 2.0, 2.1, 3.0, 3.1, 3.2, 4.0.2, 4.0.3, 4.1.3, 4.2.1, 4.2.2, 4.3.0, 4.3.1 and 4.4.0

3.4. File and directory structure

A new EB tresos Studio installation has the following directory structure:



Directory	Description
autosar	contains the AUTOSAR ECU configuration parameter definition for each supported AUTOSAR version and a Perl script to generate EPD files from each parameter definition
bin	contains the executable files to start EB tresos Studio (<code>tresos_gui.exe</code> and <code>tresos_cmd.bat</code>)
configuration	run-time configuration information for EB tresos Studio, for example the plug-in cache
demos	contains demo plug-ins for developers; the demos showcase the EB tresos Studio API
doc	program documentation, e.g. the EB tresos Studio user's guide
jre	Java run-time environment ; this is required to run EB tresos Studio
licenses	license agreements of third-party products
plugin	plug-ins and modules, which are part of the EB tresos Studio installation
tools	contains tools for developers that use the EB tresos Studio API

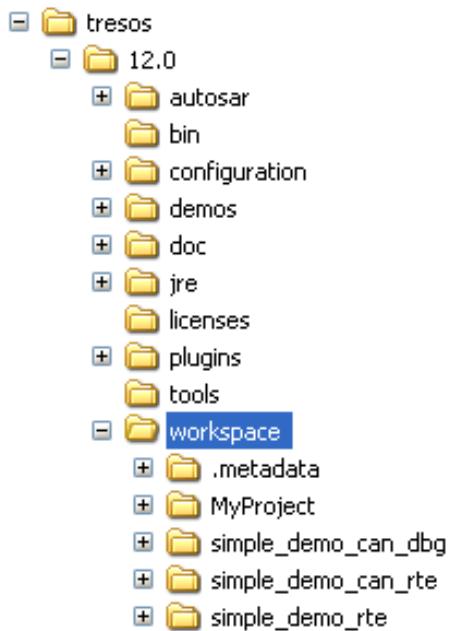
3.5. EB tresos Studio projects and workspaces



The EB tresos Studio tool enables you to work with multiple projects rather than a single configuration. Projects are arranged in a workspace.

3.5.1. The Workspace

A workspace is a folder by default in the installation tree of your EB tresos Studio installation. If the workspace folder is not present, it is automatically created upon the first start of the program. It is possible to have several workspaces managing different projects (see [Section 6.2.3.1, “Switching the workspace”](#)).



Besides projects the workspace stores non project specific preferences and also log files in a meta data area.

EB tresos Studio always operates on one workspace which is initially selected on tool startup (see [Section 6.2.3.1, “Switching the workspace”](#)). The workspace can be switched via the **File** menu, EB tresos Studio restarts in this case with the new workspace open.

3.5.2. Projects

ECU configurations, system descriptions and also companion files like the source code of an application can be stored in a project. A project is represented as a folder with some hidden files describing the project meta data. The project folder normally resides beneath the workspace folder but can also be placed somewhere else in the file system. Projects can be imported into a workspace after which they show up in the EB tresos Studio GUI.

An EB tresos Studio project can host an ECU configuration consisting of module configurations in the form of XML files or a system description model in the form of a database or both. ECU configurations and system



descriptions are stored in a folder `config` beneath the top level folder of the project. Arbitrary files can be created or imported into a project folder via the GUI of EB tresos Studio

3.5.2.1. Configuration time of projects

The *configuration time* is the phase or the process step of your project during the development of AUTOSAR basic software (BSW) modules. It defines which parts of a module must be configured in which development phase. AUTOSAR defines the following four configuration classes:

Configuration class	Description
PublishedInformation	Specifies which information is fixed before the precompile stage.
PreCompile	Specifies that the module is configured before compiling the source code.
Link	Specifies that the module is configured during link time. This means the object code of the module receives parts of its configuration from another object code file.
PostBuild	Specifies that the module gets the parameters of its configuration either by downloading a separate file to the ECU memory (<code>POST-BUILD-LOADABLE</code>) or by defining multiple configuration sets and choosing one during boot-time (<code>POST-BUILD-SELECTABLE</code>).

For more detailed information on AUTOSAR configuration classes, see [4], chapter 2.3.2 Introduction To Configuration Classes.

The order of the configuration times in the configuration process is: PublishedInformation -> PreCompile -> Link -> PostBuild.

You may define the configuration time of your project. For instructions on defining the configuration time of your project, see [Section 6.3.7.3, “Editing the ECU Configuration settings”](#). The configuration time of the project defines which parameters and references of the module configuration may be edited in the following way: Each parameter and each reference of an ECU configuration belongs to exactly one configuration class. The configuration class of a parameter or a reference depends on the configuration variant of the module definition: Each module defines the configuration variants it supports. While configuring a module, you must select one of the configuration variants of the module. Instructions for setting the configuration variant is available at [Section 6.8.4, “Setting the configuration variant of the module”](#). The selected configuration variant is then called *implementation configuration variant*. If the implementation configuration variant of a module is set, the configuration class of all parameters and references of the module are uniquely determined.

The procedure is as follows:

- ▶ Select the configuration variant of your modules (see [Section 6.8.4, “Setting the configuration variant of the module”](#)) to set the configuration class of the parameters and references.



- ▶ Select the configuration time of your project (see [Section 6.3.7.3, “Editing the ECU Configuration settings”](#)) to limit the access to the parameters and references of your configuration process step.

3.5.3. File content types

Content types define the content of files more precisely than the file extension. This influences how the information read from files is interpreted or how the information is saved to the files. In EB tresos Studio, content types are used in two locations: when using the AUTOSAR importer or exporter (see [Section 6.10.2.2, “Importing and exporting configuration data from and to the AUTOSAR format”](#)) and when using the command line (see [Section 6.12, “Working with the command line”](#)).

The content type of a file defines how a file is validated, e.g. against an XSD schema. The content type also determines how new files are written. Every input and output file for each command on the command line may be equipped with a content type.

By default, EB tresos Studio knows the following content types:

- ▶ Files containing module configuration data:
 - ▶ `xdm` for the XDM file format
 - ▶ `asc` for AUTOSAR module configuration files
- ▶ Files containing system description data:
 - ▶ `sysd` for AUTOSAR system description files
 - ▶ `tdb` for tresosDB system description files
 - ▶ `dbc` for the DBC file format
 - ▶ `ldf` for the LDF file format
 - ▶ `fibex` for Fibex files

In addition, parsers contained in other plugins or modules that you may have installed may define their own content types.

3.5.3.1. Content types for XDM files

Different versions of EB tresos Studio may use different versions of the XDM file format. Newer versions of EB tresos Studio are always able to read the older file versions. The following content types are available:

Content type	Description
<code>xdm</code>	refers to the latest version of the XDM file format (currently 4.0)
<code>xdm:2.0</code>	refers to an older version of the XDM format which was used in versions prior to and including EB tresos Studio 2007.b



Content type	Description
xdm:3.0	refers to the version of the XDM format that has been used since EB tresos Studio 2008.a.
xdm:4.0	refers to the version of the XDM format that has been used since EB tresos Studio 19.0

3.5.3.2. Content types for AUTOSAR module configuration files

EB tresos Studio enables importing or exporting project configuration data from and to the AUTOSAR format. To do so, you need to be aware of the AUTOSAR content type of the file to be imported or exported. The term content type describes the fact that the AUTOSAR specification has been released in several versions already. These content type versions are validated against different XSD schemas and their characteristics differ from each other.

The table below lists all AUTOSAR formats supported. To exchange files between different tools, use a format that can be checked by both tools when they use the same AUTOSAR XSD file.

Content type	Description
asc	Default content type.
asc:2.0	Imports/Exports files using the AUTOSAR XSD version 2.0.
asc:spec2.0	Imports/Exports files using the AUTOSAR XSD version 2.0. Writes files as specified in [3].
	Differences to asc:2.0: <ul style="list-style-type: none"> ▶ Empty values and non-configured values are allowed. This also applies to integer values. This means that empty value and value-ref tags may occur, e.g. <value></value>! However, <value/> may not occur. ▶ For configuration classes, the values are stored/accepted as defined within the specification: PreCompile, Link, PostBuild, PostBuildLoadable, and PostBuildSelectable. ▶ The order of XML elements is not specified.
asc:2.1	Imports/Exports files using the AUTOSAR XSD version 2.1. This format enables exchanging files between tools if the tools use the same AUTOSAR XSD file format and can check the files' validity.
asc:3.0	Imports/Exports files using the AUTOSAR XSD version 3.0.
asc:3.1	Imports/Exports files using the AUTOSAR XSD version 3.1.



Content type	Description
asc:3.2	Imports/Exports files using the AUTOSAR XSD version 3.2.
asc:4.0.2	Imports/Exports files using the AUTOSAR XSD version 4.0.2.
asc:4.0.3	Imports/Exports files using the AUTOSAR XSD version 4.0.3.
asc:4.1.3	Imports/Exports files using the AUTOSAR XSD version 4.1.3.
asc:4.2.1	Imports/Exports files using the AUTOSAR XSD version 4.2.1.
asc:4.2.2	Imports/Exports files using the AUTOSAR XSD version 4.2.2.
asc:4.3.0	Imports/Exports files using the AUTOSAR XSD version 4.3.0.
asc:4.3.1	Imports/Exports files using the AUTOSAR XSD version 4.3.1.
asc:4.4.0	Imports/Exports files using the AUTOSAR XSD version 4.4.0.

3.5.3.3. Content types for AUTOSAR system description files

EB tresos Studio can import system description data from AUTOSAR system description files and convert them into a tresosDB file. For details about the file format, see [Section 3.5.3.4, “Content types for tresosDB files”](#).

The AUTOSAR specification has been released in several versions already. EB tresos Studio support the following ones:

Content type	Description
sysd	Default content type, EB tresos Studio will try to automatically detect the right version
sysd:2.1.2	Imports files using the AUTOSAR XSD version 2.1 (XSD Document Version 2.1.3 Revision 0017, xmlns="http://autosar.org/2.1.2")
sysd:2.1.4	Imports files using the AUTOSAR XSD version 2.1 (XSD Document Version 2.1.4 Revision 0018, xmlns="http://autosar.org/2.1.4")
sysd:3.0.2	Imports files using the AUTOSAR XSD version 3.0 (XSD Document Version 3.0.2 Revision 0003, xmlns="http://autosar.org/3.0.2")
sysd:3.1.0	Imports files using the AUTOSAR XSD version 3.1 (XSD Document Version 3.1.0 Revision 0001, xmlns="http://autosar.org/3.1.0")
sysd:3.1.2	Imports files using the AUTOSAR XSD version 3.1 (XSD Document Version 3.2.1 Revision 0003, xmlns="http://autosar.org/3.1.2")
sysd:3.1.4	Imports files using the AUTOSAR XSD version 3.1 (XSD Document Version 3.3.0 Revision 0004, xmlns="http://autosar.org/3.1.4")
sysd:3.1.4.DAI.2	Imports files using the AUTOSAR XSD version 3.1 (XSD Document Version 3.3.0 Revision 0004 (Daimler specific version 2), xmlns="http://autosar.org/3.1.4.DAI.2")



Content type	Description
sysd:3.1.4.DAI.3	Imports files using the AUTOSAR XSD version 3.1 (XSD Document Version 3.3.0 Revision 0004 (Daimler specific version 3), xmlns="http://autosar.org/3.1.4.DAI.3")
sysd:3.1.4.DAI.4	Imports files using the AUTOSAR XSD version 3.1 (XSD Document Version 3.3.0 Revision 0004 (Daimler specific version 4), xmlns="http://autosar.org/3.1.4.DAI.4")
sysd:3.2.1	Imports files using the AUTOSAR XSD version 3.2 (XSD Document Version 3.5.0 Revision 0001, xmlns="http://autosar.org/3.2.1")
sysd:4.0.2	Imports files using the AUTOSAR XSD version 4.0.2 (XSD Document Version 4.1.0 Revision 0002, xmlns="http://autosar.org/r4.0")
sysd:4.0.3	Imports files using the AUTOSAR XSD version 4.0.3 (XSD Document Version 4.2.1 Revision 0003, xmlns="http://autosar.org/r4.0")
sysd:4.1.0	Imports files using the AUTOSAR XSD version 4.1.0 (snapshot 2012-08-09, XSD Document Version 4.3.0 Revision 0000, xmlns="http://autosar.org/r4.0")
sysd:4.1.1	Imports files using the AUTOSAR XSD version 4.1.1 (XSD Document Version 4.3.0 Revision 0001, xmlns="http://autosar.org/r4.0")
sysd:4.1.2	Imports files using the AUTOSAR XSD version 4.1.2 (XSD Document Version 4.4.0 Revision 0002, xmlns="http://autosar.org/r4.0")
sysd:4.1.3	Imports files using the AUTOSAR XSD version 4.1.3 (XSD Document Version 4.5.0 Revision 0003, xmlns="http://autosar.org/r4.0")
sysd:4.2.1	Imports files using the AUTOSAR XSD version 4.2.1 (XSD Document Version 4.6.0 Revision 0001, xmlns="http://autosar.org/r4.0")
sysd:4.2.2	Imports files using the AUTOSAR XSD version 4.2.2 (XSD Document Version 4.6.0 Revision 0002, xmlns="http://autosar.org/r4.0")
sysd:4.3.0	Imports files using the AUTOSAR XSD version 4.3.0 (xmlns="http://autosar.org/r4.0")
sysd:4.3.1	Imports files using the AUTOSAR XSD version 4.3.1 (xmlns="http://autosar.org/r4.0")
sysd:4.4.0	Imports files using the AUTOSAR XSD version 4.4.0 (xmlns="http://autosar.org/r4.0")

3.5.3.4. Content types for tresosDB files

The tresosDB format is the internal storage format for system description data. The tresosDB format is also used to exchange system description data with other tools.



There is no need to specify a version number when importing or exporting tresosDB files. During import, the version of the file is automatically detected and the data is converted to the most current version. Thus it is only possible to export a tresosDB file of the most current version.

Content type	Description
tdb	Default content type for a tresosDB file (use auto-detection for version).
tdb:3.1	Content type for a tresosDB file that represents AUTOSAR 3.1.
tdb:4.0	Content type for a tresosDB file that represents AUTOSAR 4.0.
tdb:4.1	Content type for a tresosDB file that represents AUTOSAR 4.1.
tdb:4.3	Content type for a tresosDB file that represents AUTOSAR 4.3.
tdb:4.4	Content type for a tresosDB file that represents AUTOSAR 4.4.

3.5.3.5. Content types for DBC files

DBC files can be imported by EB tresos Studio and converted into a tresosDB ([Section 3.5.3.4, “Content types for tresosDB files”](#))

It is not necessary to distinguish different versions of CANdb files, thus it is not necessary to provide a version number.

Content type	Description
dbc	DBC (CANdb) file

3.5.3.6. Content types for LDF files

LDF files can be imported by EB tresos Studio and converted into a tresosDB file ([Section 3.5.3.4, “Content types for tresosDB files”](#)).

While EB tresos Studio supports different versions of the LDF format (see [Section 6.10.2.7, “Importing from the LDF format”](#)), it is not necessary to specify one with the content type. The parser automatically detects which supported version is used in the files.

EB tresos Studio supports the following LDF file versions:

Content type	Description
ldf	LDF file, version is auto-detected
ldf:<version-string>	LDF file, <version-string> can be one of 2.0, 2.1, or 2.2. Providing the additional <version-string> string is allowed for backwards compatibility.



Content type	Description
	ibility reasons, but has no effect since the version of the LDF file is auto-detected.

3.5.3.7. Content types for Fibex files

Fibex files can be imported by EB tresos Studio and converted into a tresosDB file ([Section 3.5.3.4, “Content types for tresosDB files”](#)).

While EB tresos Studio supports different versions of the Fibex format (see [Section 6.10.2.8, “Importing from the Fibex format”](#)), it is not necessary to specify one with the content type. The parser automatically detects, which supported version is used in the files.

Content type	Description
fibex	Fibex file, version is auto-detected



4. Using EB tresos Studio for the first time

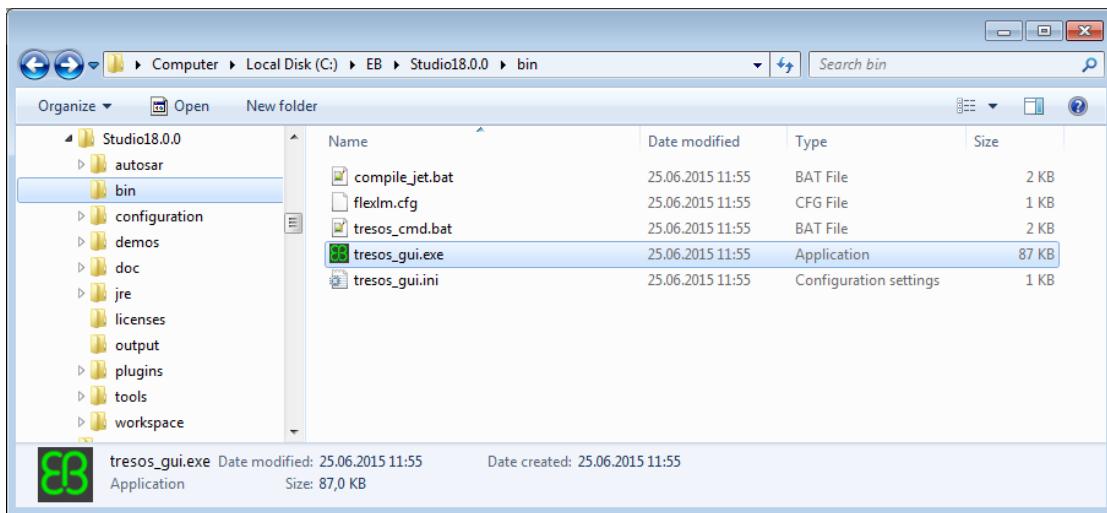
4.1. Starting the graphical user interface

By default, the EB tresos Studio installer creates an icon on your desktop and adds an entry in the Windows Start menu. Both are labeled **tresos Studio**. You can use either to start EB tresos Studio. The icon looks like this:



If you do not see such an icon:

- ▶ Locate the root directory of the EB tresos Studio installation \$TRESOS_BASE.



- ▶ In the \$TRESOS_BASE, locate bin\tresos_gui.exe.
- ▶ Double-click bin\tresos_gui.exe.

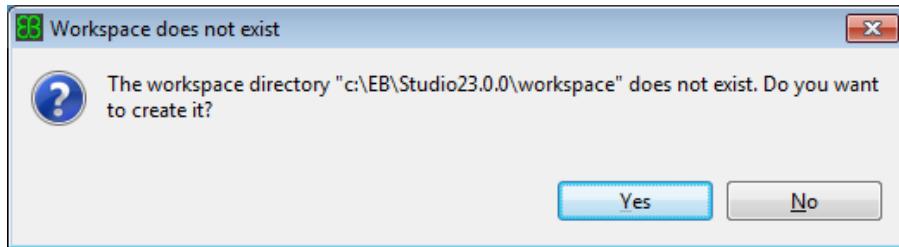
The GUI is being started.

4.2. Selecting the workspace location

The workspace is a directory where projects and settings are stored. If you start EB tresos Studio for the first time, you are asked whether you want to create the default workspace. The default workspace is the directory

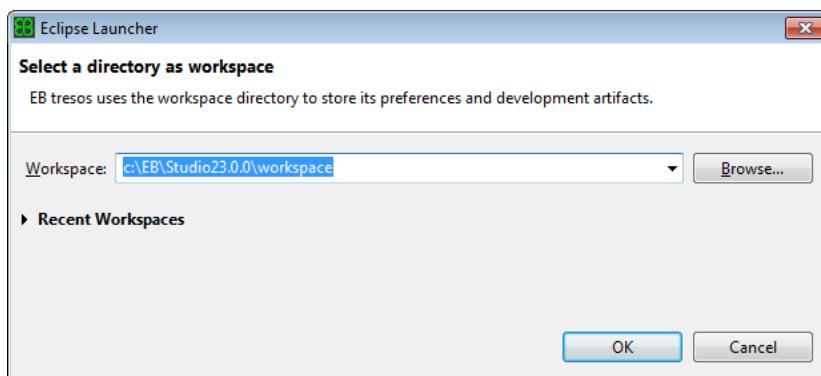


workspace within the `$TRESOS_BASE` directory. `$TRESOS_BASE` is the location at which you installed EB tresos Studio.



To select your workspace location:

- ▶ You can either accept the default workspace by pressing **Yes** or open a workspace selection dialog by pressing **No**.
 - ▶ If you press **Yes**, the default workspace is created at `$TRESOS_BASE/workspace`.
 - ▶ If you press **No**, the following dialog opens up:



- ▶ If you want to create your own workspace path, insert a path to a workspace of your choosing and press **OK**.

After pressing **OK**, the selected workspace is created and the EB tresos Studio main window appears.

- ▶ If you want to exit the startup process, press the **Cancel** button.



TIP



How to change your workspace at a later time

Once a workspace is created EB tresos Studio opens this workspace at startup. You can change the workspace afterwards with the **Switch Workspace** action under the **File** menu.

TIP

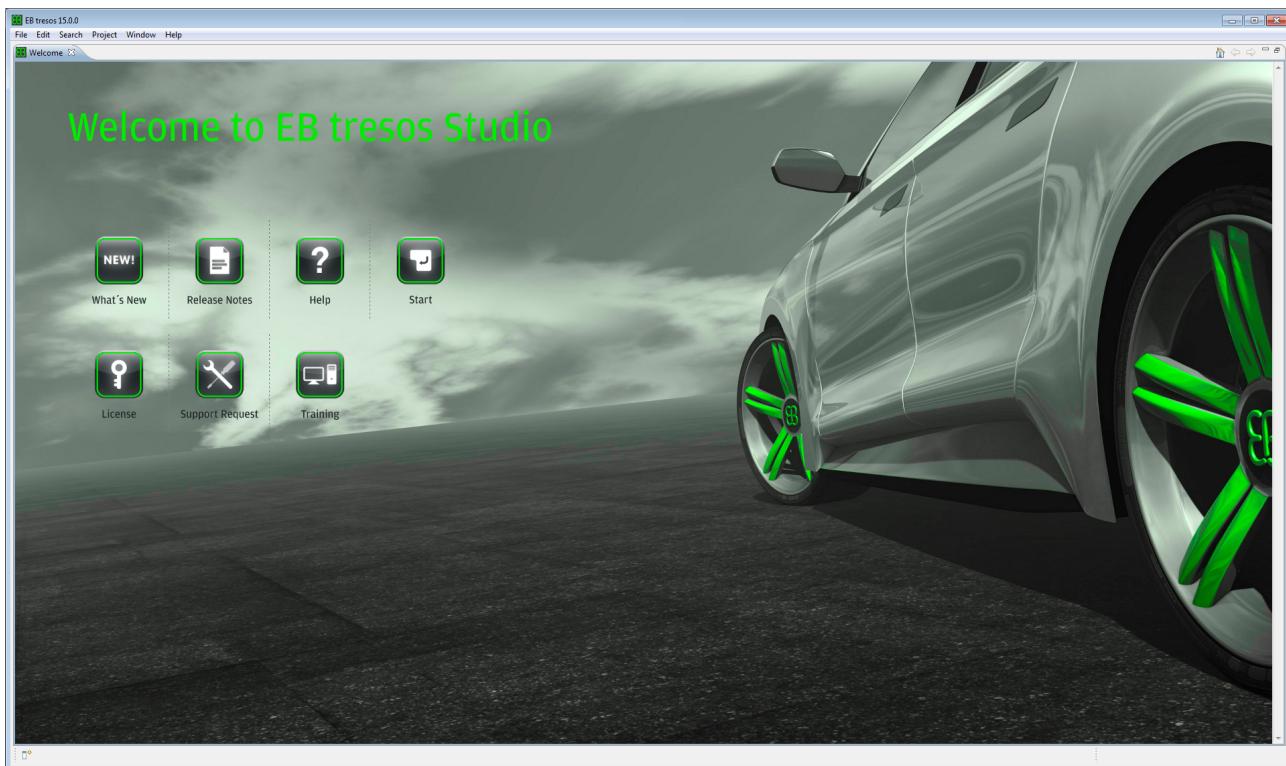


If your workspace does not exist or is being used

Any time the workspace does not exist or is already in use by another instance of EB tresos Studio you will be asked for creating or changing the workspace at startup.

4.3. The welcome page

The graphical user interface starts up with a **Welcome** view on the first launch:



The **Welcome** view displays some shortcuts to common functions that help you to familiarize yourself with EB tresos Studio.

- ▶ **What's New** - opens a document that contains information about the changes from last releases.
- ▶ **Release Notes** - shows information about fixed problems and new features.



- ▶ **Help** - opens the help dialog for EB tresos Studio.
- ▶ **Start** - closes the **Welcome** view, so you can start working with EB tresos Studio.
- ▶ **License** - opens the **Licenses** dialog.
- ▶ **Support Request** - links to the EB Support page.
- ▶ **Training** - links to the EB Automotive Technology Training Courses.

To start working with EB tresos Studio, click the **Start** button or close the **Welcome** view. To close the **Welcome** view, click the **Close** button next to the **Welcome** title.

If you want to open the view again, choose the **Welcome** item from the **Help** menu.

After closing the view, you see the EB tresos Studio main window. The contents of this window are explained in [Chapter 5, “The graphical user interface \(GUI\)“](#).

4.4. Starting the Linux Command Line

Execute the below command to extract the EB tresos Studio Linux UIP package

```
$ tar -xzvf EBtresosStudio_EBtresosStudio.uip
```

Once the UIP extracted, you can find the **tresos_cmd.sh** under **bin** directory, to start the EB tresos Studio Linux command line run the below command

```
$ sudo sh tresos_cmd.sh
```

Working with the command line explained in [Section 6.12, “Working with the command line”](#)



5. The graphical user interface (GUI)

This chapter is going to introduce you to the main window of EB tresos Studio, also called the *workbench*. You will get to know how the workbench is organized and what functions are available through the menus and the tool bars.

5.1. Using the Workbench

The graphical user interface (GUI) of EB tresos Studio consists of views that are arranged in the workbench, e.g. the **Project Explorer**. An arrangement of views is called a *perspective*. Perspectives are either pre-defined or user-defined.

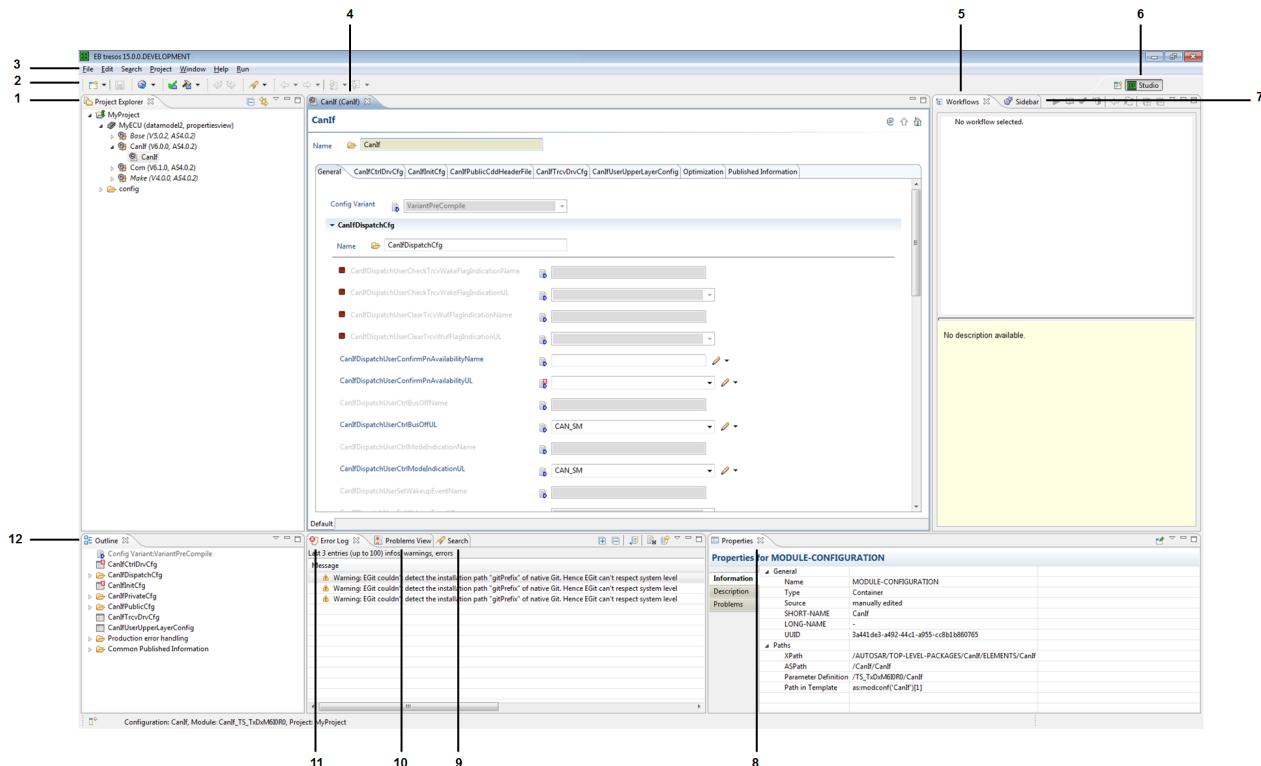


Figure 5.1. The EB tresos Studio Workbench

Number	Element	Description
1	Project Explorer	The Project Explorer displays all EB tresos Studio configuration projects in a workspace (see Section 3.5, “EB tresos Studio projects and workspaces”) and allows you to edit projects and open editors. The default location of the Project Explorer view is on the left of the screen.



Number	Element	Description
2	Tool bar	The tool bar is located in the upper left corner of the screen.
3	Menu bar	The menu bar is located in the upper left corner of the screen. In addition to the default menu items, there may additional menu items present. These are contributed by the additional plug-ins and modules you have installed and should be documented there.
4	Editor view	The Editor view is in the center of the workbench. It contains editors, with which you may edit the data of a project.
5	Workflows view	The Workflows view displays workflows which guide you to accomplish a certain task, e.g. to configure a Com stack. The default location is in the upper right corner of the screen.
6	Perspectives bar	The perspectives bar in the upper right corner of the screen is an element for handling workbench perspectives, i.e. opening, closing, switching, customizing, saving and resetting perspectives.
7	Sidebar view	The Sidebar view displays assistance dialogs for your current task. Per default the Sidebar view is located in the upper right corner of the screen, as a tab behind the Workflows view.
8	Properties view	The Properties view shows information about the selected parameter in the Generic Configuration Editor. This view contains three tabs: Information , Description and Problems . The default location of the Properties view is on the right bottom of the screen.
9	Search view	The Search view enables searching and finding files, specific configuration parameters, or text strings.
10	Problems View view	The Problems view displays the messages found in the configuration of a loaded project. The default location of the Problems view is on the bottom of the screen.
11	Error Log view	The Error Log view is the place in the GUI where all errors are logged. The Error Log receives error messages from EB tresos Studio components and from other Eclipse components. The list of messages in the Error Log is preserved in between the sessions of EB tresos Studio. The default location of the Error Log view is on the bottom of the screen.
12	Node Outline view	The Node Outline view on the left bottom of the workbench, which shows a tree view of the editable configuration parameters that you may edit in the Editor view.



5.1.1. Customizing the workbench

When you open EB tresos Studio for the first time, it opens up with a default layout for the views, namely the **Studio perspective** (see also [Section 5.1.2, “Using perspectives”](#)). The Studio perspective composes the views you need to edit ECU configuration projects. The layout is arranged for medium sized screens. You might want to change this layout so that it better suits your task. To do so, you may rearrange the perspective via drag and drop, stack the views as tabs over each other or maximize, minimize and close the views.

NOTE

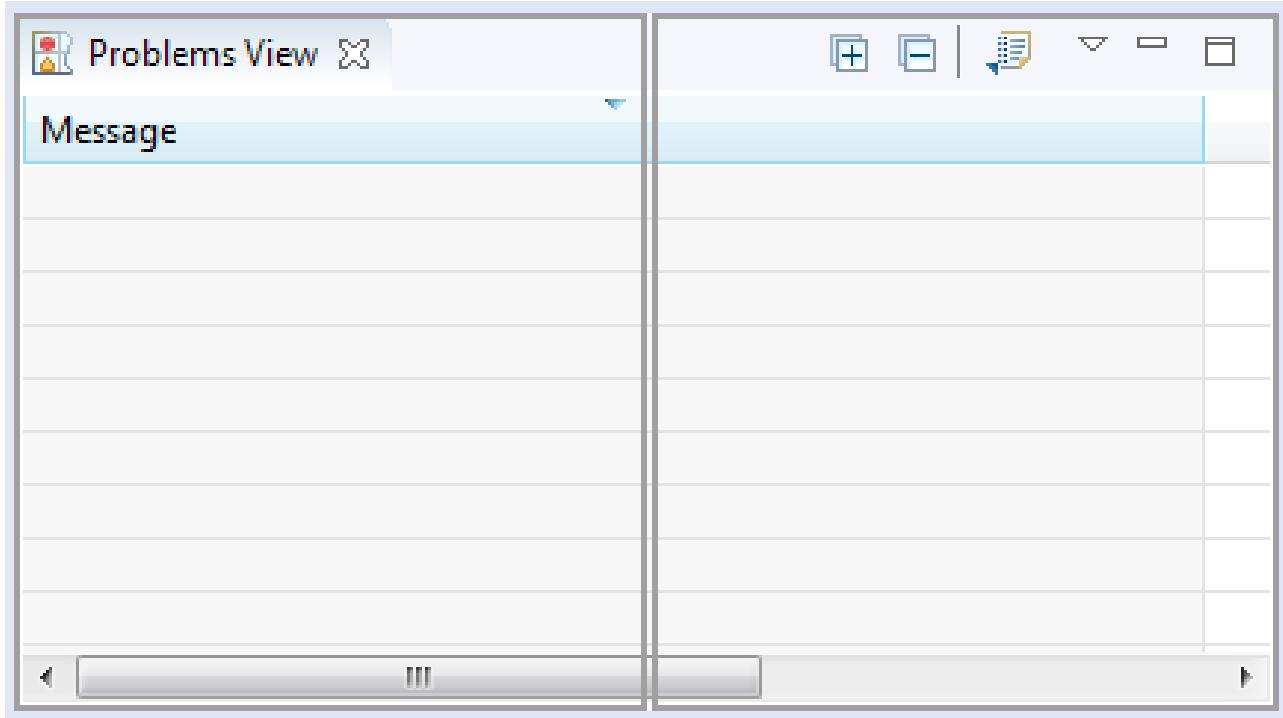


Not all views are visible

Some views are not visible by default, but they appear when they are needed. The **Search** view, for example, is usually not visible. Only if you have started a search, the view appears and displays the search results to you.

5.1.1.1. Moving views to a different location

Every view has a title area. If you click this area, keep the mouse button pressed and move around you can drag the view to a different location. The location to which the view will be placed when you release the mouse button, is indicated by a gray rectangle.



If you drop the view at a place where already another view is present, the views will be stacked. This means that there is always one view visible and the other view is behind it. The title areas of the views then work like tabs and you can move the other view to the foreground by clicking on its title area.



TIP



Moving a whole stack of views

In the default window layout, the **Problems** view (see [Section 5.3.6, “Problems view”](#)) and **Error Log** view (see [Section 5.3.5, “Error Log view”](#)) are examples for stacked views. You can also move a whole stack of views. To do this, drag the empty area of the title area and drop it at the location of your choice.

If you drop a view outside of the workbench, it is detached from the workbench and becomes a window of its own. To reverse this, drag it at its title area back into the workbench.

For editors, some extra rules apply:

- ▶ You cannot stack editors together with standard views and vice versa.
- ▶ Editors can only reside in the **Editor** view.
- ▶ You cannot detach editors from the workbench.

5.1.1.2. Closing views, and minimizing or maximizing view stacks



Number	Button	Description
1	Close	Closes the view so that it disappears completely. If you want to re-open the view again, continue reading at Section 5.1.1.3, “Opening a view” .
2	Minimize	Minimizes this view or view stack. Minimized views occupy a small space at the border of the window. In this place only the icons of the minimized view stack are shown and you can click them to restore the views again. This is useful if you temporarily want to hide a view, e. g. to have more space for other views.
3	Maximize	Maximizes the view. The view will occupy most of the workbench space. This is useful if you temporarily want to focus on one view only. E. g. sometimes you might want to maximize the Problems view in order to see the complete message text of all configuration problems.

To restore maximized or minimized views or view stacks, click on the **Restore** button . The **Restore** button is located either near the icons of the minimized view, or as a replacement of the **Maximize** button if the view has been maximized.



TIP



Double-click the title area of a view to restore or maximize it

To maximize a view, or to restore a maximized view, you can also double-click its title area.

5.1.1.3. Opening a view

Sometimes you might want to open a new view or a view you have closed before.

To open a view:

- ▶ In the **Window** menu, select **Show View**.
- ▶ A submenu opens up.

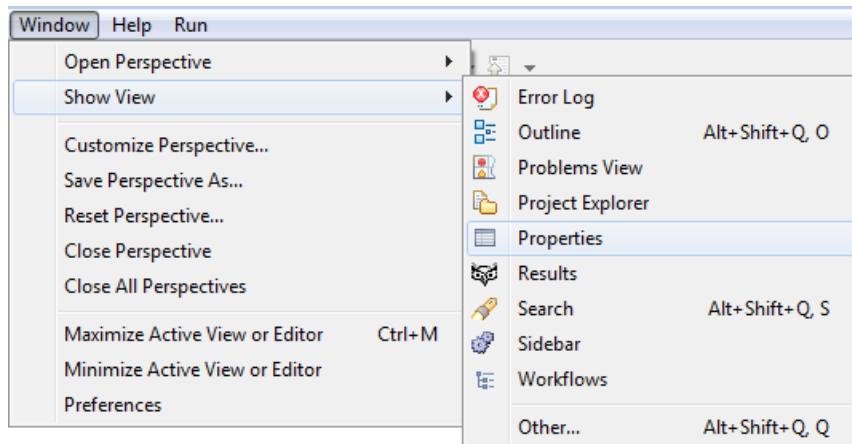


Figure 5.2. Opening a view that is not displayed in the workbench

- ▶ Select one of the views listed or select **Other...** to open the **Show View** dialog.

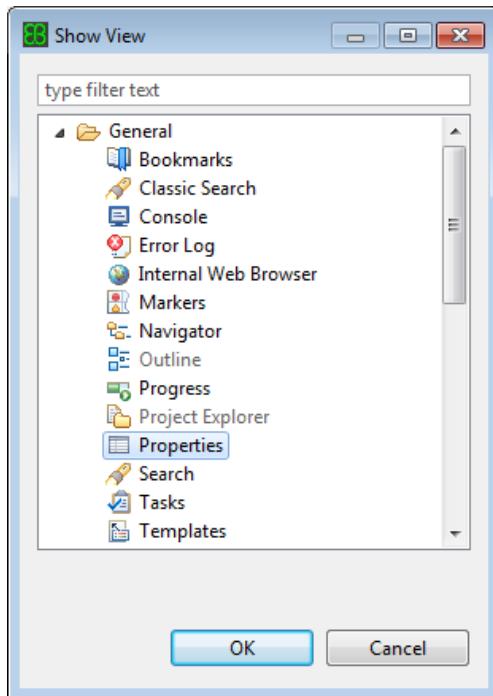


Figure 5.3. Selecting a view to open

- ▶ Select one or more views to open and click **OK**.

Note that views that are already open, are displayed in gray font in the **Show View** dialog.

- ▶ The selected views open at their last position.

5.1.2. Using perspectives

Tightly related to workbench customizing (see [Section 5.1.1, “Customizing the workbench”](#)) is the use of workbench perspectives. Perspectives are named arrangements of views and editors, together with menu and tool bar actions.

There are pre-defined perspectives, such as the Studio perspective, and user-defined perspectives, in which you may store your customized workbench layouts.



NOTE

Only one perspective can be active



While there can be more perspectives opened at a time, only one perspective can be active.

5.1.2.1. Saving user-defined perspectives

You can store custom layouts of your workbench into named perspectives.

To save the current perspective layout into your user-defined perspective:

- ▶ In the **Window** menu, select **Save Perspective As...**
- ▶ The **Save Perspective As...** dialog opens.

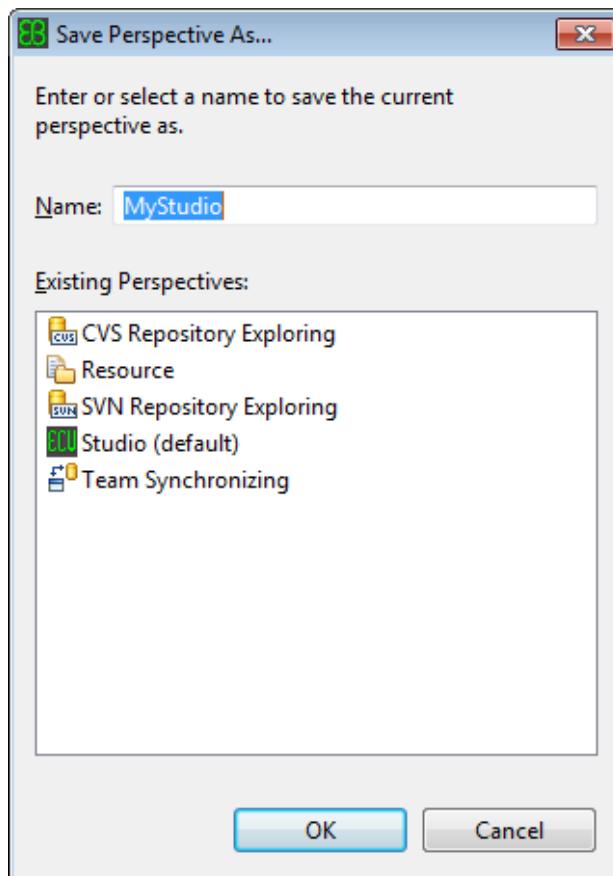


Figure 5.4. Saving your customized perspective

- ▶ Enter a name and click **OK**.
- ▶ Your perspective is saved.



-
- WARNING** **Do not store your perspective with the name of a pre-defined perspective**
-  It is technically feasible to save your customized perspective with the name of an existing pre-defined perspective. However, this is not recommended, as you then won't be able to restore the pre-defined perspective anymore.
-

5.1.2.2. Opening perspectives

To open a perspective:

- ▶ In the **Window** menu, select **Open Perspective** or click on the **Open Perspective** button  in the perspective bar.
- ▶ A context menu opens up.

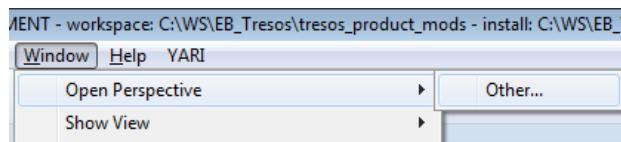


Figure 5.5. Opening a perspective

- ▶ Select one of the perspectives listed. If the desired perspective is not listed in the context menu, select **Other....**

The **Open Perspective** dialog opens up.

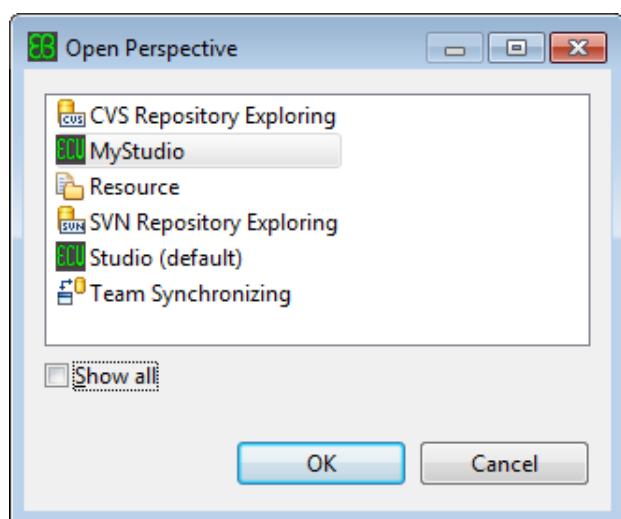


Figure 5.6. Selecting a perspective to open

- ▶ Select the perspective of your choice and click **OK**.
- ▶ The selected perspective opens up and is added to the perspective bar.



5.1.2.3. Switching between perspectives

To change the currently active perspective to another open one:

- ▶ Click the **Perspective** button you want to switch to in the perspective bar.

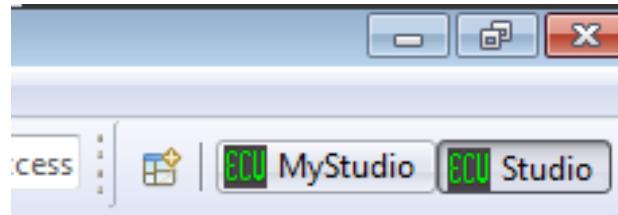


Figure 5.7. Switching perspectives

- ▶ Click the **Switch** button if more than two perspectives are available.

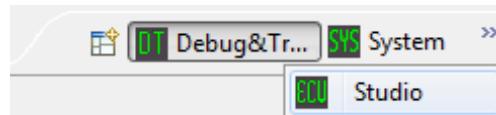


Figure 5.8. Switching more than two perspectives

- ▶ The currently active perspective is changed.

5.1.2.4. Closing perspectives

To close the current perspective:

- ▶ In the **Window** menu, select **Close Perspective**.
- ▶ The current perspective closes.

To close any open perspective:

- ▶ In the perspective bar, right-click on the icon of the perspective you want to close.

A context menu opens up.

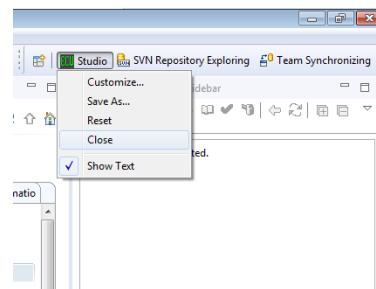


Figure 5.9. Closing perspectives

- ▶ The selected perspective closes.

If another perspective is open, it becomes activated. If you close the last open perspective, the workbench is emptied.

5.1.2.5. Resetting perspectives

You may reset a perspective to its default state, i.e. when you want to undo the last changes you made in the layout.

To reset the current perspective:

- ▶ In the **Window** menu, select **Reset Perspective** or right-click on the current perspective in the perspective bar.

A context menu opens up.

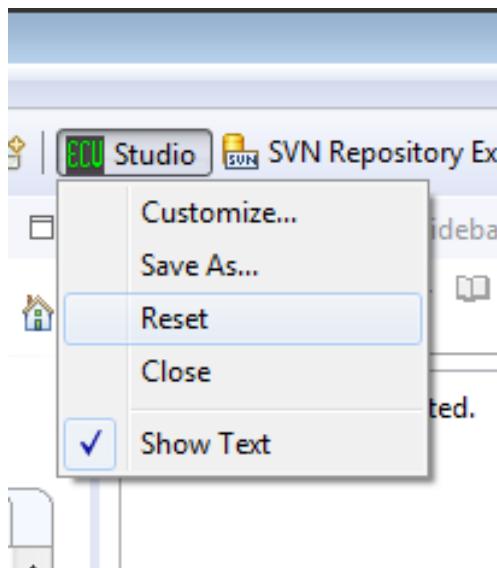


Figure 5.10. Resetting perspectives

- ▶ In the context menu, select **Reset**.
- ▶ The current perspective is reset, either to its pre-defined or to its last saved state.

5.1.2.6. Deleting perspectives

You may also delete your user-defined perspectives. To delete a user-defined perspective:

- ▶ In the **Window** menu, select **Preferences...**



- ▶ The **Preferences** dialog opens up.
- ▶ In the left tree, navigate to the **General/Perspectives** page.
- ▶ Locate the section **Available perspectives**:

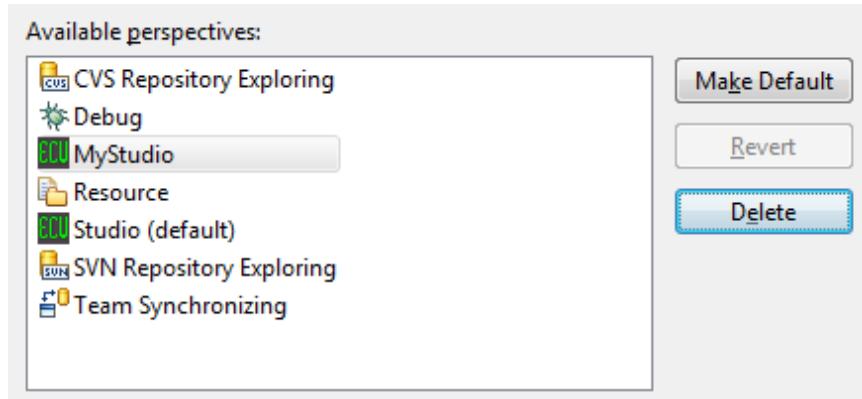
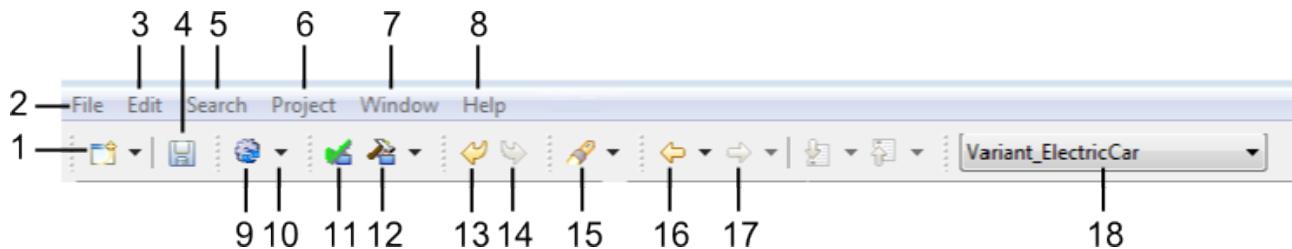


Figure 5.11. Deleting perspectives in the **Preferences** dialog

- ▶ Select the user-defined perspectives you want to delete and click **Delete**.
- ▶ Click **OK** in the **Preferences** dialog.
- ▶ The selected user-defined perspectives are deleted.

5.2. Menu bar and tool bar

The menu bar and the tool bar are located in the upper left corner of the screen:



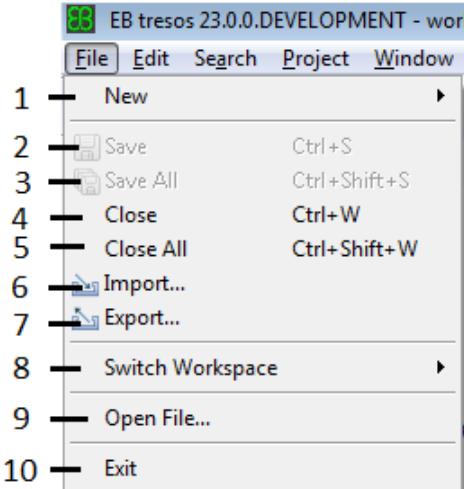
Number	Menu	Description
1	New button	Click to open a new project wizard.
2	File menu	Select for project unspecific tasks.
3	Edit menu	Select for tasks related to editing project data.
4	Save button	Click to save the configuration project of the currently active editor (see Section 6.3.2, "Saving a project").
5	Search menu	Select to search for and to find files, specific configuration parameters, or text strings



Number	Menu	Description
6	Project menu	Select to verify and generate code, see details in Section 6.9, “Generating code for projects” or to configure and run unattended wizards, see details in Section 6.8.10, “Autoconfiguring routine jobs” .
7	Window menu	Select to set preferences and to choose perspectives and views, see details in Section 6.3.6, “Closing and opening projects” , Section 5.1.1, “Customizing the workbench” and Section 5.1.2, “Using perspectives” .
8	Help menu	Select to access all global information and help functions for EB tresos Studio, see details in Section 6.2.1, “Viewing setup information” .
9	Run all unattended wizards marked with a '*' button	Click to run all Unattended Wizard(s) for a specified project, which are selected to run manually.
10	Menu button	Click to configure or run unattended wizards, see details in Section 6.8.10, “Autoconfiguring routine jobs” .
11	Verify selected project button	Click to start verifying code, see details in Section 6.9.1, “Verifying a project”
12	Generate button	Click to start generating code, see details in Section 6.9, “Generating code for projects” .
13	Undo button	Select to undo the last edit. The button is active only if undoable change options are available.
14	Redo button	Click to redo the last undone edit. The button is active only if redo change options are available.
15	Search button	Click to open the Search window default page, which is currently the File Search page.
16	Back button	Click to move to the last editing position.
17	Forward button	Click to move forward to the next editing position. This option is only available if the Back button or the corresponding menu was selected previously.
18	Change the PostBuild Selectable variant combo box	Click to switch between selectable variants in the EcuC module in the currently selected project. For more information, see Section 6.13.1.4, “Fast access tool bar” .

5.2.1. File menu

The **File** menu is located on the far left of the EB tresos Studio main window.

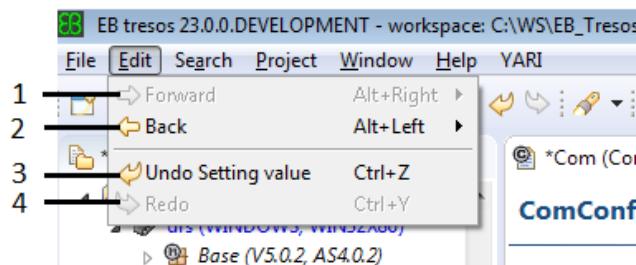


Number	Menu	Description
1	New	Select Project in the sub menu to create a new EB tresos Studio project Section 6.3.1, “Creating a new project” .
2	Save	Select to save the content of the currently active editor. If the editor is a configuration editor, the associated configuration project is saved Section 6.3.2, “Saving a project” .
3	Save all	Select to save the content of all open editors.
4	Close	Select to close the currently active editor. If you have modified any content since saving, a pop-up window is going to ask you to save unsaved content (see Section 6.3.6.1, “Closing projects”).
5	Close all	Select to close all open editors. If content of one or more editors is unsaved, a pop-up window is going to ask you to save unsaved content.
6	Import	Select to import projects (see Section 6.10.1.1, “Importing a project”).
7	Export	Select to export projects (see Section 6.10.1.2, “Exporting a project”).
8	Switch Workspace	Select to select a new workspace directory. If you select a new workspace directory, EB tresos Studio shuts down and restarts with the new workspace. See the concept of workspaces Section 3.5, “EB tresos Studio projects and workspaces” .
9	Open File	Select to open a file.
10	Exit	Select to close EB tresos Studio. A pop-up window is going to ask you if you want to save any changes before closing the tool.



5.2.2. Edit menu

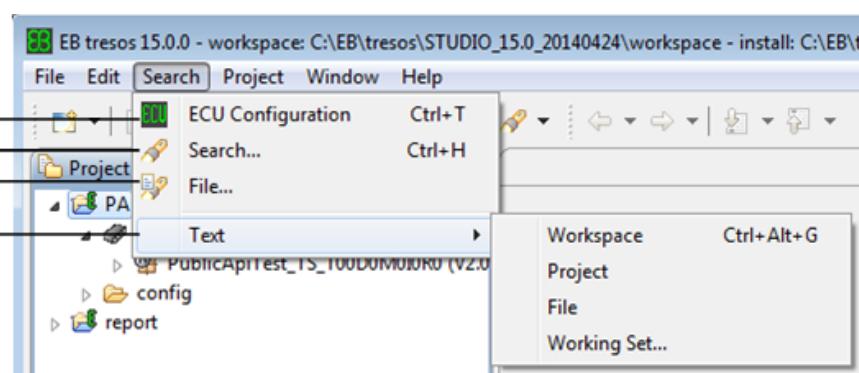
The **Edit** menu is the second drop down menu from the left. The menu items in this menu are active only if you are working with an editor.



Number	Menu	Description
1	Forward	Select to move to the next editing position in the editor.
2	Back	Select to move to the previous editing position you worked on in the editor.
3	Undo ...	Select to undo the last change. Instead of "..." the name of the action that would be undone is displayed.
4	Redo ...	Select to redo the last change. Instead of "..." the name of the action that would be redone is displayed.

5.2.3. Search menu

The **Search** menu enables you to search for and to find files, specific configuration parameters, or text strings.



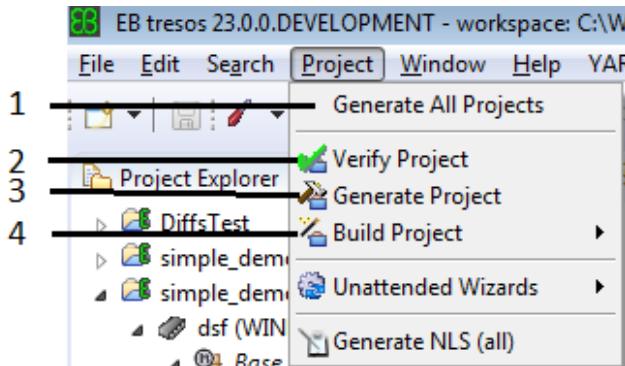
Number	Menu	Description
1	ECU Configuration	Select to open the configuration parameter search within open EB tresos Studio projects. Instead of clicking the menu



Number	Menu	Description
		item, an alternate way of opening the dialog is to press Ctrl + T .
2	Search	Select to open the Search window, which is a dialog with a tab for every different type of search. If this dialog is invoked from a configuration editor, the ECU Configuration Search is active directly when the dialog is opened.
3	File	Select to open the file search. Within the file search you may search for files with a specific name or content.
4	Text	Select to search for text strings. As search-pattern, the currently selected text is used. This menu option will open a submenu with different search scope options.

5.2.4. Project menu

The **Project** menu is the fourth drop down menu from the left.



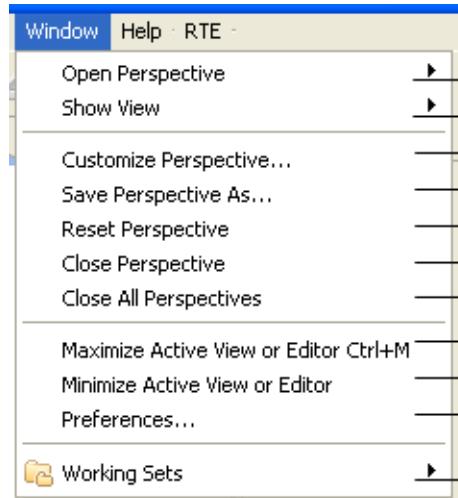
Number	Menu	Description
1	Generate All Projects	Select to generate the code for all projects Section 6.9.2, "Generating a project" .
2	Verify Project	Select to verify the project currently selected in the Project Explorer Section 6.9.1, "Verifying a project" . If no project is selected in the Project Explorer , this entry is grayed out.
3	Generate Project	Select to generate the project that is currently selected in the Project Explorer (see Section 6.9.2, "Generating a project"). If no project is selected in the Project Explorer , this entry is grayed out.
4	Build Project	Select to open a submenu that contains a list of generation modes Section 6.9.3, "Building a project" . The available gen-



Number	Menu	Description
		eration modes are determined by the modules of the currently selected project. See the documentation of your installed modules. If no project is selected in the Project Explorer , this entry is grayed out.

5.2.5. Window menu

The **Window** menu is the fifth drop down menu from the left.



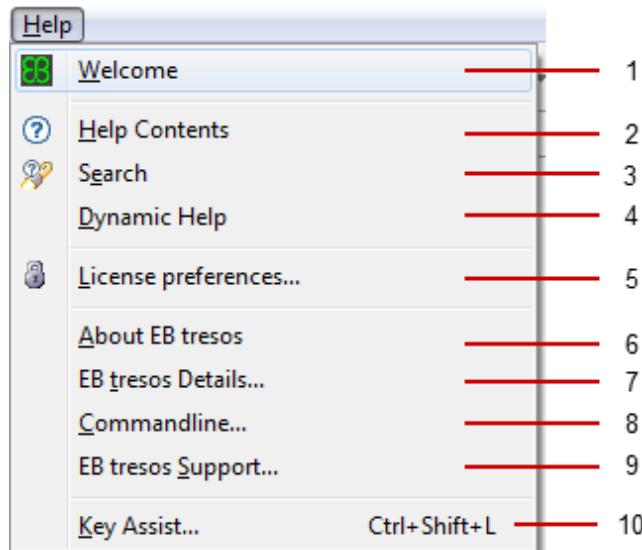
Number	Menu	Description
1	Open Perspective	Select Open Perspective , then select Other.. to open a perspective. A perspective is an arrangement of views. See also Section 5.1.2.2, "Opening perspectives" .
2	Show View	Select Show View , then select Other to reopen a view. A view in this context is a part of a window such as the Project Explorer .
3	Customize Perspective...	Currently not used by EB tresos Studio.
4	Save Perspective As...	Select to store the current workbench layout as a named perspective so that you can restore the layout later on. See also Section 5.1.2.1, "Saving user-defined perspectives" .
5	Reset Perspective	Select to reset the current perspective to its pre-defined or saved state. See also Section 5.1.2.5, "Resetting perspectives" .
6	Close Perspective	Select to close the current perspective. If you close the last open perspective, the workbench is emptied. See also Section 5.1.2.4, "Closing perspectives" .
7	Close All Perspectives	Select to close all perspectives and empty the workbench.



Number	Menu	Description
8	Maximize Active View or Editor	Select to enlarge the currently selected view to its maximum. This means that all other views will be minimized and may disappear from view.
9	Minimize Active View or Editor	Select to minimize the currently active view. This means that the active view may disappear from view and just the other views are displayed.
10	Preferences	Select to change the EB tresos Studio preferences Section 6.2.2, "Setting preferences" .
11	Working Sets	Currently not used by EB tresos Studio.

5.2.6. Help menu

The **Help** menu provides information about the tool, software updates, and license preferences.



Number	Menu option	Description
1	Welcome	Select to reopen the welcome screen.
2	Help Contents	Select to open the help window and browse the online help function of EB tresos Studio.
3	Search	Select to search the online help of EB tresos Studio.
4	Dynamic Help	Currently not supported in EB tresos Studio.
5	License preferences...	Select to open the preferences dialog on the Licenses page, see Section 6.2.2.10, "Setting License preferences" .

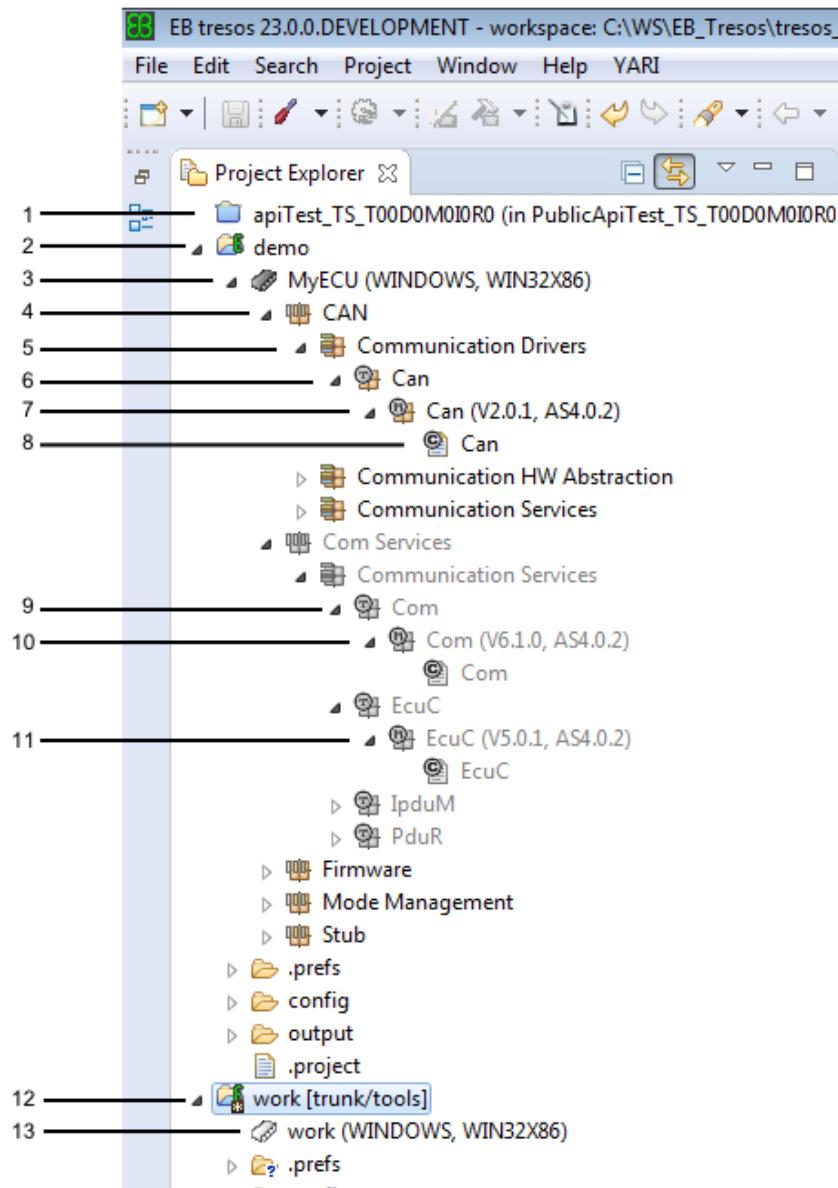


Number	Menu option	Description
6	About EB tresos	Select to access information about the EB tresos home page and support page.
7	EB tresos Details	Select to access detailed information about EB tresos Studio, to list the variant, the release version, the build number, the install location, the workspace location, all features, all modules and all bundles.
8	Command line	Select to browse the EB tresos Studio command line options and view the command line options help messages.
9	EB tresos Support...	Opens the EB tresos support web page in an Internet browser.
10	Key Assist	Currently not supported in EB tresos Studio.

5.3. Views

5.3.1. Project Explorer view

The **Project Explorer** is the main utility to work with EB tresos Studio projects. The **Project Explorer** displays all EB tresos Studio configuration projects in a workspace and allows you to open editors. The project explorer displays a tree. The tree starts with the projects on the first level. The second level contains the ECU configuration node. Beneath that level, the tree continues with several layers down to the module configurations. All items of the tree are represented by different icons. The items may change their state which is also reflected by their icon. The following picture gives you a comprehensive overview:



Number	Description
1	Closed project
2	Project name
3	Loaded configuration
4	Category (node)
5	Layer (node)
6	Type (node)
7	Module (node)
8	Module configuration (node)

Number	Description
9	Disabled module configuration
10	Disabled module configuration
11	Disabled module configuration
12	Open project
13	Unloaded project

NOTE

A project which contains an ECU configuration usually is called a *configuration project* for short.



5.3.1.1. Project and ECU configuration states

Projects and ECU configurations can be in different states each. A project can be either open or closed. An ECU configuration can either be loaded or unloaded.

The following table gives an overview over the different project states and explains their use:

Project State	Icon	Description
open		In this state the Project Explorer displays the ECU configuration node for this project. This is the default state for projects after start-up.
closed		In a closed project you cannot edit any configurations, nor can you generate any code. The project consumes the least possible resources.

The following table gives an overview over the different ECU configuration states and explains their use:

ECU Configuration State	Icon	Description
unloaded		In an unloaded ECU configuration you can view the configuration's properties such as the ECU Id or the target and derivate values, but you cannot edit them. The ECU configuration data is not loaded and therefore you can neither edit configurations, nor can you generate any code in this state. The benefit of this state is its very low resource consumption. This state is



ECU Configuration State	Icon	Description
		the default state for projects when EB tresos Studio is started.
loaded		A loaded ECU configuration is fully functional. Loaded configurations can be edited, modules can be added and code can be generated. Beneath the ECU configuration node, a hierarchical configuration tree displays configurations grouped according to categories. Newly created projects are in this state.

5.3.1.2. Layering and grouping of module configurations

The **Project Explorer** groups module configurations according to the modules they belong to. The modules themselves are grouped into hierarchical layers of categories.

Layer name	Icon	Description
Loaded configuration	icon in the shape of a chip	Represents the loaded configuration of a project.
Category node/layer	icon contains three vertically ordered bars	Provides a horizontal grouping of modules within an operating system stack.
Layer node/layer	icon contains three horizontally ordered bars	Provides a vertical grouping of modules within an operating system stack.
Type node/layer	icon with a T on the folder	Describes the functionality of the module, e.g. Can for a CAN driver.
Module node/layer	icon with an M on the folder	Is the actual implementation of a module, such as a CAN driver for a specific target and derivate.
Module configuration node/layer	icon with a C on the folder	Is the configuration for an actual implementation of a module (see module node); the leaf node represents either: ▶ a module configuration or



Layer name	Icon	Description
		► an editor if the module provides multiple editors for configuration.

5.3.1.3. Changing the visible layers

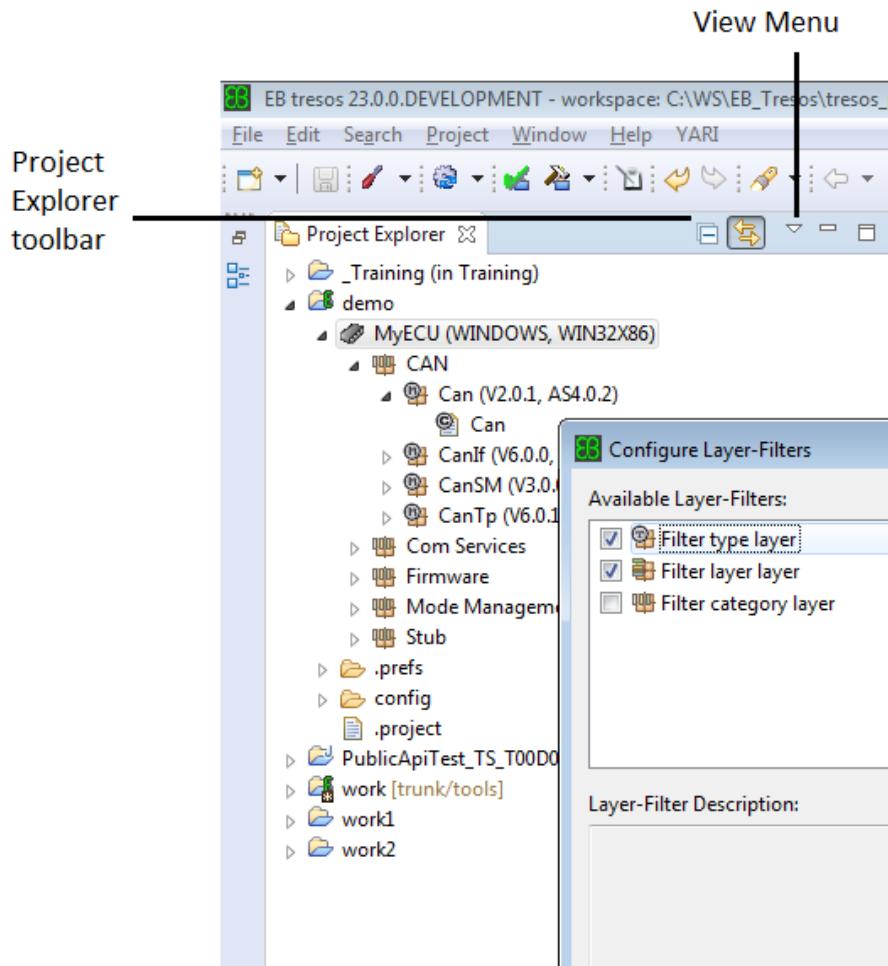
You can customize the layer grouping of the **Project Explorer** by selecting which layers shall be displayed. To change this:

- Click the **View** button in the Project Explorer tool bar.

A context menu opens up.

- Select the **Layer Filters** menu option.

The **Configure Layer Filters** window opens up.





If a layer is disabled, it is excluded from the tree. The content beneath the hidden nodes however, is not hidden. Instead, the content of the hidden nodes is displayed one level higher than it normally would have been. Nodes are grouped together if their parents are the same and if they share the same label.

The **Configure Layer Filters** window provides check boxes to disable layers.

To disable the *category* layer

- ▶ Select the check box **Filter category layer**

To disable the *layer* layer

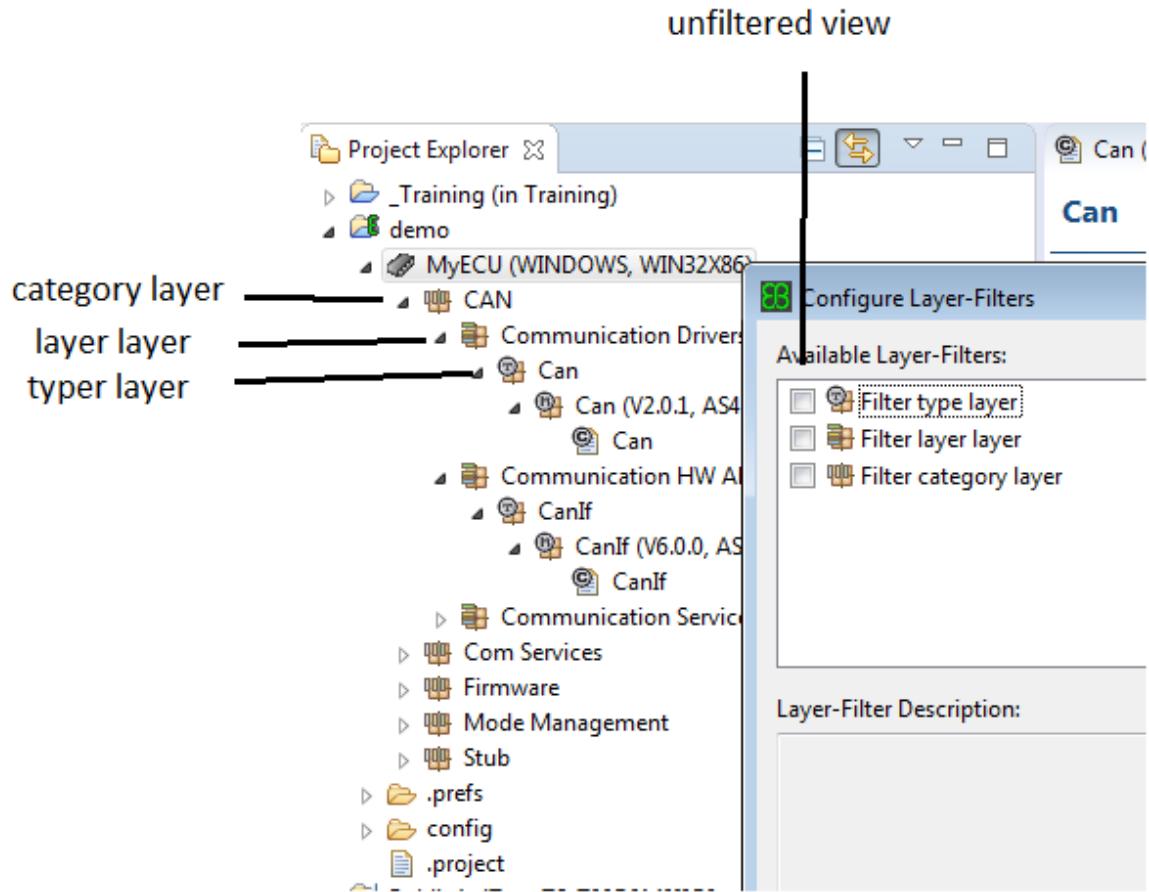
- ▶ Select the check box **Filter layer layer**

To disable the *type* layer

- ▶ Select the check box **Filter type layer**

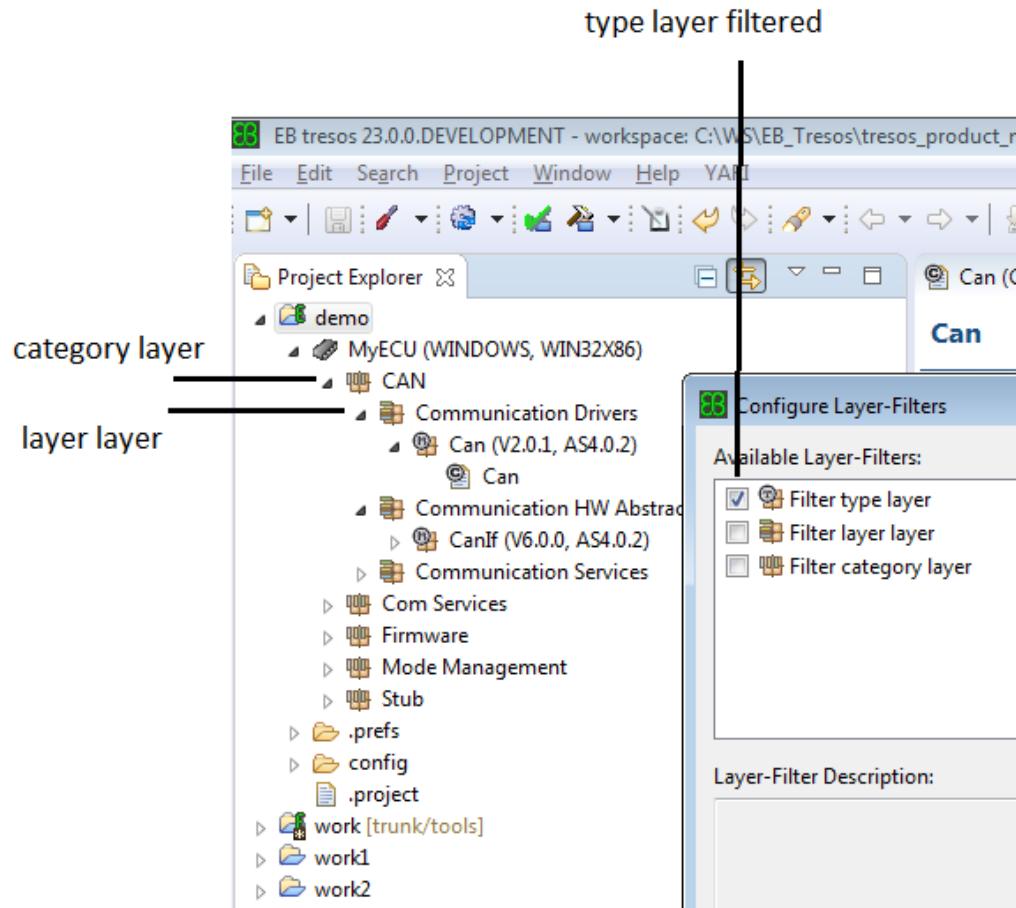
Example: Unfiltered view

- ▶ The unfiltered view shows
 - ▶ the category layer
 - ▶ the layer layer
 - ▶ the type layer
- as well as the module layer and the module configuration layer



Example: Type layer filtered out

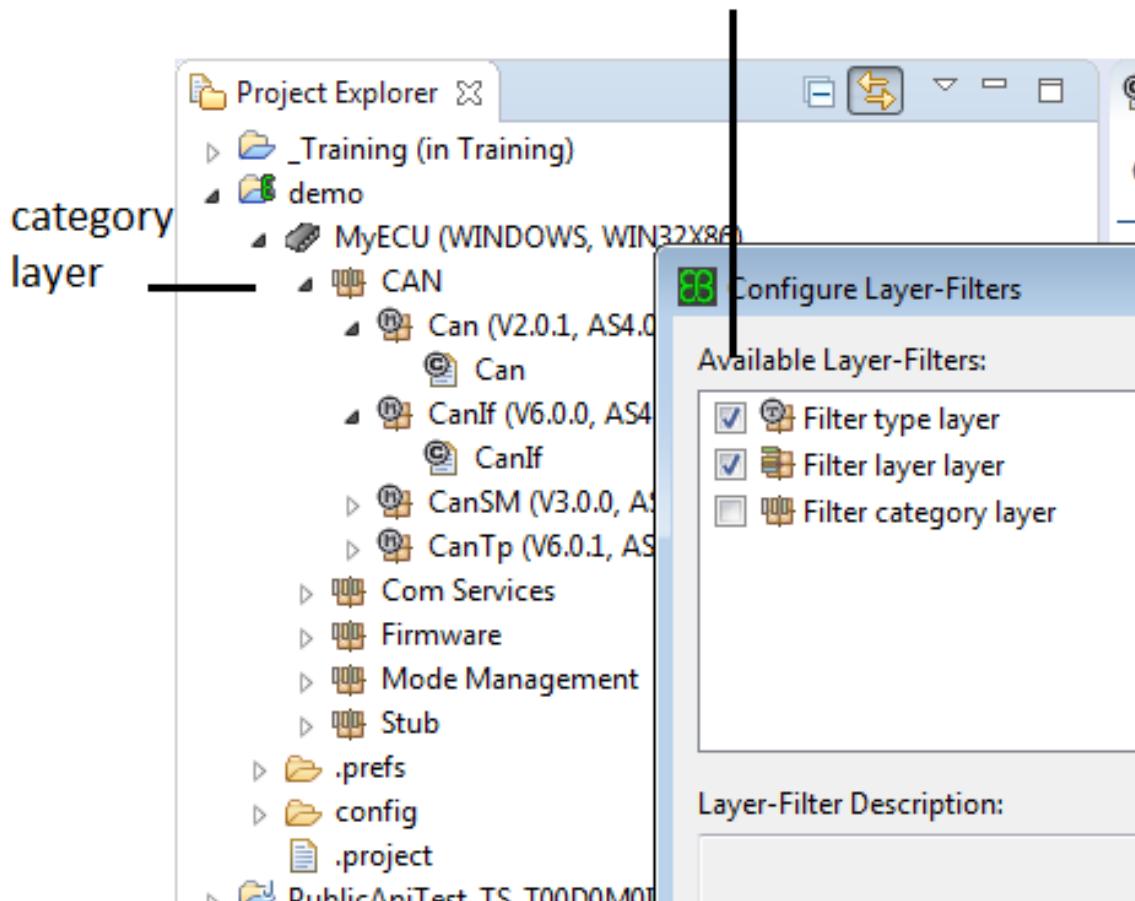
- ▶ If the type layer is filtered out, the **Project Explorer** shows
 - ▶ the category layer
 - ▶ the layer layer
 - ▶ as well as the module layer and the module configuration layer



Example: Type and layer layer filtered out

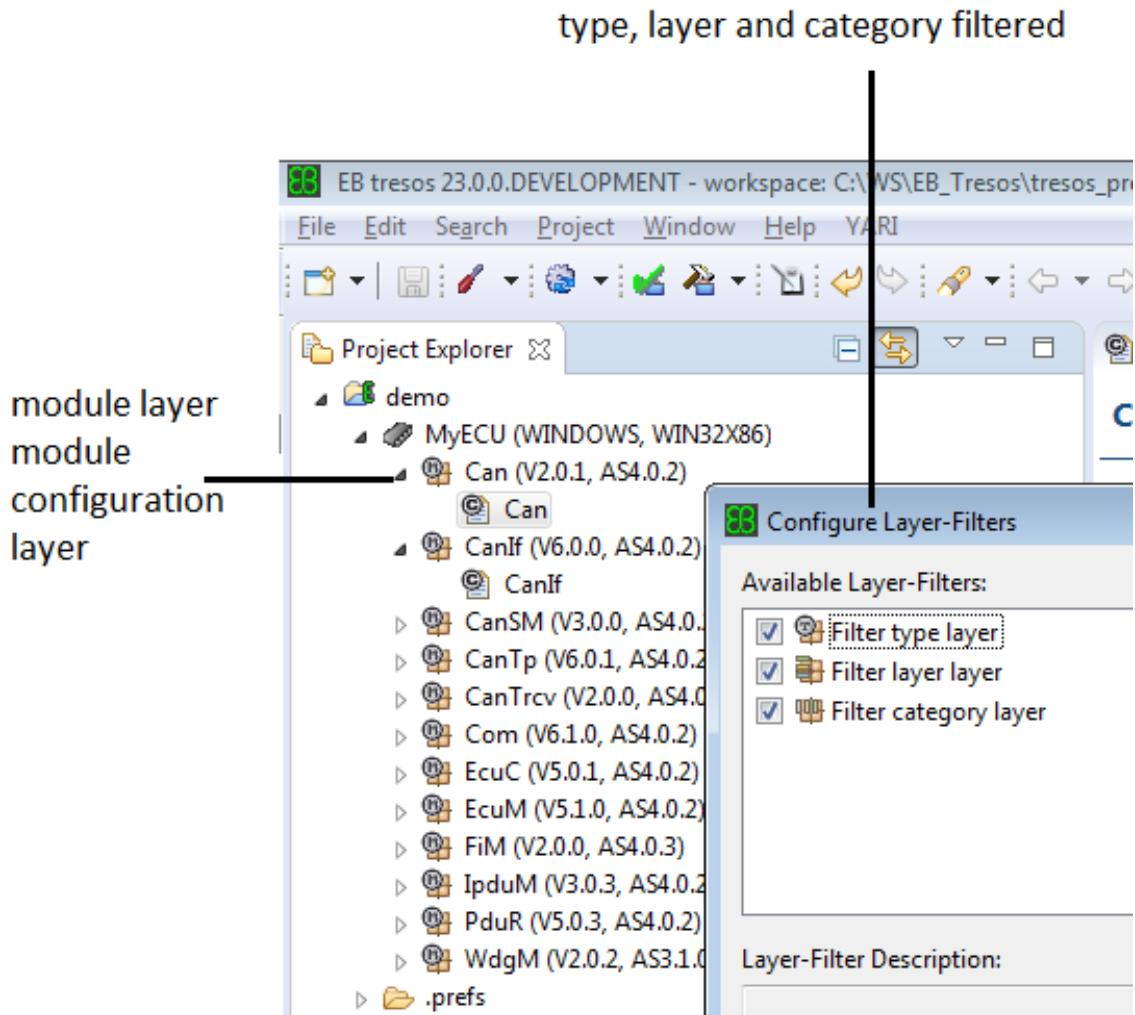
- ▶ If the type and layer layer is filtered out, the **Project Explorer** shows
 - ▶ the category layer
 - ▶ as well as the module layer and the module configuration layer

type layer and layer layer filter



Example: Type, layer, and category layer filtered out

- If the type, layer, and category layer are filtered out, the **Project Explorer** shows the only module layer and the module configuration layer.

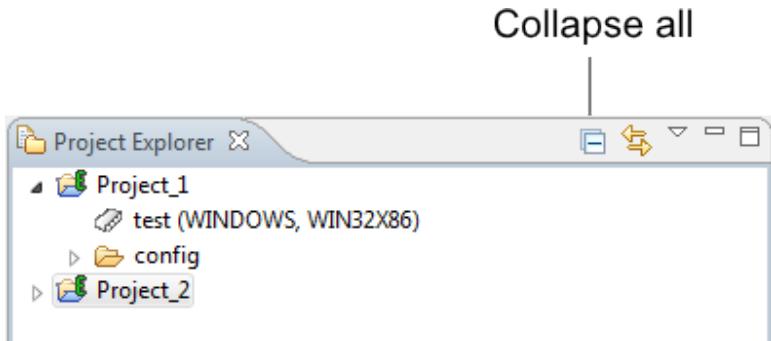


5.3.1.4. Collapsing the Project Explorer tree

To collapse the whole project tree inside the **Project Explorer** so that only the project names are visible:

- ▶ Click the **Collapse all** button from the **Project Explorer** tool bar.

Only project names are visible now.



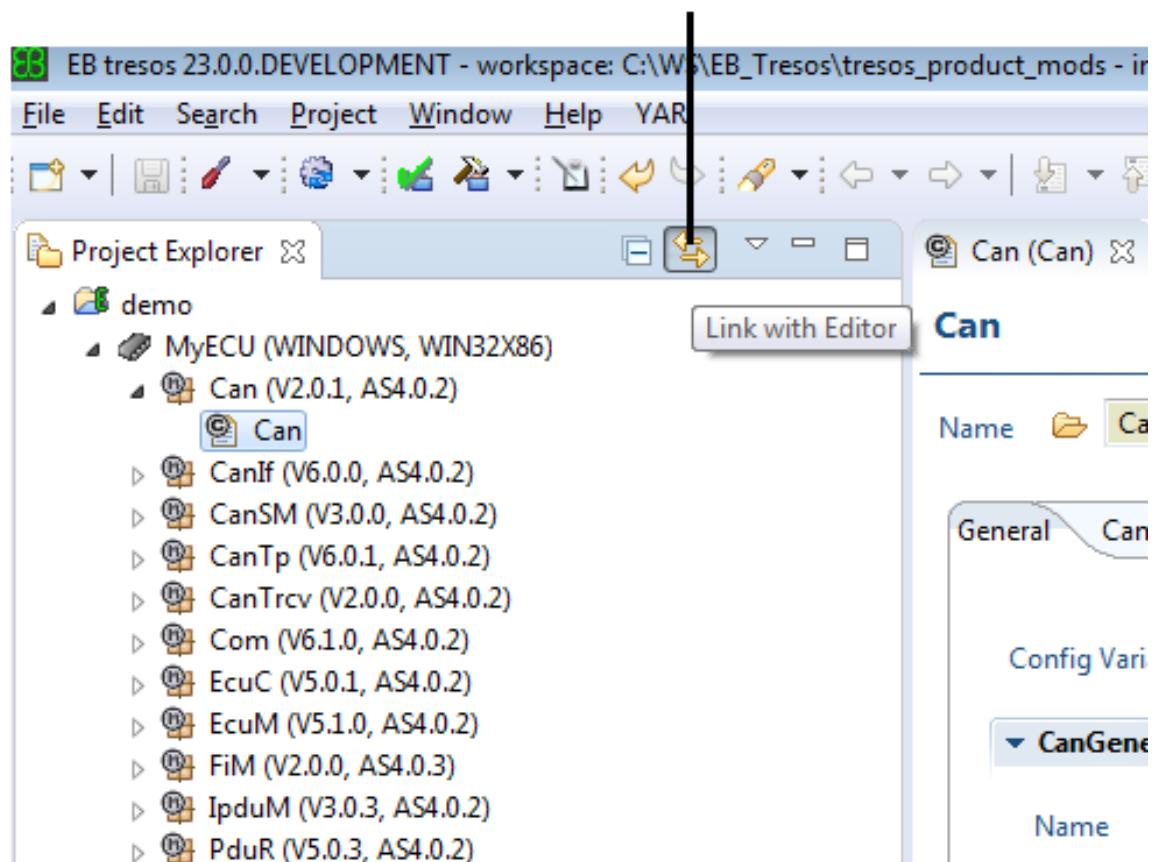
5.3.1.5. Synchronizing the Project Explorer with an editor

To synchronize the **Project Explorer** view with the active editor on the right:

- ▶ Click the **Link with editor** button.

The **Project Explorer** expands a module configuration and scrolls it into view for which ever corresponding editor is selected.

Link with Editor



5.3.1.6. Indicating different project states with colors

The **Project Explorer** uses colors to indicate the different states that the module configuration can be in. To change colors of any or all meanings, see [Section 6.2.2.12, "Setting colors and fonts"](#).

Color	Meaning
Black	default color for nodes
Blue	<ul style="list-style-type: none">▶ default color for modified and not yet saved module configurations▶ used in layers in helping to find the modified module configuration if the Project Explorer tree is collapsed▶ used in editors to highlight modified parameters
Gray	<ul style="list-style-type: none">▶ default color for disabled module configurations▶ if one or more module configurations are disabled, the whole subtree of the disabled module configurations is shown as disabled in gray



5.3.2. The Editor view

The **Editor** view is the main utility to edit a module configuration and the parameters involved in this module. The **Editor** view opens the Generic Configuration Editor, which corresponds to the module you want to edit. Thus, the **Editor** view may look a little different, i. e. have different tabs, for each module. The following image gives you an overview of the main elements of the **Editor** view. A more detailed description of the **Editor** view and instructions for working with the **Editor** view are available at [Section 6.8, “Editing parameters of a module configuration”](#).

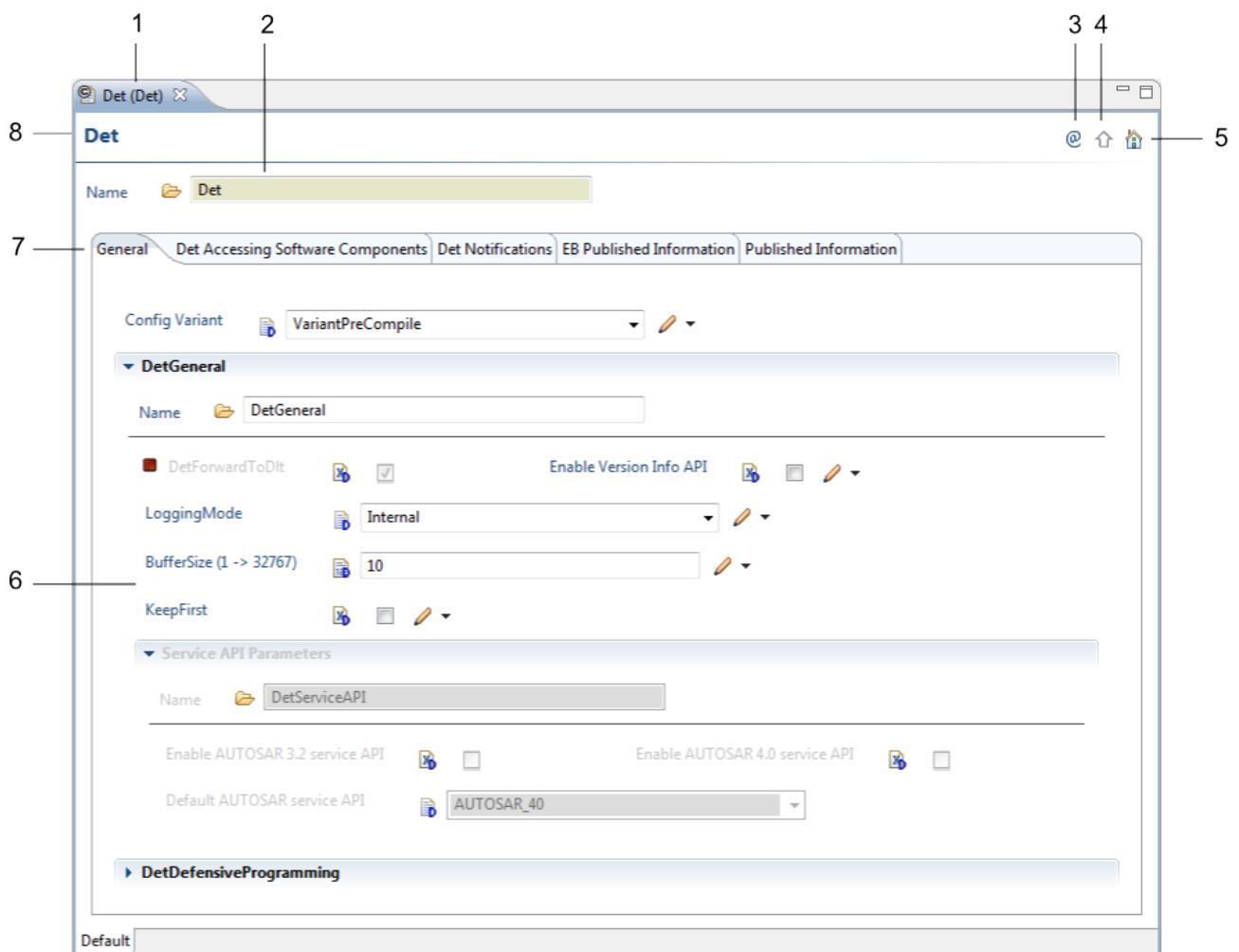


Figure 5.12. The GUI elements of the Editor view: page header, name, title and tabs

Number	Name	Explanation
1	Title	Shows the title text with the module name, the module configuration name in brackets, and a close button.



Number	Name	Explanation
		<p>NOTE  The AUTOSAR module name is the title The equivalent of a title in an AUTOSAR module is the name of the CONTAINER-DEF or MODULE-DEF.</p> <p>TIP  Point the mouse over the title for tool tips To display a tool tip with the module configuration name, the module-ID, and the project the editor belongs to, place your mouse pointer over the title.</p>
2	Name	<p>Displays the name of the element in the configuration that is represented by the page. If the element is an AUTOSAR module, the SHORT-NAME is displayed.</p> <ul style="list-style-type: none"> ▶ If the page represents a MODULE-CONFIGURATION element, you cannot change the name. ▶ If the page represents a CONTAINER, you may change the name.
3	 Navigate button	Click to enter an XPath path to a configuration parameter. The editor navigates to this parameter.
4	 Upward button	Click to navigate to one page up in the page hierarchy. The editor navigates to the page that hosts the current page.
5	 Home button	Click to navigate to the home page of this module.
6	Editing area	<p>Shows the current configuration parameters. This area displays one page/tab at a time. Pages are organized hierarchically in the form of a tree: The editor is the home page and all other pages are arranged below this home page.</p> <p>NOTE  The AUTOSAR equivalent of a page The equivalent of a page in an AUTOSAR module is a CONTAINER or a MODULE-CONFIGURATION.</p>
7	Tabs	Display the parameter in the editor. Only one tab at a time is displayed.



Number	Name	Explanation
		<p>NOTE</p> <p></p> <p>How tabs are organized</p> <ul style="list-style-type: none">▶ The General tab contains all containers, parameters, and references that appear exactly once. In AUTOSAR modules those elements are:<ul style="list-style-type: none">▶ containers: CONTAINER▶ parameters: *-VALUE▶ references: REFERENCE-VALUE▶ elements that appear exactly once: LOWER-MULTIPLICITY==UPPER-MULTIPLICITY==1▶ All elements that appear more often are displayed as a tab of their own.▶ The AUTOSAR INSTANCE-REFERENCE-VALUE is also displayed in its own tab, because an instance reference contains a table with context references.
8	Page header	Shows the title of the page on the left.

A more detailed description of the **Editor** view and instructions for working with the **Editor** view are available at [Section 6.8, “Editing parameters of a module configuration”](#).

5.3.3. Node Outline view

The **Node Outline** is a tree view outline of the editor contents. The **Node Outline** shows the editor contents in a hierarchical structure. Its main purpose is to navigate between parameters. The **Node Outline** displays the same icons as the Editor view for parameters.

The Node Outline in EB tresos Studio shows whether a parameter or container is variant affected or not. The state icon shown for each node provides an  overlay icon on the left bottom corner if the parameter or container can have variants.

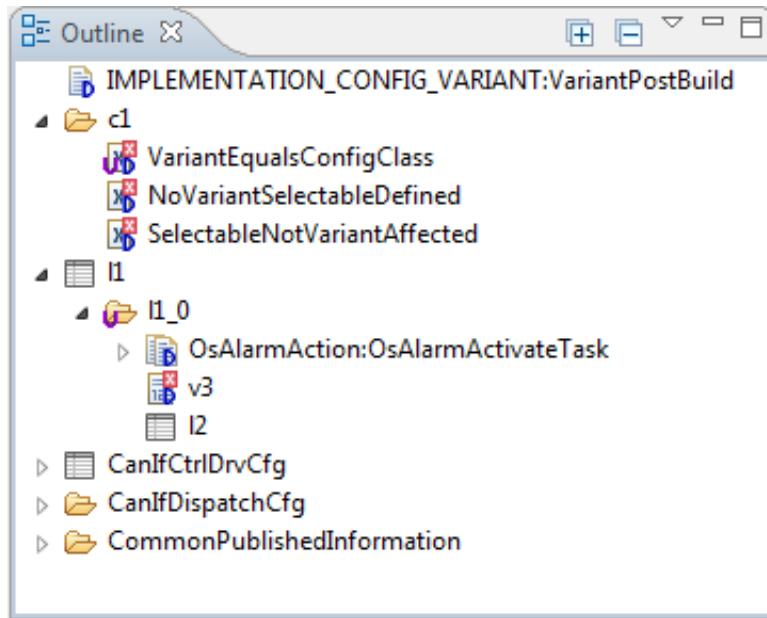


Figure 5.13. The **Node Outline** with variant-affected parameters

5.3.3.1. Changing the display settings of the Node Outline view

To change the display settings of the **Node Outline** view, either use the tool bar buttons or the actions from the context menu.

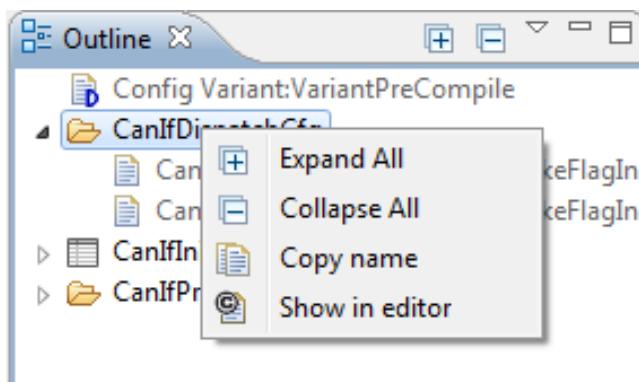


Figure 5.14. The **Node Outline** context menu

To display the complete node hierarchy of the selected node(s):

- ▶ Click the **Expand All** button in the tool bar or select the **Expand All** item from the context menu.

The complete node hierarchy of the selected node(s) is displayed.

To show the selected node(s) without showing their underlying structure:



- ▶ Click the **Collapse All** button  in the tool bar or select the **Collapse All** item from the context menu.

The selected node(s) are collapsed and the subelements are not displayed anymore.

If no node is selected, use the tool bar buttons to expand or collapse the complete tree.

5.3.3.2. Copying the node name

To copy the node name to the clipboard:

- ▶ Select one node in the tree.
- ▶ Right-click the node to open the context menu.
- ▶ Select **Copy name**.

5.3.3.3. Navigating to a node in the Editor view

To navigate to a parameter in the Generic Configuration Editor:

- ▶ Select one node in the tree.
- ▶ Right-click the node to open the context menu.
- ▶ Select **Show in editor**.

Now the parameter is shown in the Generic Configuration Editor and has the focus.

5.3.3.4. Editing variant conditions

To edit the variant conditions of a variant affected node:

- ▶ Select one variant affected node in the tree.
- ▶ Right-click the node to open the context menu.
- ▶ Select **Edit variant conditions**.

Refer to [Section 6.8.5, “Editing variant conditions”](#) for further information.

5.3.3.5. Deleting a variant node



To delete a variant of a variant affected node:

- ▶ Select one variant affected node in the tree.
- ▶ Right-click the node to open the context menu.
- ▶ Select **Delete variant**.

NOTE



Automatically re-added nodes

If you delete a variant node from the current selected PredefinedVariant and the parameter is mandatory, then EB tresos Studio automatically adds a new instance of the node which by default gets the variant conditions of the currently selected PredefinedVariant.

5.3.3.6. Filtering the displayed items

To filter the displayed items in the **Node Outline**:

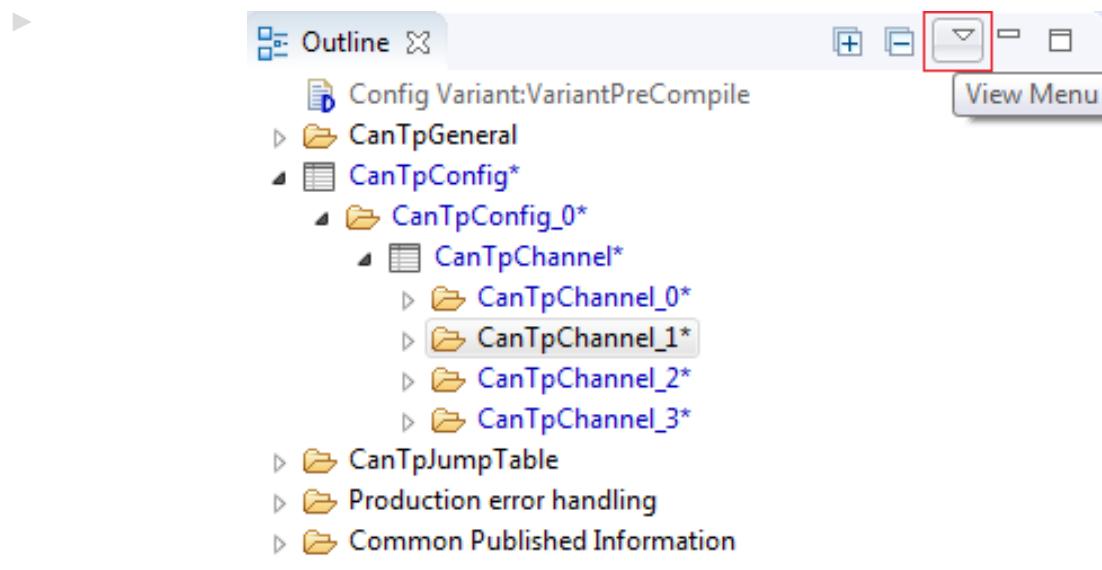


Figure 5.15. Opening the **Node Outline** menu

Open the **Menu** symbolized by a small arrow down in the **Node Outline** view tool bar.

A context menu opens up.



Figure 5.16. Opening the **Node Outline Filter** menu

- ▶ Select **Filters....**

The **Outline View Filter** window opens up.

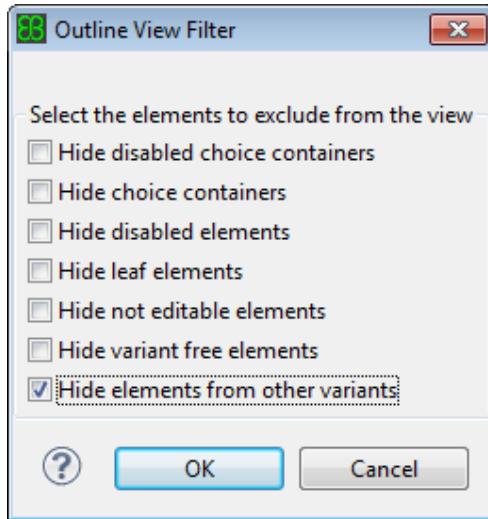


Figure 5.17. The **Outline View Filter** window

- ▶ Select the check boxes according to the results you wish to obtain:
 - ▶ To display just the active elements of a choice, select **Hide disabled Choice Containers**.
 - ▶ To hide containers inside a choice element, select **Hide choice Containers**.
 - ▶ To hide grayed out elements, select **Hide disabled elements**.
 - ▶ To hide parameters and references that show just the structure of the pages in the Generic Configuration Editor, select **Hide leaf element**.
 - ▶ To hide elements that cannot be changed in the Generic Configuration Editor such as AUTOSAR CommonPublishedInformation, select **Hide not editable elements**.
 - ▶ To hide all elements which can not have variants, select **Hide variant free elements**. By default this filter is disabled. This filter shall help you to navigate through all variant affected parameters and specify the variant conditions for them.
 - ▶ To hide all elements which are not part of the current variant, select **Hide elements from other variants**. By default this filter is enabled, so the Node Outline only shows elements which are part of the current variant.
- ▶ Click **OK**.

Your filter settings are saved and the items displayed in the **Node Outline** view are displayed according to these settings.

5.3.3.7. Appearance of the displayed items

To control whether the node names or the node label shall be shown for the items in the **Node Outline**:

- ▶ Click the **Menu** button symbolized by a small arrow down.



A menu opens up.

- ▶ Select **Show**.

A sub-menu with two options opens up.

- ▶ Select the option **Node Names** to display the names of the items shown in the **Node Outline** view. The names of the items in the view are identical to the names of the nodes in the Data Model underlying the editor representation.
- ▶ Select the option **Node Labels** to display the labels of the items. If an item does not have a label, the name of the item is shown instead. Labels usually provide a more meaningful description of the items.

The state the **Node Outline** view is currently in can be checked by opening the menu: a check mark is drawn before the item currently active. The setting is persistent and will be remembered if you close the tool.

5.3.4. Element Outline views

The **Element Outline** consists of three views that display information about the parameter currently selected in the editor. The three views are:

- ▶ the **Element Errors** view
- ▶ the **Element Information** view
- ▶ the **Element Description** view

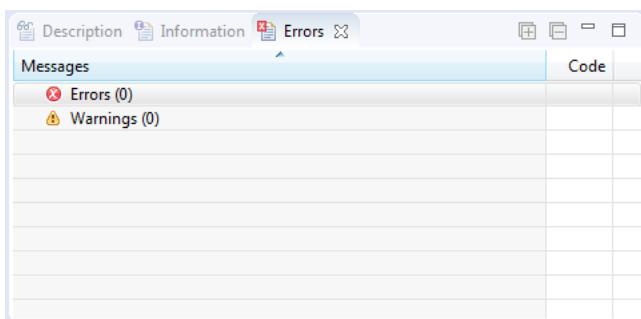
NOTE



Opening Element Outline views

The **Element Outline** views are not visible by default. To display the views, access **Window** → **Show View** → **Other** → **tresos Studio** and select either **Errors**, **Description** or **Information**, (see [Section 5.1.1.3, “Opening a view”](#)).

5.3.4.1. Element Errors view





The **Element Errors** view displays messages and errors just as the **Problems** view (see [Section 5.3.6, “Problems view”](#)) but only for the parameter currently selected in the editor. The error code provided is a tool-internal code used by EB tresos Studio developers to locate an error in the source code of the tool.

To display detailed message information:

- ▶ Double-click the entry in the **Element Errors** view.

The **Message Details** dialog opens up.

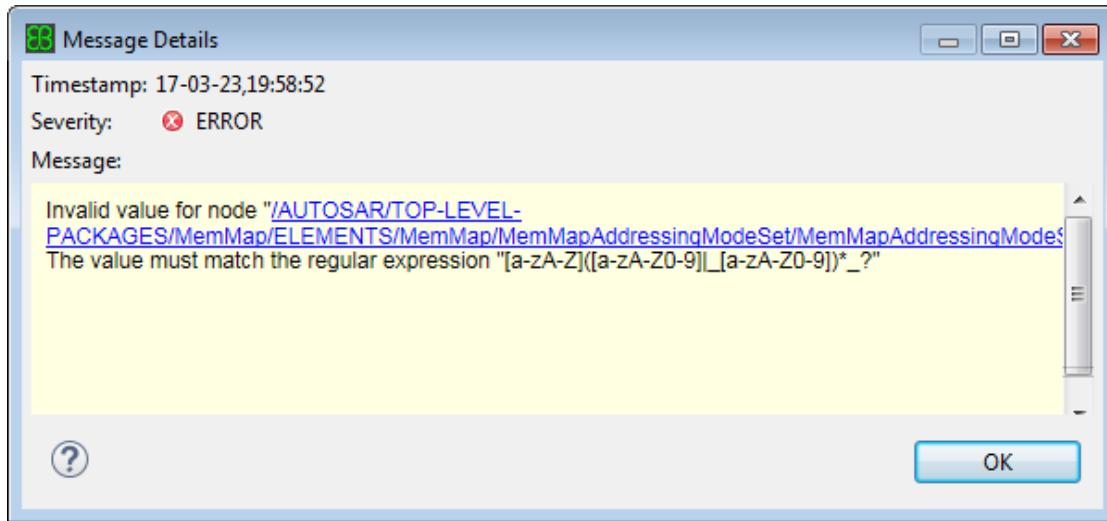
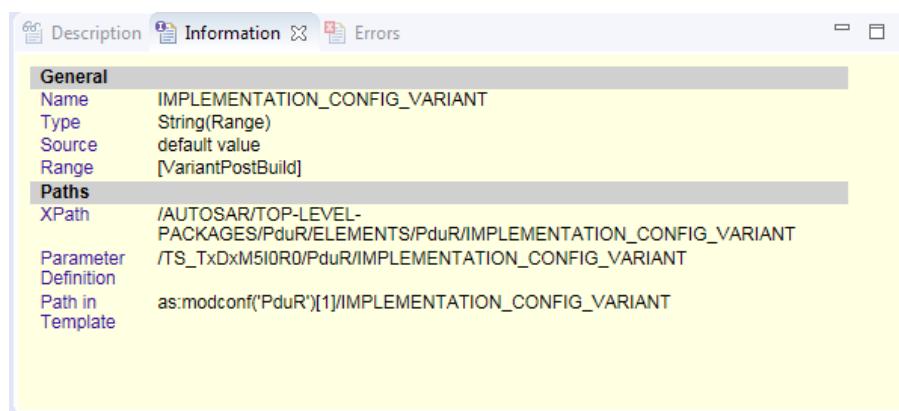


Figure 5.18. Details of the **Element Errors** view

The dialog displays the following information:

- ▶ **Timestamp**: the time when the error was reported.
- ▶ **Severity**: the message is an error or a warning.
- ▶ **Message**: the message information. You can select and copy the message.

5.3.4.2. Element Information view





The **Element Information** view shows parameter meta information of the parameter currently selected in the editor. This meta information is derived from the AUTOSAR parameter definition and may contain the following information:

- ▶ **Name**: the name of the element
- ▶ **Type**: the element type (integer, floating point unit, Boolean, string, etc.)
- ▶ **Source**: what the source of the parameter is (e.g. **manually edited** or **default value**).
- ▶ **SHORT-NAME**: the AUTOSAR SHORT-NAME of the parameter
- ▶ **LONG-NAME**: the AUTOSAR LONG-NAME of the parameter
- ▶ **Min**: the minimum number of entries in a table
- ▶ **Max**: the maximum number of entries in a table
- ▶ **Range**: the valid values for this parameter
- ▶ **ORIGIN**: the AUTOSAR ORIGIN of the parameter
- ▶ **UUID**: the AUTOSAR UUID of the parameter
- ▶ **XPath**: the XPath path expression via which it is possible to navigate to the parameter
- ▶ **ASPath**: The AUTOSAR path expression of the parameter. For reference parameters, the path describing all referenceable nodes is shown here. This path is also shown if there is no referenceable node available (e. g. not yet configured). It may contain a prefix such as `ASPathDataOfSchema:`, which means, that all configured nodes for a specific schema-node are referenceable. In this case, the AUTOSAR SHORT-NAME-path of the schema-node is the last part of this path.
- ▶ **Parameter Definition**: displays the path to the AUTOSAR parameter definition element
- ▶ **Path to referenceable nodes**: the path to the schema node whose data nodes may be referenced by this reference
- ▶ **Multiple Configuration**: displays **Yes** if the selected element is an AUTOSAR multiple configuration container
- ▶ **Config Class**: the AUTOSAR CONFIG-CLASS of the parameter
- ▶ **Config Variant**: the AUTOSAR CONFIG-VARIANT of the parameter
- ▶ **Category**: the AUTOSAR CATEGORY of the parameter
- ▶ **Post Build**: if the AUTOSAR attribute POSTBUILDCHANGEABLE is set to true, **true** is displayed here
- ▶ **Path in Template**: When referencing a parameter within a code template of the template-based code generator, the displayed path may be used. The path is only displayed if you select the preference **Show template path in Properties view**.

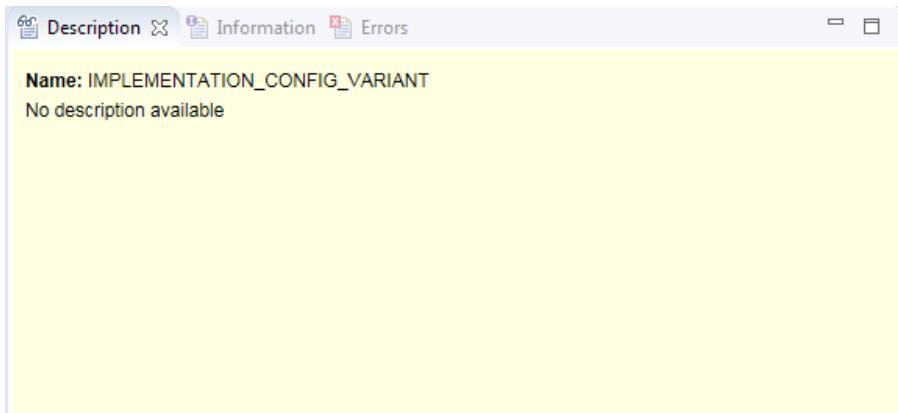


NOTE



Not all information is available for every configuration parameter. E. g. the **ASPath** is omitted if the parameter is not referenceable by an AUTOSAR path.

5.3.4.3. Element Description view



The **Element Description** view describes the parameter selected in the editor and gives configuration hints, references other parameters, etc. The description may contain hyperlinks to related parameters (not shown in the screenshot).

To navigate to the related parameter:

- ▶ Select the linked parameter from the **Element Description** view.

This opens up information about the parameter in the **Element Description** view.

5.3.5. Error Log view

The **Error Log** view is where all logged errors are visible in the GUI. The **Error Log** receives error messages from EB tresos Studio components and from other framework components. By default the list of messages in the **Error Log** is preserved in between the sessions of EB tresos Studio, thus the **Error Log** may start up filled with messages.

The line above the header of the shown table contains information about entries which are filtered out.



TIP



Section [Section 6.2.2.9, “Setting Error Log preferences”](#) describes ways to filter out messages and to customize the behavior of the **Error Log**.

The default location of the **Error Log** view is on bottom of the screen in the middle.

Message	Error Code	Entry Type	Origin	Timestamp
✖ The following exception occurred while executing the operati	1952	Tresos Status	Tresos	17-03-23,17:54:10
✖ Exception occurred: java.lang.reflect.InvocationTargetException		Exception	Java	17-03-23,17:53:54
✖ Error in file "C:\WS\EB_Tresos\tresos_product_mods\Diffstes	1636	Tresos Status	Tresos	17-03-23,17:53:53
✖ Error in file "C:\WS\EB_Tresos\tresos_product_mods\Diffstes	1636	Tresos Status	Tresos	17-03-23,17:53:53
⚠ Cannot import into project simple_demo_rte because no Mo	20015	Tresos Status	Tresos	17-03-23,17:40:40
⚠ Nothing to import in file D:\Share\ts_mirr\cessar-ct\2014C\S	20011	Tresos Status	Tresos	17-03-23,16:07:18
⚠ The xml-Tag "AUTOSAR" contains an invalid tag (which is no	23007	Tresos Status	Tresos	17-03-23,16:07:18
✖ Unknown moduleId "Os_TS_T19D1M4I4R0" for the extensio	11065	Tresos Status	Tresos	17-03-23,16:01:52
✖ cannot checkout	-6	IStatus	org.tigris.subversion.subclipse...	17-03-23,16:00:36
⚠ Authorization infrastructure (org.eclipse.core.runtime.compa	0	IStatus	org.eclipse.core.runtime	17-03-23,15:08:04
⚠ Missing mandatory module "Make_TS_TxDxM4I0R0" in projec	11115	Tresos Status	Tresos	17-03-23,14:43:21
⚠ Missing mandatory module "Base_TS_TxDxM5I0R0" in projec	11115	Tresos Status	Tresos	17-03-23 14:43:21

The **Error Log** displays a list of log entries, which can have the following properties:

Number	Information type	Explanation
1	Message	<p>Indicates the actual error message. The type of message and its severity is indicated by the icon in front of the message.</p> <ul style="list-style-type: none">▶ ✖ indicates an error message (something did not go as expected and it had to be aborted)▶ ⚠ indicates a warning message (something did not go as expected, but it could continue)▶ ⓘ indicates an info message (just a notification or feedback for the user)
2	Error Code	Indicates the error number, which is used by EB tresos Studio developers to identify the error in the software code.
3	Entry Type	Indicates the error type, e.g. an exception, the EB tresos status, etc.
4	Origin	Indicates the component that logged the message.
5	Timestamp	Indicates the day and time the message was produced.

To find out more detailed information about an error:



- ▶ Double-click the entry you want to learn more about in the **Error Log**.

The **Error Log Entry Details** dialog opens up.

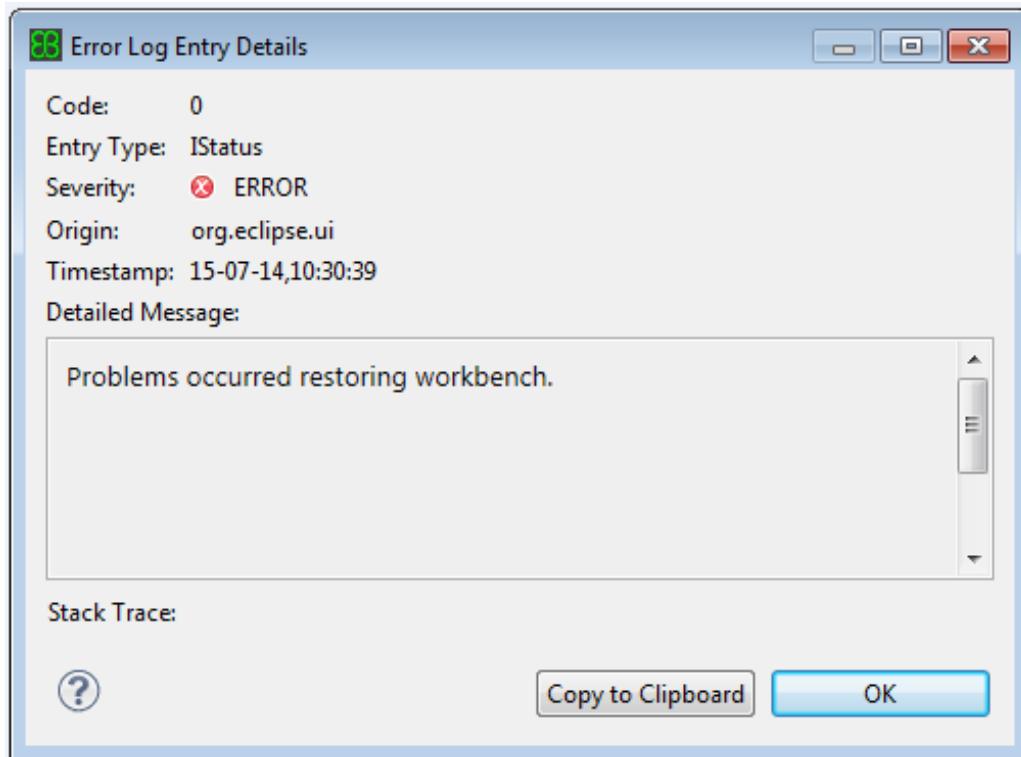


Figure 5.19. Details of the **Error Log** view

Additionally to the **Error Log** view information, the dialog displays a detailed message text. Links may be available and will navigate you directly to the point of interest. You can select and copy the message.

5.3.5.1. Changing the display settings of the **Error Log** view

To display the message hierarchy of all log entries into messages and submessages:

- ▶ Click the **Expand all** button .

The view of all messages is expanded and all messages with submessages are displayed.

To show just the top-level messages instead of all submessages:

- ▶ Click the **Collapse all** button .

The message view collapses and displays just top-level messages.

To write a message to a text file:



- Click the **Export log to a file** button .

A dialog window opens up, which enables you to save the message to your computer. This may be useful, e.g. if you want to contact EB tresos Studio support with a problem connected with the message.

To clear the Error Log view:

- Click the **Clear log view** button .

All messages disappear from the Error Log view. The messages are not lost but can be restored with the restore button.

To restore messages to the Error Log view:

- Click the **Restore log view** button .

All messages you previously cleared are restored to the Error Log view.

5.3.5.2. Filtering Error Log messages by their severity level

You may filter out messages depending on their severity level.

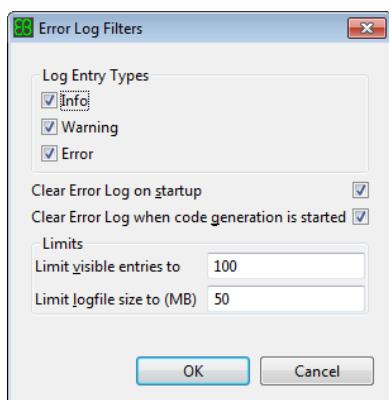
- Click the **View** button .

A context menu opens up.



- Select **Filter Entries**.

The **Error Log Filters** window opens up.





- ▶ Select the **Log Entry Types** you want the Error Log view to display.
- ▶ Click **OK**.

The Error Log view displays only messages that you selected in the **Error Log Filters** window.

5.3.5.3. Filtering Log messages by their Message Code

To filter out a certain unwanted message permanently, you have to add an entry to the file `tresos_gui.ini` within the `bin` directory of your EB tresos Studio installation. In this file you can filter out any warning or info message (e.g. during load or generate) by adding one line for each message:

`-Dmsg<Message Code>=false`

For example:

`-Dmsg1134=false`
`-DmsgMemMapG_3=false`

5.3.5.4. Restricting the amount of shown Error Log messages and the backup logfile size

You may restrict the amount of shown Error Log messages and the size of the backup logfiles.

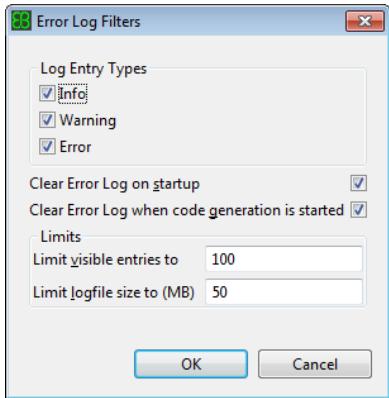
- ▶ Click the **View** button .

A context menu opens up.



- ▶ Select **Filter Entries**.

The **Error Log Filters** window opens up.



- ▶ Enter the amount of Error Log Entries you want to see at most.
- ▶ Enter the maximum size for backup logfiles.
- ▶ Click **OK**.

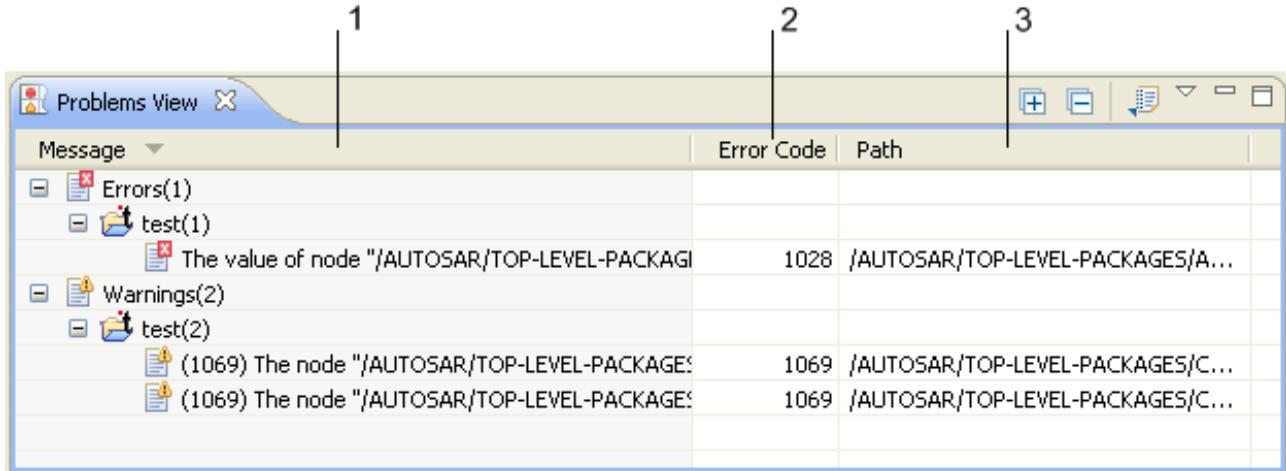
The Error Log view displays at most the number of messages you configured. All backup logfiles created in the future will not be bigger than the configured value.

5.3.5.5. Copying Error Log entries into text documents

You can copy an Error Log entry as plain text to the Windows clipboard and paste it in any text application. This is useful e. g. if you want to contact EB support because of an error message.

To do so, select the Error Log entry you want to copy and use either keyboard shortcut **Ctrl+C** or right-click and select **Copy** in the context menu. A text representation (including more information than you can see in the GUI) of the Error Log entry is now inside the Windows clipboard and can be pasted into other applications.

5.3.6. Problems view





While you edit your configuration, EB tresos Studio constantly checks the configuration for configuration errors and warnings in the background. This background operation is called the *verifier* thread. If a problem in the configuration is found, the verifier adds an entry to the **Problems** view.

The items in the **Problems** view are grouped by the project names and by the type of problem. Possible types are *error*, *warning* and *info*.

To display all levels of errors and warnings available:

- ▶ Open up the top-level tree of the **Problems** view.

The number in brackets behind the group entries indicates the number of messages in the respective category.

Number	Information type	Explanation
1	Message	Indicates the message text. The message can be either a warning or an error.
2	Error Code	Indicates the error number, which is used by EB tresos Studio developers to identify the error in the software code.
3	Path	Indicates the XPath term in the configuration model to the parameter that issued a warning or error.

To edit the parameter that has issued a message:

- ▶ Single-click on the entry in the **Problems** view.

The editor of the respective parameter opens up directly at the position where the problem is and you may edit the parameter.

To find out more detailed information about the entry:

- ▶ Double-click the entry in the **Problems** view.

The Problems View Entry Details dialog opens up.

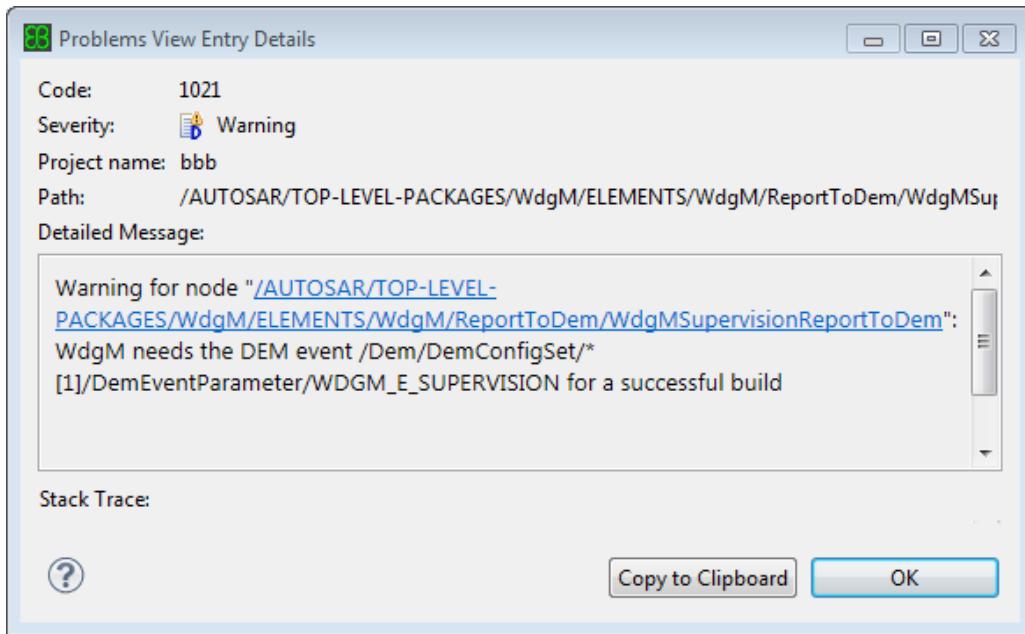


Figure 5.20. Details of the **Problems View** view

Additionally to the **Problems** view entry this dialog shows:

- ▶ The project name in which the message occurred.
- ▶ The detailed message text. Links may be available and will navigate you directly to the point of interest. You can select and copy the message.

5.3.6.1. Changing the display settings of the Problems View

To display the message hierarchy of all log entries into messages and submessages:

- ▶ Click the **Expand all** button

The view of all messages is expanded and all messages with submessages are displayed.

To show just the top-level messages instead of all submessages:

- ▶ Click the **Collapse all** button

The message view collapses and displays just top-level messages.

5.3.6.2. Export the messages of the Problems View

To write a message to a text file:



- Click the **Export log to a file** button .

A dialog window opens up, which enables you to save the message to your computer. This may be useful, e.g. if you want to contact EB tresos Studio support with a problem connected with the message.

5.3.6.3. Filtering Problems view messages by their severity level

You may filter out messages depending on their severity level.

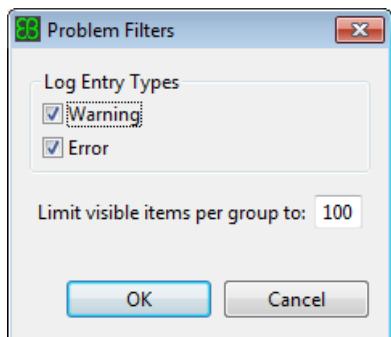
- Click the **View** button .

A context menu opens up.



- Select **Filter Entries**.

The **Problem Filters** window opens up.



- Select the **Log Entry Types** you want the **Problems** view to display.
► Click **OK**.

The **Problems** view displays only the types that you selected in the **Problem Filters** window.

5.3.6.4. Limit the number of entries shown in the Problems View

If you have a many entries in the **Problems** view and experience the GUI slowing down, you can restrict the number of entries shown in the **Problems** to an upper bound:

- Click the **View** button .

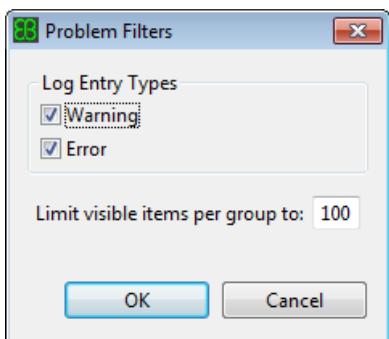


A context menu opens up.



- ▶ Select **Filter Entries...**.

The **Problem Filters** window opens up.



- ▶ Enter a positive number in the entry **Limit visible items per group to**.
- ▶ Click **OK**.

The **Problems** view restricts the number of shown entries.

5.3.7. Workflows view

The **Workflows** view displays one workflow at a time. A workflow consists of instructions that help you to accomplish a certain task. The steps to proceed are structured as a tree, through which you may go and perform each single step.

The **Workflows** view consists of three parts:

tool bar

The tool bar provides buttons to perform the actions of the workflow.

step area

The **step area** displays the workflow structured as a tree. The step area may contain action steps and group steps. For further information on action steps and group steps, see [Section 5.3.7.3, “Performing steps of a workflow”](#).

description area

The **description area** describes the step currently selected.

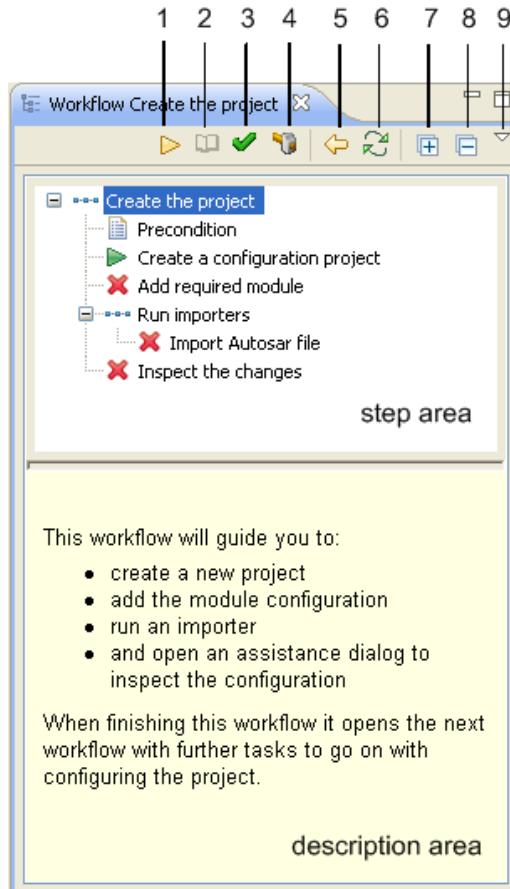


Figure 5.21. Workflows view

In the tool bar of the **Workflows** view, the following buttons are available:

Number	Action	Description
1	Run/Jump	<p>This action shows different icons depending on the current selection in the workflow tree:</p> <p>▶ Jump button to jump to the next step. This action is shown for all group steps and for all unavailable action steps (marked with)</p> <p>▶ Run button to execute the selected operation. This action is shown for all available action steps.</p> <p>If you click the Run button, the default action of the selected step is performed and the next step is selected. If the action is not repeatable, it is marked as finished afterwards. If the step currently selected only has a description or is a group step, this action only selects the next step.</p>



Number	Action	Description
2	Configure	The Configure button is only enabled if the currently selected step is an action step with a configuration command. Clicking the Configure button executes the configuration command associated with this step.
3	Finish	The Finish button marks the step currently selected as finished. If a group step is selected, all sub-steps are marked as finished, too. If a step is repeatable, the only way to mark it as finished is to click the Finish button.
4	To do	If you cannot perform a step right away, you may mark this step with a To do marker. You can thus see which steps still need to be performed so that you won't forget them.
5	Back	The Back button allows you to go back to the last step performed. Reverting the last step performed is only possible if you had used the tool bar buttons to perform the steps. If you use the cursor keys to select steps, you cannot use the Back button to revert the selection change.
6	Reset	With the Reset button, you may reset all step states of the complete workflow tree.
7	Expand all	Expands all nodes of the workflow tree.
8	Collapse all	Collapses all nodes of the workflow tree.
9	Workflow selection menu	In the workflow selection menu, you can select the last three workflows used. You may also open the workflow selection dialog to choose a workflow from the list of all registered workflows.

5.3.7.1. Selecting a workflow

In the **workflow selection** menu, you can select one of the last three workflows used or open a dialog in which all registered workflows are displayed.

To select one of the last three workflows used:

- ▶ Click the **View** button  in the tool bar.

The **workflow selection** menu opens up.

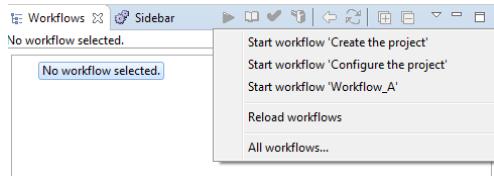


Figure 5.22. Opening the workflow selection menu

- ▶ Select one of the listed workflows.

The selected workflow is opened in the **Workflows** view.

To open a workflow from the list of all registered workflows:

- ▶ In the **workflow selection** menu, select the item **All workflows....**

The **Select a workflow** dialog opens up.

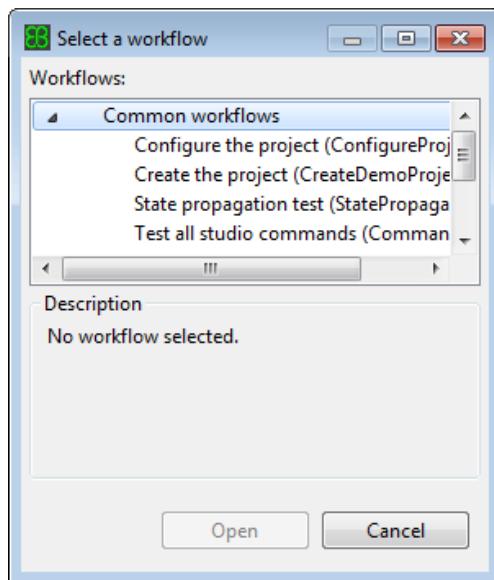


Figure 5.23. The **Select a workflow** dialog

- ▶ Select the workflow you would like to open in the **Workflows** view. Note that only one workflow can be selected at a time.

The selected workflow is displayed in the **Workflows** view.

5.3.7.2. Reload workflows action

If you select the preference **Show action 'Reload workflows' and show workflow IDs in Workflows view**, you can see the **Reload workflows** action in the **Workflow** view selection menu.

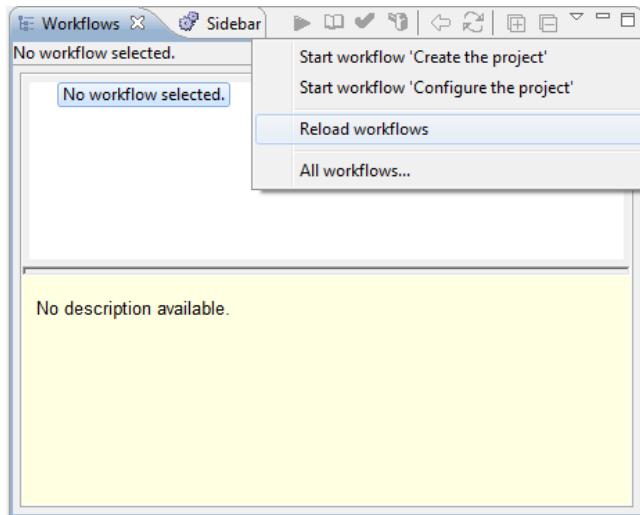


Figure 5.24. The **Reload Workflows** action

The action reloads all existing workflow files:

- ▶ project-specific workflow files in the `.workflows` folder of a project
- ▶ common workflows registered via `plugin.xml` of an installed plugin

It is also possible to make new project-specific workflows available in the **Workflows** view or to remove them from it. Just adding or removing workflow files to/from the `.workflows` folder of a project and selecting the **Reload workflows** action. Adding or removing commonly registered workflow files requires a restart of EB tresos Studio.

5.3.7.3. Performing steps of a workflow

The **step area** of the **Workflows** view displays the steps that are necessary to complete the task. These steps may be either *group* steps or *action* steps. Group steps are steps that contain other steps, action steps are steps that contain commands that can be performed when you process the workflow.

The group steps are displayed as follows:

Icon	Description
	This icon represents a group step of the type <i>choice</i> . This means that only one of the steps must be performed to finish the task.
	This icon represents a group step of the type <i>sequence</i> . This means that all steps must be executed successively to finish the task.

The action steps are displayed as follows:



Icon	Description
	This icon represents an action step whose command can be performed either by clicking the Run button or by double-clicking the step itself.
	This icon represents an action step that provides only a description.

The state of each step is displayed with the following icons and overlay icons:

Icon	Description
	This icon is displayed if the step is not available for the current selection in the Project Explorer view.
	You can mark each step with a To do marker by using the tool bar button To do . The selected step and all steps on the path upwards to the root node of the workflow tree are then marked as <i>To do</i> . This helps you to remember steps that haven't been completed yet, but must not be forgotten.
	This overlay icon informs you that this step already has been finished. If you go through the workflow tree with the Run button, a finished step will be skipped.
	This overlay icon is displayed if the step is a repeatable step. Repeatable steps are not automatically marked as finished after running the step. The only way to mark a repeatable step as finished is to click the Finish button in the tool bar.

5.3.8. Sidebar view

The **Sidebar** view displays assistance dialogs available for the elements currently selected, e.g. the CAN Buffer Assignment Editor and the Frlf Joblist Editor. The content of the **Sidebar** view is split into categories, e.g. **OS**, **Debugging**. These categories may be organized in a hierarchical manner, which is displayed as a tree in the **Sidebar** view. Which assistance dialogs are available depend on the context, i.e. on the opened editors, the selected items in the project explorer, the module configurations of your project etc.

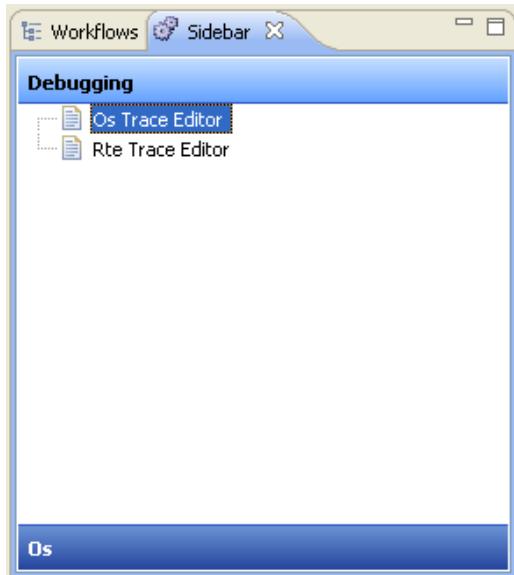


Figure 5.25. The **Sidebar** view

You can open these assistance dialogs by double-clicking them in the **Sidebar** view. You find detailed information on the assistance dialogs in the documentation of the respective modules.

5.3.9. Properties view

The **Properties** view is a standard **Eclipse** view. For further information on the **Properties** view, see the [Eclipse Workbench User Guide](#), which is also included in the EB tresos Studio on-screen help.

When you select a parameter in the Generic Editor, the **Properties** view displays the following tabs:

- ▶ **Information**
- ▶ **Description**
- ▶ **Problems**
- ▶ **PostBuildVariantConditions**

These tabs display information about the selected parameter from the **Editor** view. The title of the **Properties** view contains the name of the currently selected parameter.

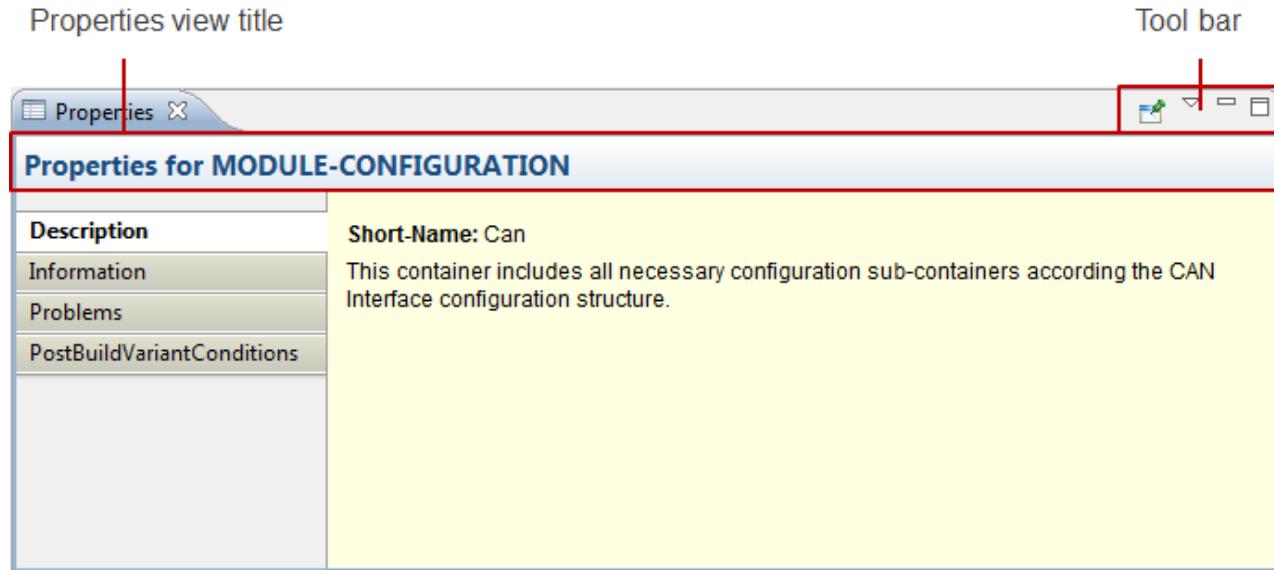


Figure 5.26. The **Properties** view

You can pin the **Properties** view to the current selection. To do this, click the **Pin Properties view** button  in the **Properties** view tool bar. If you select another parameter in the generic editor, the **Properties** view is not affected.

You can also display multiple **Properties** views. To create a new **Properties** view, click the **View** button  and then select **New Properties View**. If you use multiple **Properties** views you can display different tabs for the currently selected parameter at the same time.

In combination with pinning the view, you can display information on different tabs for several parameters.



5.3.9.1. Information tab

The screenshot shows the 'Properties for MODULE-CONFIGURATION' dialog with the 'Information' tab selected. The table contains the following data:

Properties for MODULE-CONFIGURATION		
	General	
Information	Name	MODULE-CONFIGURATION
Description	Type	Container
Problems	Source	recommended configuration
	SHORT-NAME	Com
	LONG-NAME	-
	UUID	ECUC:2e6263f8-c461-4abc-ba24-cd51feab06dd
	Paths	
	XPath	/AUTOSAR/TOP-LEVEL-PACKAGES/Com/ELEMENTS/Com
	ASPath	/Com/Com
	Parameter Definition	/TS_TxDxM6I1R0/Com
	Path in Template	as:modconf('Com')[1]

Figure 5.27. The **Information** tab

The **Information** tab displays parameter meta information of the currently selected parameter in the editor. The meta information is derived from the AUTOSAR parameter definition. This tab is divided in two sections **General** and **Paths**.

The **General** section contains the following information:

- ▶ **Name:** the name of the element
- ▶ **Type:** the element type (integer, floating point unit, Boolean, string, etc.)
- ▶ **Source:** the source of the parameter (e.g. **manually edited** or **default value**)
- ▶ **SHORT-NAME:** the AUTOSAR short name of the parameter
- ▶ **LONG-NAME:** the AUTOSAR long name of the parameter
- ▶ **Min:** the minimum number of entries in a table
- ▶ **Max:** the maximum number of entries in a table
- ▶ **Range:** the valid values for this parameter
- ▶ **ORIGIN:** the AUTOSAR origin of the parameter
- ▶ **UUID:** the AUTOSAR UUID of the parameter
- ▶ **Multiple Configuration:** displays **Yes** if the selected element is an AUTOSAR multiple configuration container
- ▶ **Config Class:** the AUTOSAR configuration class of the parameter
- ▶ **Config Variant:** the AUTOSAR configuration variant of the parameter
- ▶ **Category:** the AUTOSAR category of the parameter



- ▶ **Post Build:** if the AUTOSAR attribute POSTBUILDCHANGEABLE is set to true, **true** is displayed here

The **Paths** section contains information about:

- ▶ **XPath:** the XPath path expression via which it is possible to navigate to the parameter
- ▶ **ASPath:** the AUTOSAR path expression of the parameter. For reference parameters, the path that describes all referenceable nodes is shown here. This path is also shown if there is no referenceable node available (e.g. not yet configured). It may contain a prefix such as `ASPathDataOfSchema:`, which means, that all configured nodes for a specific schema-node are referenceable. In this case, the AUTOSAR short name path of the schema-node is the last part of this path.
- ▶ **Parameter Definition:** displays the path to the AUTOSAR parameter definition element
- ▶ **Path to referenceable nodes:** the path to the schema node whose data nodes are referenced by this reference
- ▶ **Path in Template:** When you reference a parameter within a code template of the template-based code generator, you can use the displayed path. The path is only displayed if you enable the **Show template path in Properties view** preference.

NOTE

Not all information is available for every configuration parameter. For example, the **ASPath** is omitted if the parameter is not referenceable by an AUTOSAR path.



5.3.9.1.1. Copying Information tab values to the clipboard

You can copy the value of an **Information** tab attribute as plain text to the Windows clipboard and paste it in any text application. To copy the value, right-click the desired attribute and select **Copy to clipboard**.

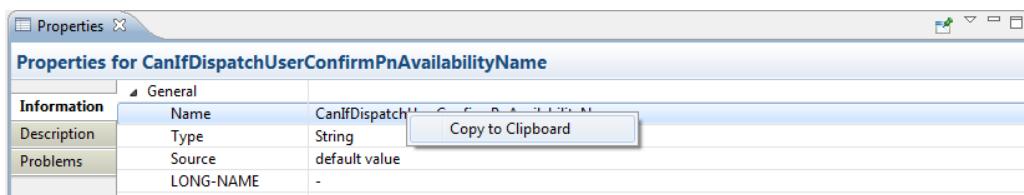


Figure 5.28. Copying values to the clipboard

5.3.9.2. Description tab

The **Description** tab describes the parameter selected in the editor, e.g. it gives configuration hints or lists references to other parameters. The description may contain hyperlinks to related parameters.

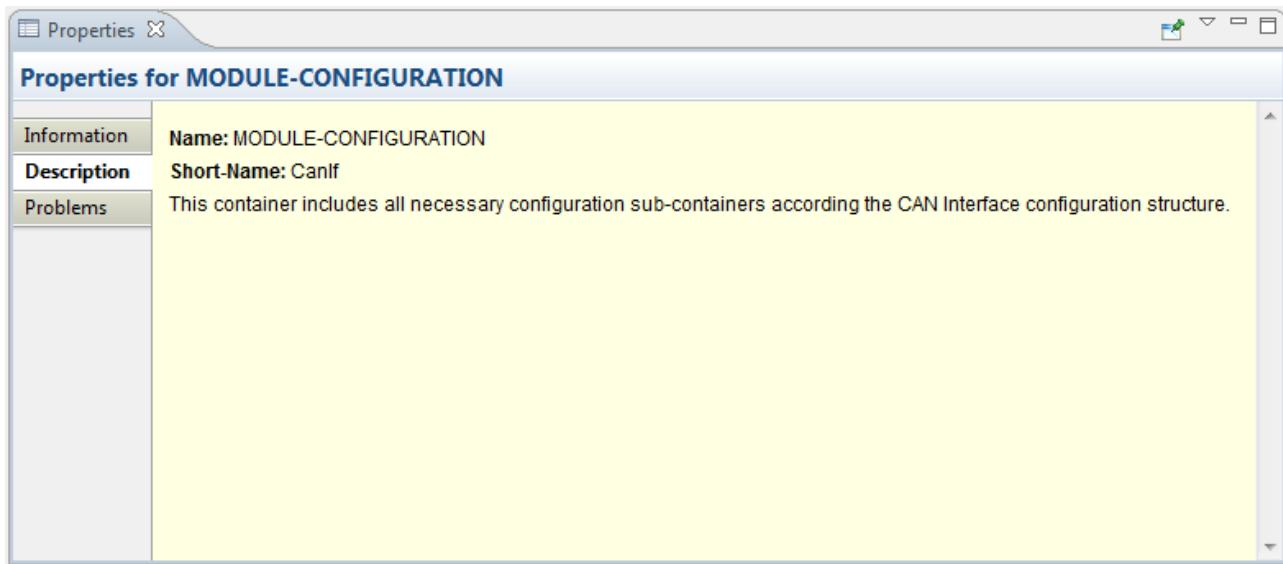


Figure 5.29. The **Description** tab

5.3.9.3. Comments tab

The **Comments** tab contains the comments associated with the selected node. You can use the **Comments** instead of the **Comments for <parameter name>** dialog. For further information on the dialog, see [Section 6.8.7, "Adding parameter comments"](#).

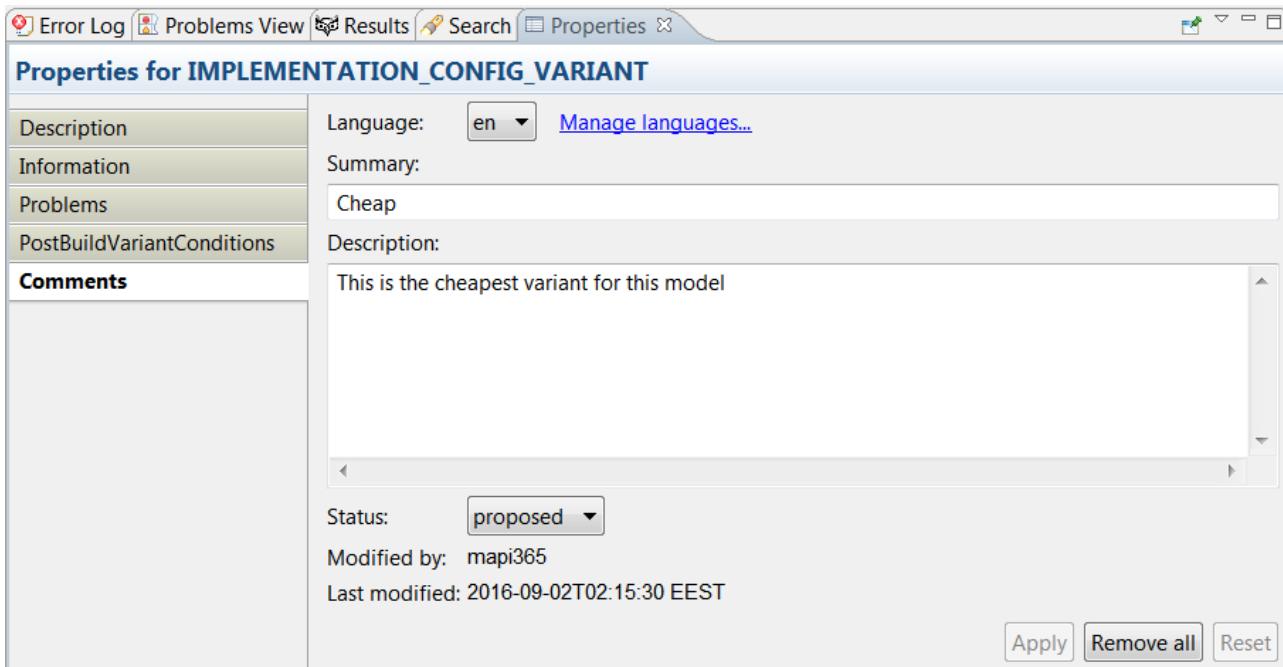


Figure 5.30. The **Comments** tab



The **Comments** tab has the same use as the **Comments for <parameter name>** dialog. Some differences are notable in the **Comments** tab:

- ▶ To add a comment, insert the summary and/or description and/or change the status for a selected language and then click **Apply**.
- ▶ To modify a comment, change the summary and/or description and/or change the status for a selected language and then click **Apply**.
- ▶ To reset a comment, click on **Reset** and the value currently stored in data model is loaded.
- ▶ To remove a comment, click on **Remove all**. All variants in all languages are removed.

5.3.9.4. Problems tab

The **Problems** tab displays all errors and warnings for the currently selected parameter in the editor. The problems are sorted by message types in descending relevance (errors > warnings).

The error code provided is a tool-internal code used by EB tresos Studio developers to locate an error in the source code of the tool.

Properties for CanIfDispatchUserConfirmPnAvailabilityName		
	Code	Messages
Information		
Description		
Problems		
	1019	Invalid value for node "/AUTOSAR/TOP-LEVEL-PACKAGES/CanIf/ELEMENTS/CanIf/CanIfDispatchCfg/CanIfDispatchUserCor
	1019	Invalid value for node "/AUTOSAR/TOP-LEVEL-PACKAGES/CanIf/ELEMENTS/CanIf/CanIfDispatchCfg/CanIfDispatchUserCor
	1019	Invalid value for node "/AUTOSAR/TOP-LEVEL-PACKAGES/CanIf/ELEMENTS/CanIf/CanIfDispatchCfg/CanIfDispatchUserCor
	1021	Warning for node "/AUTOSAR/TOP-LEVEL-PACKAGES/CanIf/ELEMENTS/CanIf/CanIfDispatchCfg/CanIfDispatchUserConfirm
	1021	Warning for node "/AUTOSAR/TOP-LEVEL-PACKAGES/CanIf/ELEMENTS/CanIf/CanIfDispatchCfg/CanIfDispatchUserConfirm
	1021	Warning for node "/AUTOSAR/TOP-LEVEL-PACKAGES/CanIf/ELEMENTS/CanIf/CanIfDispatchCfg/CanIfDispatchUserConfirm

Figure 5.31. The **Problems** tab

To see detailed message information:

- ▶ Double-click the entry in the **Problems** tab.

The **Message Details** dialog appears.

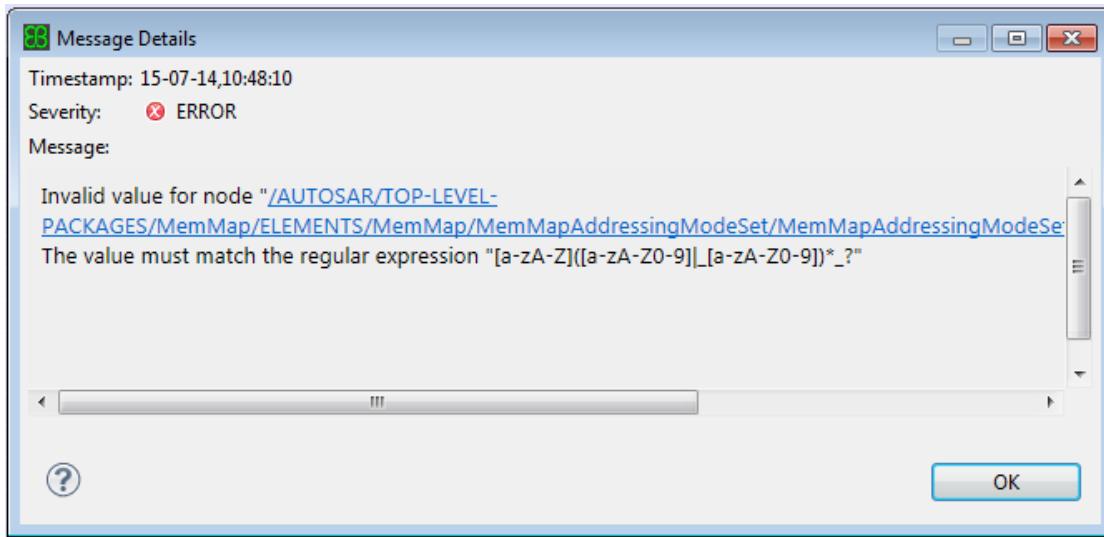


Figure 5.32. Details of the **Problems** tab

The dialog displays the following information:

- ▶ **Timestamp**: the time when the error is reported.
- ▶ **Severity**: the message is an error or a warning.
- ▶ **Message**: the message information. You can select and copy the message.

5.3.9.4.1. Copying error and warning messages to the clipboard

You can copy an error or warning from the **Problems** tab as plain text to the Windows clipboard and paste it in any text application. Right-click a row and select **Copy to clipboard**. The clipboard contains the code number in brackets and the complete message.

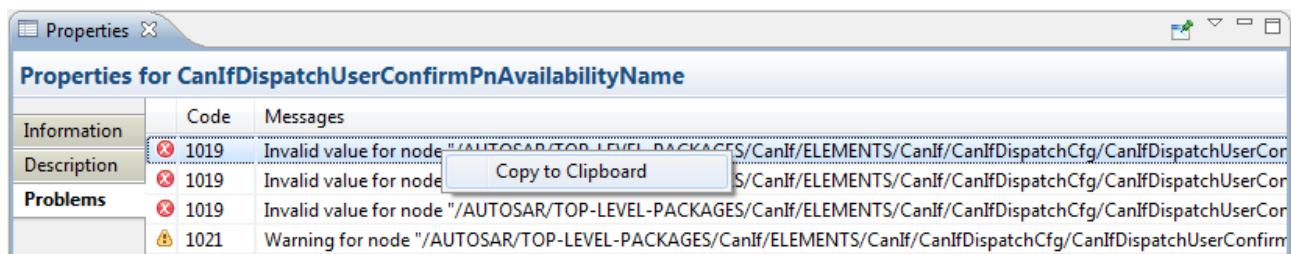


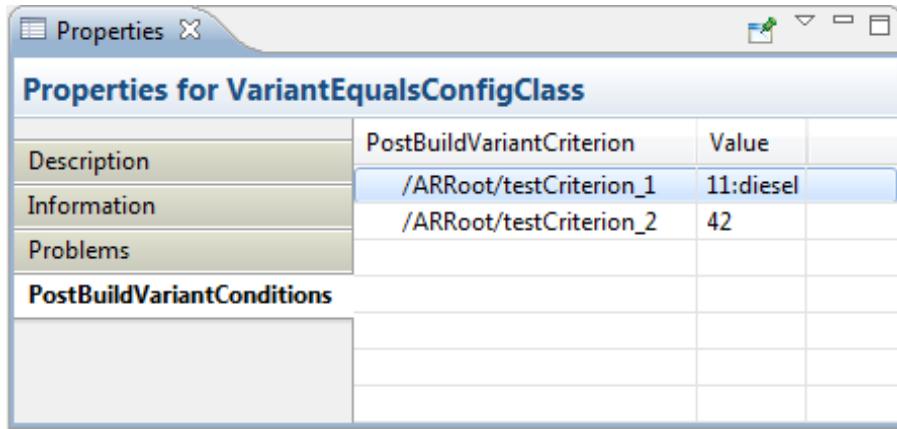
Figure 5.33. Copying error and warning messages to the clipboard

5.3.9.4.2. Adapting column width in Problems tab

You can change the width of each column inside the **Problems** tab. This setting is retained when you start EB tresos Studio the next time.

5.3.9.5. PostBuildVariantConditions tab

The **PostBuildVariantConditions** tab displays all Variant conditions for the currently selected parameter in the editor. If the selected parameter is not variant affected or does not have variant conditions, the tab stays empty.



The screenshot shows the 'Properties' dialog box for a 'VariantEqualsConfigClass'. The 'PostBuildVariantConditions' tab is selected. It contains a table with two rows:

Description	PostBuildVariantCriterion	Value
	/ARRoot/testCriterion_1	11:diesel
	/ARRoot/testCriterion_2	42

Figure 5.34. The **PostBuildVariantConditions** tab

If the system model contains textual descriptions for the criterion values, then those are shown in the column **Value** in addition to the integer value in the form **<integerValue>:<textual description>**

5.3.9.5.1. Copying PostBuildVariantConditions tab values to the clipboard

You can copy the value of an **PostBuildVariantConditions** tab conditions as plain text to the Windows clipboard and paste it in any text application. To copy the value, right-click the desired attribute and select **Copy to clipboard**.

5.3.9.6. CustomAttributes tab

The **CustomAttributes** tab displays all custom attributes for the currently selected parameter in the editor.

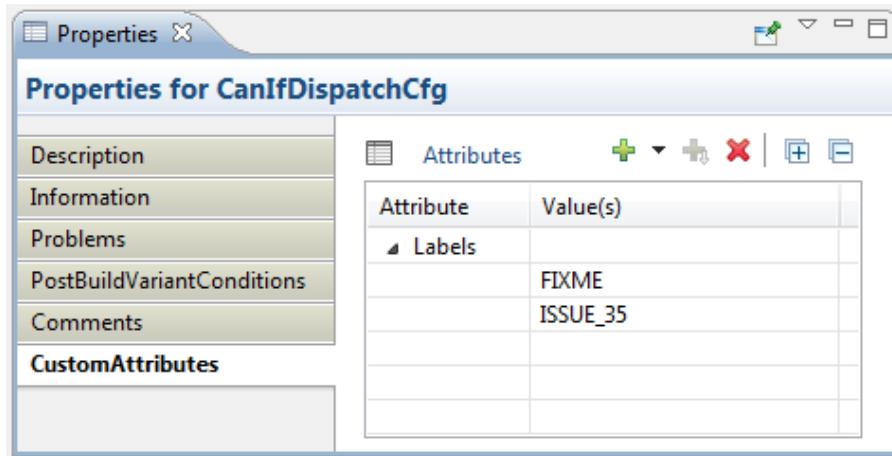


Figure 5.35. The **CustomAttributes** tab

5.3.9.6.1. Adding a custom attribute to a data node

To add a custom attribute to a data node, right-click the **CustomAttributes** table and select the desired custom attribute which you want to add. You can see the newly created custom attribute with a default value displayed in the table. You can also use the tool bar button to create a custom attribute in the same way.

5.3.9.6.2. Adding a value to a custom attribute

To add a value to a custom attribute, right-click the custom attribute and select the **Add value** action. You can see a newly created default value displayed in the custom attribute table. You can also use the tool bar button to create a value for a custom attribute.

5.3.9.6.3. Removing a custom attribute from a data node

To remove a custom attribute, right-click the custom attribute and select the **Remove** action. You can also use the tool bar button to remove a custom attribute.

5.3.9.6.4. Editing a custom attribute value

To edit a custom attribute value, click the cell which displays the value. A specific cell editor opens in which you can change the attribute value.



5.3.9.6.5. Copying the content of the CustomAttributes tab to the clipboard

You can copy custom attribute values of a data node as plain text to the Windows clipboard and paste it in any text application. To copy the value, right-click on the custom attribute table and select **Copy to clipboard**.

5.3.10. Results view

After running an unattended wizard or dialog, the results are displayed in the **Results** view.

The **Results** view contains several tab pages. The first tab, named **Overview** always exists and displays a summary of all wizard runs. For every wizard run, an additional tab will be created that shows the results of the wizard run.

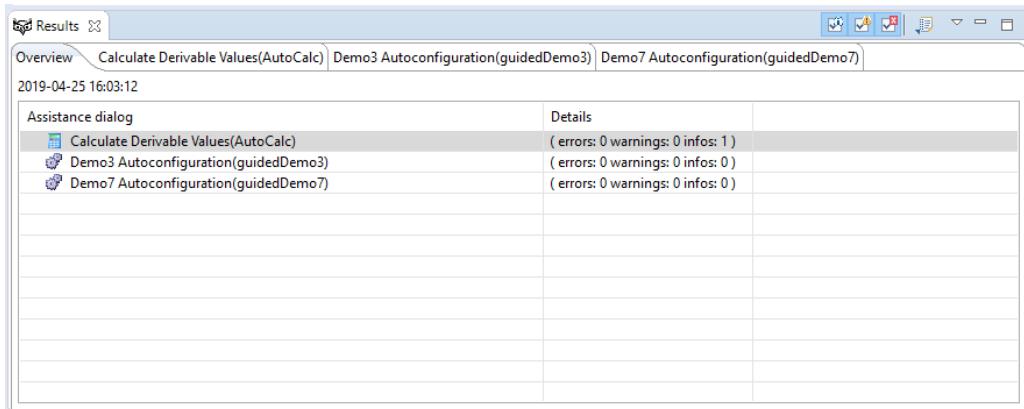


Figure 5.36. **Results** view overview

5.3.10.1. Filtering Error Log messages by their severity level

You can filter out messages depending on their severity level.

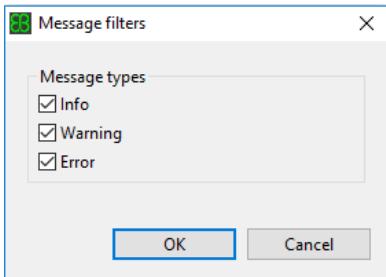
- ▶ Click the **View** button .

A context menu opens up.



- ▶ Select **Filter Entries**.

The **Message filters** window opens up.



- ▶ Select the **Message Types** you want the Results view to display.
- ▶ Click **OK**.

The Result view displays only messages that you selected in the **Message filters** window.

Alternatively, you can also use the **Filter** buttons in the view to filter the messages.

5.3.10.2. Export the messages of the Results view

To export the messages to a text file:

- ▶ Click the **Export results to a file** button

A dialog window opens up, which enables you to save the messages to your computer. This can be useful, e.g. if you want to contact EB tresos Studio support with a problem which is connected with the message.

5.4. Editors

5.4.1. The Generic Configuration Editor

The Generic Configuration Editor is the central tool for editing ECU configurations. The content of this editor is rendered out of the schema of the module to which a configuration belongs. The content below the header depends on what you are currently viewing. E.g. if you jump into a subcontainer, the editor will show the content of the subcontainer. You can use the **Node Outline** view to see your current position and to navigate inside the editor.

5.4.1.1. Sections and parameters

The following image shows you how parameters are displayed in the Generic Configuration Editor:

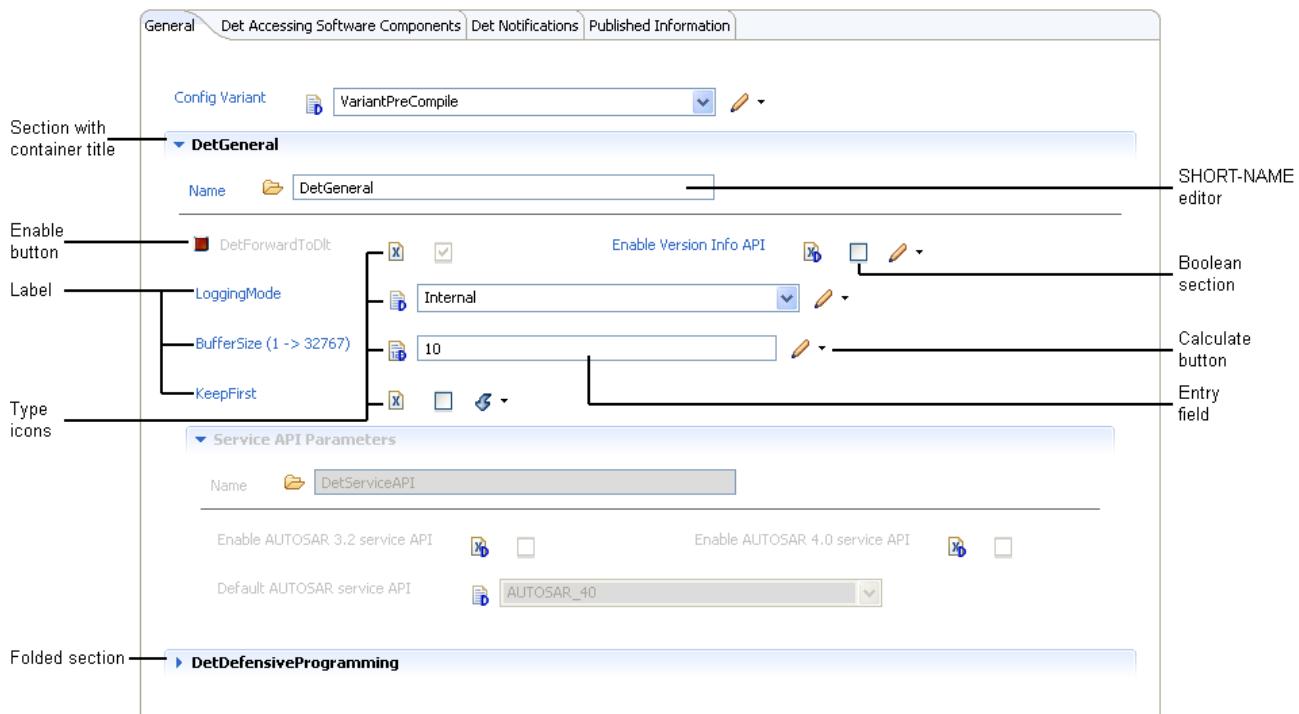


Figure 5.37. The GUI elements of the Generic Configuration Editor: sections, labels, type icons

Name	Explanation
Section with container title	Parameters and references in the General tab are sorted into sections. For AUTOSAR any section represents a CONTAINER that appears only once inside the element.
SHORT-NAME editor	In this entry field you can edit the SHORT-NAME of an AUTOSAR CONTAINER. The default value is the SHORT-NAME of the corresponding container in the AUTOSAR module definition.
Boolean section	In the Boolean section, you may edit the Boolean value of a parameter. <ul style="list-style-type: none"> ▶ To set the Boolean value to <code>true</code>, check the check box. ▶ To set the Boolean value to <code>false</code>, uncheck the check box.
Calculate button	Selecting the Calculate button initiates specified parameter calculations. The appearance of the icon depends on the current state of the parameter: <ul style="list-style-type: none"> ▶ <p>The current value of the element is not edited manually but comes from another source (e.g. is the default value or was imported). Clicking this button marks the element as manually edited. With this operation you accept the value. Operations such as running unattended wizards or importers that</p>



Name	Explanation
	<p>would normally overwrite the default, calculated or imported values do not change the element any more.</p> <ul style="list-style-type: none"> ▶ The small calculator icon indicates that the value of the element depends on the value of other elements and that you can let EB tresos Studio calculate its value. The current value of the element is not edited manually but comes from another source (e.g. is the default value or was imported). Clicking this button marks the element as manually edited. With this operation you accept the value. Operations like running unattended wizards or importers that would overwrite default, calculated or imported values do not change the element any more. ▶ The current value of the element was edited manually. Clicking the button recalculates the element value and marks it as a calculated default value. ▶ The current value of the element was edited manually. Clicking the button sets the value of the element to its default. ▶ The current value of the element was edited manually and there is no default value to set. Clicking the button clears the value of the element. ▶ The value of this integer element may be calculated automatically by the generator. If you click this button, a context menu opens up in which you may choose whether the value shall be calculated automatically or whether you want to calculate this value manually:
Entry field	



Name	Explanation
	<p>The entry field next to the type icon shows a parameter value you may change. The entry field is enabled or disabled, depending on the configuration class of the parameter and on the configuration time of the project. Depending on the parameter type, the entry field is displayed either as text box, as check box or as combo box:</p> <ul style="list-style-type: none">▶ Arbitrary text strings are displayed as text boxes or as combo boxes if they have a predefined value list. No context menu is provided.▶ Boolean values are displayed as check boxes.▶ Integers or floating point numbers are displayed as text boxes or - if the editor provides a predefined value list - as combo boxes. A context menu enables selecting the display formats:<ul style="list-style-type: none">▶ Autodisplay as <code>HEX</code>▶ Autodisplay as <code>BIN</code>▶ Autodisplay as <code>DEC</code>▶ Autodisplay as <code>OCT</code>The display format does not affect the format in which the number is stored in the configuration. To place separators (e.g. periods or spaces) into the displayed number, select Show separators from the context menu. You may enter integers in the following number formats:<ul style="list-style-type: none">▶ Hexadecimal digits: If the number starts with a <code>0x</code>, it is interpreted as a hex number.▶ Binary digits: If the number starts with a <code>b</code>, it is interpreted as binary number.▶ Octal digits: If the number starts with a <code>\0</code> it is interpreted as an octal number.▶ other digits: All other numbers are interpreted as decimal numbers.
Folded section	<p>For a better overview you can fold sections to temporarily hide their content.</p> <ul style="list-style-type: none">▶ To fold a section, click the container title. The section folds and the blue arrow before the container title becomes horizontal.▶ To unfold a folded section, click again the container title. The section unfolds and the blue arrow before the container title becomes vertical.



Name	Explanation
Type icons	<p>The type icon indicates the type of parameter. The following list explains the meaning of all type icons:</p> <ul style="list-style-type: none"> ▶ You may enter arbitrary text strings. ▶ You may enter integer or floating point numbers. ▶ You may enter Boolean values. <p>There are some overlay icons that might be displayed over the type icon. The following list explains the meaning of all overlay icons:</p> <ul style="list-style-type: none"> ▶ Indicates a wrong parameter. ▶ Indicates an editor warning. ▶ The element is currently set to its default value. ▶ The element value is calculated. ▶ The parameter is set to a preconfigured value. ▶ The parameter is set to a recommended configured value. ▶ The parameter is calculated automatically by the generator. ▶ The element value was imported. ▶ The parameter has a comment. <p>This parameter comment is displayed as a tool tip and shows a summary, status and the default language of the comment.</p> <ul style="list-style-type: none"> ▶ Depending on the parameter, the icons might host a context menu with further functionality.
Label	<p>The label describes the element. In the majority of cases it indicates the parameter name. In the case of an AUTOSAR module, the label indicates the SHORT-NAME. It might also be named differently, e.g. to indicate the functionality of the element.</p>

Name	Explanation
	NOTE Labels with a star indicate unsaved values  If the parameter contains an unsaved value, the label is displayed with a star and turns into darker blue.
Enable button	This button is for enabling or disabling optional elements. For details see Section 5.4.1.5, "Optional elements".

5.4.1.2. References

References are elements that refer to parameters. AUTOSAR distinguishes between REFERENCE-VALUE and INSTANCE-REFERENCE-VALUE:

- ▶ REFERENCE-VALUE (simple reference) references a single parameter with a reference path.
- ▶ INSTANCE-REFERENCE-VALUES (instance reference) need a target reference path and various context reference paths to reference a parameter.

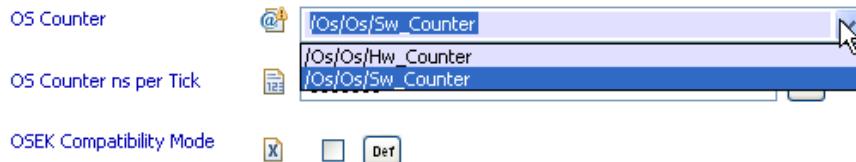


Figure 5.38. The GUI elements of the Editor view: references

In the entry field of a simple reference, you may enter:

- ▶ a path expression, or
- ▶ an AUTOSAR path, or
- ▶ select an AUTOSAR path from the paths determined in the parameter definition.

NOTE**A warning icon indicates an unverified path of the reference**

If the reference element is not part of the loaded configuration and thus EB tresos Studio cannot verify the path, the element is marked with a warning icon.

TIP**Navigating with context menus**

You may navigate to the editor of the referenced parameter via the context menu 'Go to referenced node' of the type icon. This is possible only if the parameter is part of the configuration project.

You can also navigate to the referenceable nodes e.g. to add some more entries to a list. This is possible via the context menu 'List referenceable nodes' of the type icon. If there is exactly one referenceable node, then the editor navigates directly to this node. If there are more referenceable nodes, then a list of nodes is shown in the Search view.

AUTOSAR instance references consist of a target reference and a list of context references.

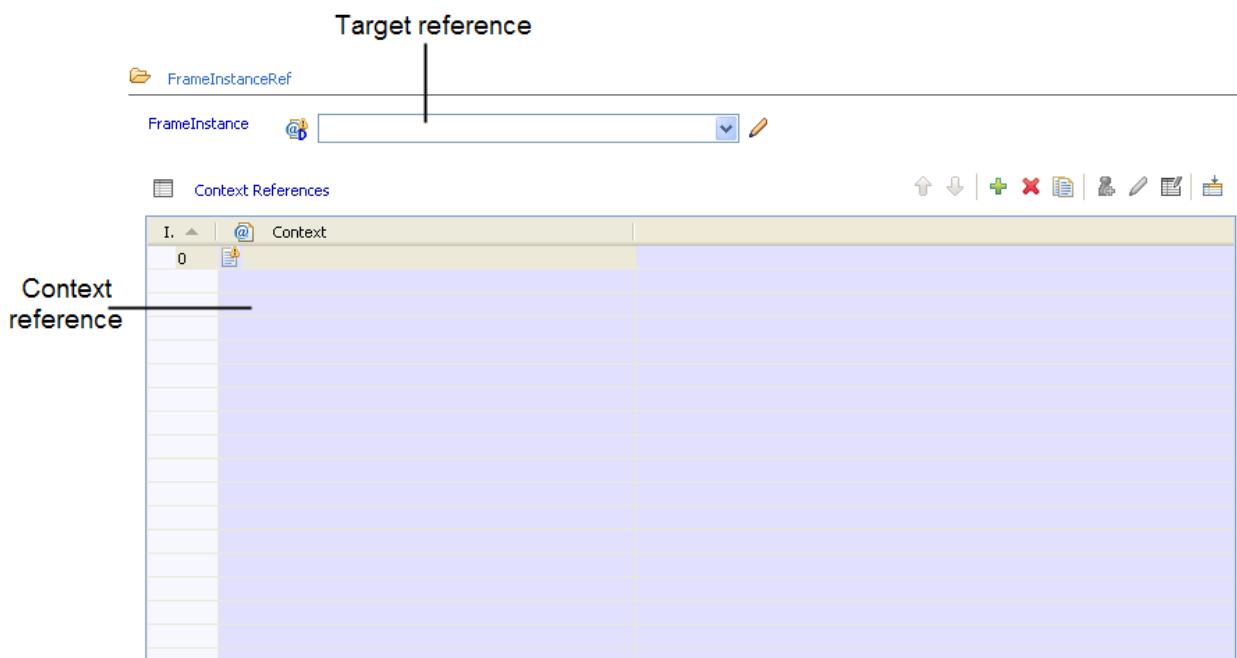


Figure 5.39. The GUI elements of the Editor view: instance references

In the editor, the target reference is represented by a drop-down list box and the context references are displayed as the elements of the table. In the drop-down list box you may select the target of the reference.

If the target cannot be uniquely referenced in your configuration, you need to specify the context of the target reference by adding one or more context references to the table. If valid selections are available in your configuration, you may select them in all drop-down list boxes of the instance reference.

5.4.1.3. Choices

AUTOSAR provides choices to select between different parameter sets. A choice in the AUTOSAR parameter definition model consists of a CHOICE-CONTAINER-DEF that includes PARAM-CONF-CONTAINER-DEFS.

In EB tresos Studio, a choice is displayed as a section of the Editor view. The section contains a drop-down list box, which represents the choice container CHOICE-CONTAINER-DEF, as well as the selectable subcontainers PARAM-CONF-CONTAINER-DEFS in separated tabs. The drop-down list box provides all available choices to select. EB tresos Studio enables the subcontainer of the currently selected choice and displays the title of its tab in bold font. The other subcontainers are disabled.

NOTE**Changing the selected choice**

If you change the selected choice in the drop-down list box, the subcontainer previously enabled is now disabled. The newly selected one is then enabled instead and its tab is brought to the front.

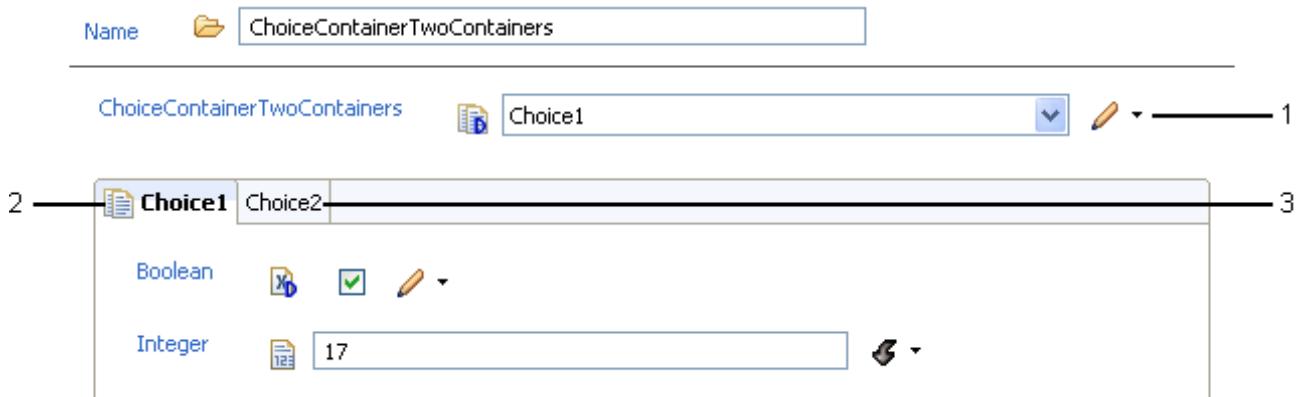


Figure 5.40. The GUI elements of the Editor view: choices

Name	Explanation
1	The drop-down list box, in which you may select the choice you want to activate. You can select only one choice at a time.
2	The subcontainer of the selected choice is enabled and its tab is initially brought to front. The tab's title is displayed in bold font.
3	EB tresos Studio provides the content of the other subcontainers in separated tabs.



TIP



Displaying subcontainers of not selected choices

To have a look at the subcontainer of a not selected choice without changing the selected choice in the drop-down list box, click on the title of a tab. The content of the disabled subcontainer is then displayed.

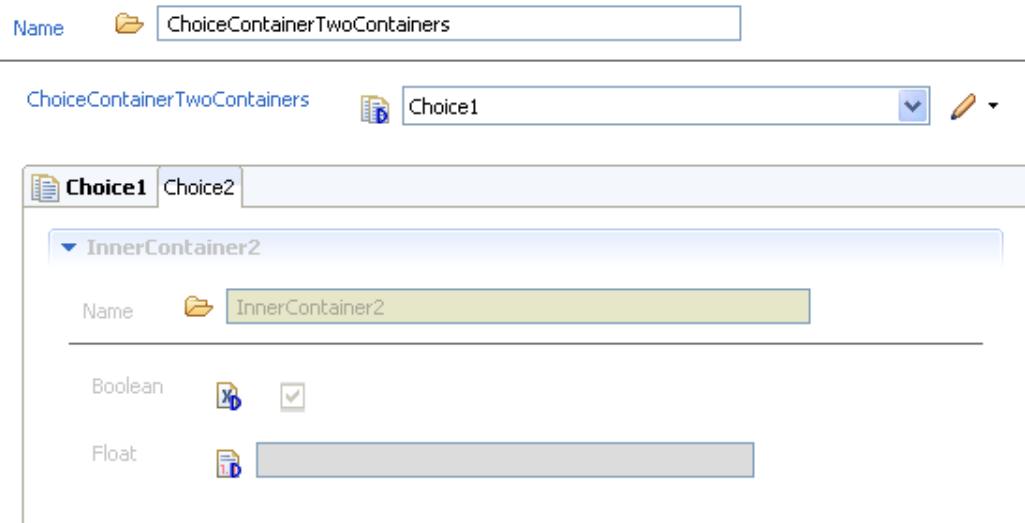


Figure 5.41. Viewing the content of a disabled choice

5.4.1.4. Tables

Tables represent all elements for the configuration of a user-selectable number. In AUTOSAR terms this means:
All elements that have a

- ▶ LOWER-MULTIPLICITY != UPPER-MULTIPLICITY , or
- ▶ LOWER-MULTIPLICITY == UPPER-MULTIPLICITY > 1.

How many instances of an element you need depends on the parameter and on the context you are using it in.

Parameters that must be displayed in a table are provided with their own tab.

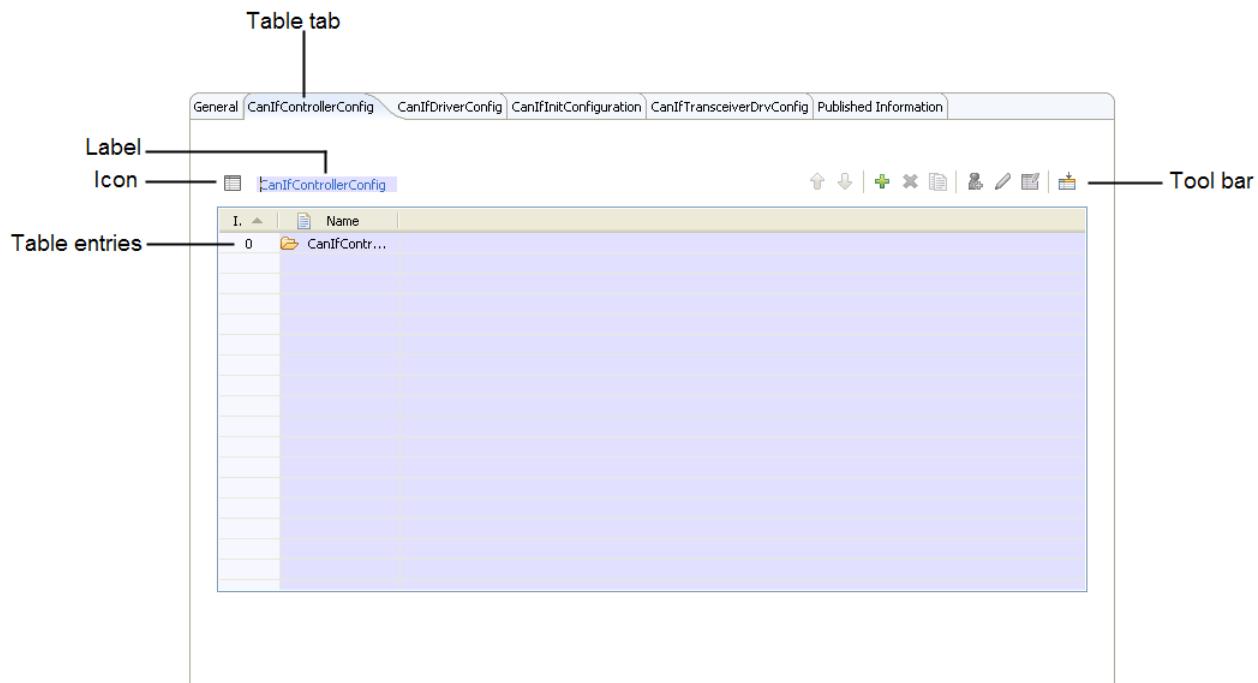


Figure 5.42. The GUI elements of the Editor view: tables

A table consists of:

- ▶ a label with an icon
- ▶ a list with table entries
- ▶ a tool bar

Name	Explanation
Table entries	<p>The layout of the table entries depends on the type of elements displayed. A table may display elements that group other elements, such as</p> <ul style="list-style-type: none"> ▶ AUTOSAR containers are displayed with an index column and a column in which the container name is editable. Additional columns may be available for parameters beyond the container. ▶ To navigate to the editor page in which the container content can be edited, double-click a cell in the index column. The tab of the container content opens up in the Editor view. ▶ To display information about the selected entry, click once on a cell in the index column. The information is displayed in the Element Outline view.



Name	Explanation
	<ul style="list-style-type: none"> ▶ Choices are displayed at least with an index column, a name column, and a choice column. ▶ To edit the content of the choice, double-click a cell in the index column. The tab of the choice content opens up in the Editor view. ▶ To display information about the selected entry, click once on a cell in the index column. The information is displayed in the Element Outline view. ▶ Parameters are displayed in two columns: The first is the index column, the second displays the current parameter value. The icon in the column header of the second column represents the entry type (e.g. reference). ▶ References are displayed in two columns: The first is the index column, the second displays the current reference value. The icon in the column header of the second column represents the entry type (e.g. reference).
Icon	<p>Shows an error or warning marker if there are too few or too many entries in the table.</p> <hr/> <p>TIP Displaying information about the table  Click on the icon to display information about the table itself in the Element Outline view.</p>
Label	<p>Displays the name of the list and indicates the saved status of entries.</p> <hr/> <p>NOTE Blue labels indicate unsaved values  If the list contains unsaved changes, the label turns blue.</p>
Table tab	Displays the configuration parameters in a table.
Tool bar	<p>Depending on the table, a tool bar with buttons is available. Unavailable buttons are grayed out. The following list explains the usage of the buttons:</p> <ul style="list-style-type: none"> ▶ Move up button  Click to move selected elements upwards in the list. ▶ Move down button  Click to move selected elements downwards in the list. ▶ Add element button  Click to add new elements with default values. ▶ Remove element button  Click to remove selected elements.



Name	Explanation
	<ul style="list-style-type: none">▶ Duplicate element button Click to duplicate selected elements. See also Section 5.4.1.4.1, "Duplicating elements".▶ Add required element button Click to add required elements.▶ Edit button Click to open row in a separate editor or window.▶ Bulk change dialog button Click to open the Bulk Change dialog, which allows to edit various rows simultaneously. This button is enabled if at least one row is selected.▶ Edit visibility button Click to edit visibility of columns.

To help increase navigation and editing speed, tables provide the following keyboard shortcuts:

Shortcut	Explanation
Arrow keys	Use to change the selected row.
Control+A	Selects all rows in the table.
Delete	Deletes the rows selected.
Insert	Inserts a new row at the end of the table.
F2	Starts editing the first editable column in the row selected.
▶ Tab	Switches between edited cells.
▶ Shift+Tab	
Space	Opens the editor of the container in the row selected.

TIP

To add a new entry to the table, double-click the unoccupied part of the table. A new entry is automatically inserted at the end of the list.



5.4.1.4.1. Duplicating elements

When you duplicate an element that has references inside which point inside the copied element, the duplicated references are updated to point to the duplicated reference-target inside the duplicated element.

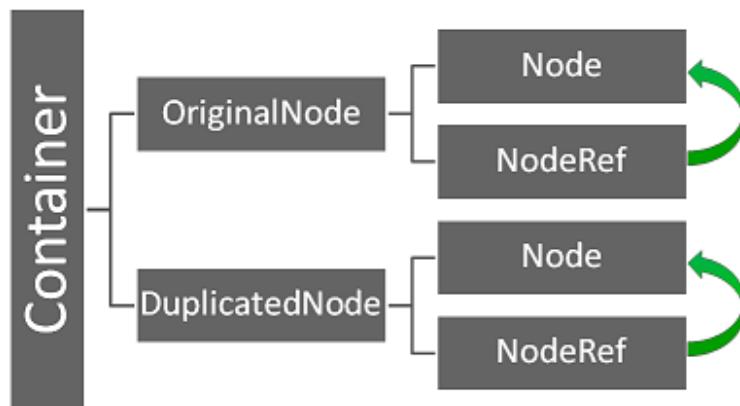


Figure 5.43. Duplicating elements with references

5.4.1.5. Optional elements

An AUTOSAR configuration contains optional configuration elements. You may decide if you want these configuration elements to be part of the configuration or not.

- ▶ To add the element to the configuration, click the toggle button when it is red:



Figure 5.44. The GUI elements of the Editor view: red toggle button

If you click the red toggle button, it turns green. The element is added to the configuration.

- ▶ To delete the element from the configuration, click this toggle button when it is green.



Figure 5.45. The GUI elements of the Editor view: green toggle button

If you click the green toggle button, it turns red. The element is no longer part of the configuration. But the configured values of the element (and its possible child elements) are saved. You may add it again at any time, because it is not permanently deleted.

If an element is switched off (not part of the configuration), the GUI of the element is grayed out.

5.5. Keyboard shortcuts

EB tresos Studio provides the following keyboard shortcuts which might help you speed up your work:



5.5.1. Shortcuts for the Workbench

Keyboard shortcut	Task
F1	Opens the EB tresos Studio online help.
Ctrl + N	Opens the New Project Wizard .
Ctrl + E	Opens a context menu in which you may switch between open editors.

5.5.2. Shortcuts for views

Keyboard shortcut	Task
Ctrl + F10	Opens a context menu in which you may customize the currently active view.
Ctrl + M	Maximizes the currently active view.

5.5.3. Shortcuts for editors and projects

Keyboard shortcut	Task
Ctrl + S	Saves the changes made in the currently open editor(s).
Ctrl + T	Opens the Search dialog in which you may search for configuration parameter names and their values within files or within the data model.
Ctrl + Z/Y	Undo/redo the last change in your editor.
Alt + left arrow/right arrow	The cursor jumps to the previous/next tab of the editor.

5.5.4. Shortcuts for tables

Keyboard shortcut	Task
Ctrl + A	Selects all rows in the currently active table.
F2	Moves the cursor to the first editable column in the row selected.
Space	Opens the editor of the container in the row selected.
Tab/ Shift + Tab	Moves the cursor to the next editable cell.
Insert	Inserts a new row below the last edited row of the table.
Delete	Deletes the selected row.



6. Use cases

6.1. Overview

The EB tresos Studio user's guide provides information on operating the tool via the graphical user interface and also from the command line.

If you have never worked with EB tresos Studio, we encourage you to start reading [Chapter 3, "Background information"](#) before proceeding to the other chapters.

If you have worked with EB tresos Studio and are familiar with the concepts or you just need to look up something, proceed to the task-oriented instructions.

- ▶ The chapter [Section 6.2, "Changing the EB tresos Studio setup"](#) describes how to get information about the current set-up of the tool and how to change the settings.
- ▶ If you want to create your first configuration project, or want to learn more about what you can do with projects, you should take a look at [Section 6.3, "Working with projects"](#).
- ▶ The chapter [Section 6.6, "Editing module configurations"](#) describes how you can create an ECU configuration for an AUTOSAR module and what you can do with it.
- ▶ The chapter [Section 6.7, "Upgrading projects and configurations"](#) covers upgrading to a newer AUTOSAR version, or to a new version of a module.
- ▶ All tasks around editing an ECU configuration are described in [Section 6.8, "Editing parameters of a module configuration"](#).
- ▶ How source code can be generated out of an ECU configuration is explained in [Section 6.9, "Generating code for projects"](#).
- ▶ The chapter [Section 6.10, "Importing and exporting"](#) provides information around the topic *importing and exporting configuration data*:
 - ▶ The chapter [Section 6.10.2, "Importing and exporting configuration data"](#) provides step-by-step instructions on how to import data in general.
 - ▶ The chapter [Section 6.10.2.2, "Importing and exporting configuration data from and to the AUTOSAR format"](#) shows how to import or export project configuration data in/from the AUTOSAR format.
 - ▶ The chapter [Section 6.10.2.6, "Importing CAN configurations from the DBC format"](#) shows how to import data from DBC file format and explains the mapping of the DBC information to AUTOSAR tags.
 - ▶ Accordingly, the chapter [Section 6.10.2.7, "Importing from the LDF format"](#) shows you how to import LDF data.
 - ▶ [Section 6.10.2.5, "Importing from the OIL format"](#) shows you how to import OIL files.



- ▶ EB tresos Studio also provides a command line mode, which is described in the chapter [Section 6.12, “Working with the command line”](#). The command line mode includes two sub-modes:
 - ▶ project mode for already existing code and
 - ▶ legacy mode for projects that have not been started yet.
- ▶ [Section 6.13, “Working with variants and Post-build loadable”](#) describes how variant handling is covered by EB tresos Studio and how to use it in your project.
- ▶ EB tresos Studio also supports [Section 6.14, “Post-build loadable configuration using multiple configuration containers”](#) prior to AUTOSAR 4.2.
- ▶ EB tresos Studio now provides the possibility to create a customer support package to make it easier for the support team to analyze problems. Refer to [Section 6.15, “Creating a customer support package”](#) on how to use this feature.

6.2. Changing the EB tresos Studio setup

This section describes how you can change several aspects of the EB tresos Studio setup, including finding information about installed modules and plugins and changing the program-wide preferences.

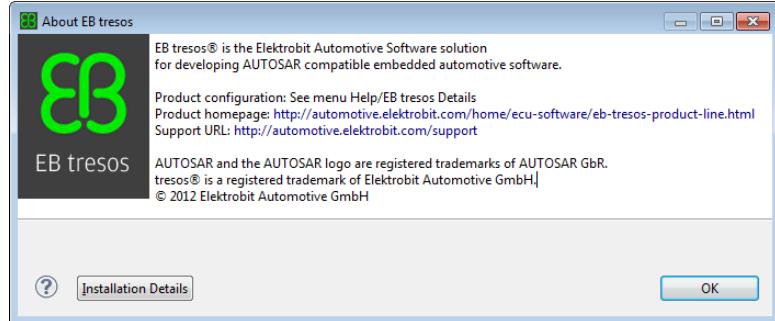
6.2.1. Viewing setup information

EB tresos Studio provides two dialogs which display information about the program and its setup: the **About EB tresos** dialog and the **EB tresos Details** dialog. This section will show you how these dialogs can be opened and will describe the information they contain.

6.2.1.1. Opening the About EB tresos dialog

To open the **About EB tresos** dialog:

- ▶ Select the **Help** menu.
 - A context menu opens up.
 - ▶ Select **About EB tresos** from the context menu.
- The **About EB tresos** dialog opens up.

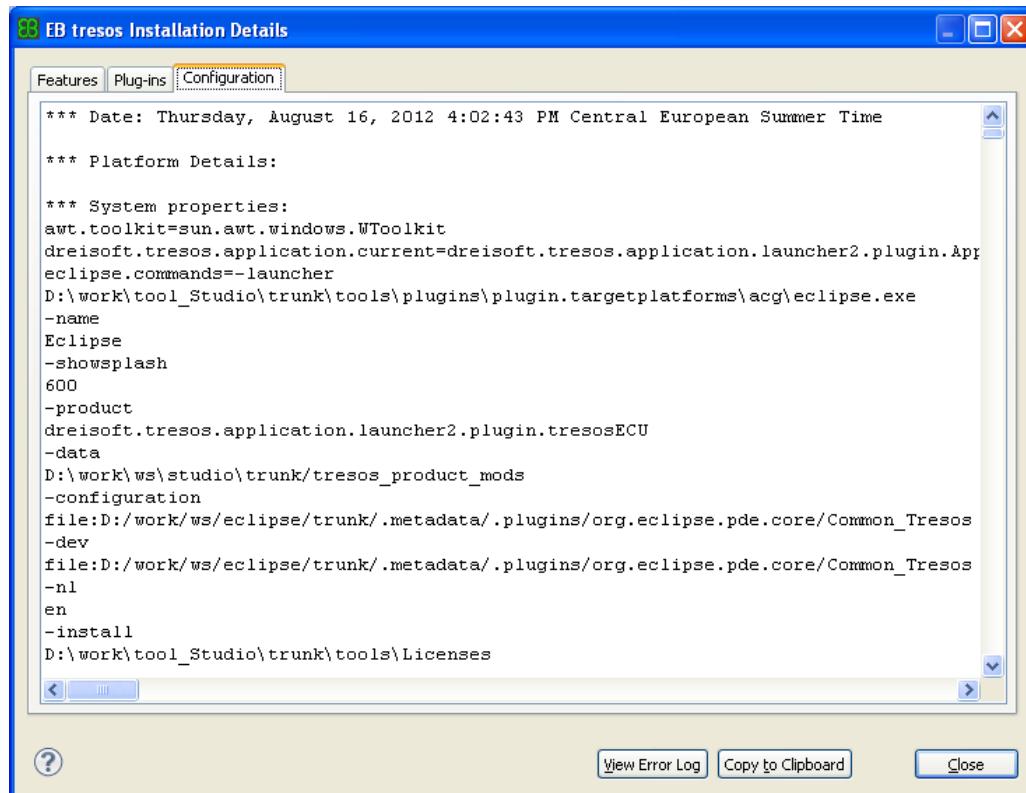


6.2.1.2. Finding program startup parameters and system property settings

To view internal information about how EB tresos Studio was launched and what internal system properties are set:

- ▶ In the **About EB tresos** window, click the **Installation Details** button.

The **EB tresos Installation Details** window opens up.



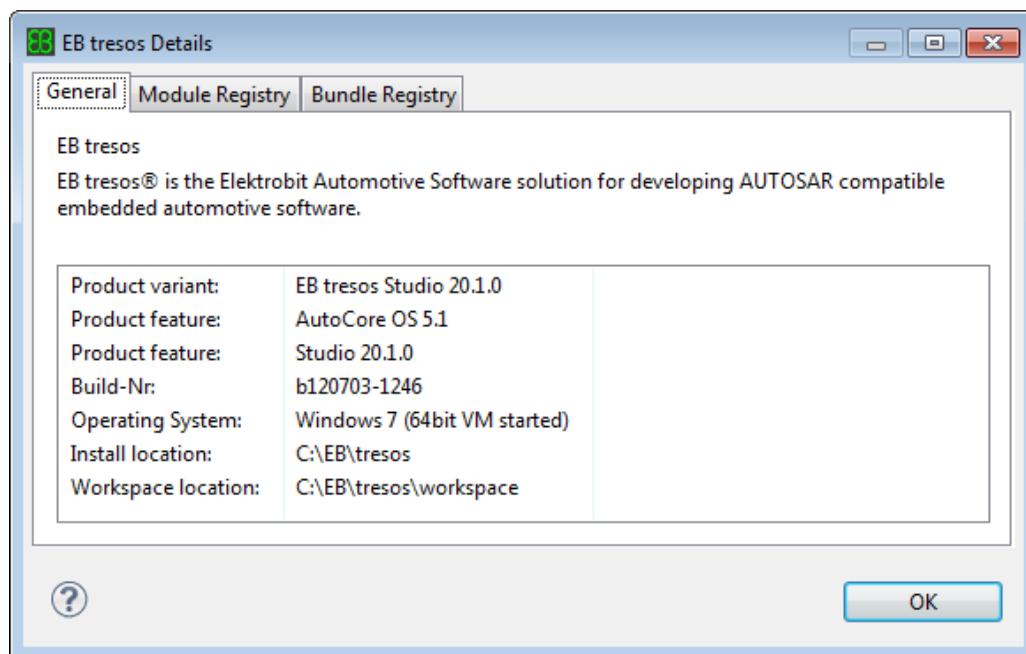
On the tab **Configuration**, this window provides detailed configuration information about program startup parameters and system property settings. In case of problems this information can be useful for EB support.



6.2.1.3. Opening the EB tresos Details dialog

To open the **EB tresos Details** dialog:

- ▶ Select the **Help** menu.
 - A context menu opens up.
 - ▶ Select **EB tresos Details...** from the context menu.
- The **EB tresos Details** window opens up.



6.2.1.4. Finding the location of the installation and the workspace

Select the **General** tab to display:

- ▶ The product variant and product features
- ▶ The build number
- ▶ The operating system
- ▶ The installation location on your computer
- ▶ The workspace location on your computer

6.2.1.5. Finding all installed modules



To find out if a specific module is part of your installation and which Autosar specification version it implements:

- ▶ Select the **Module Registry** tab.

The list below the tab displays all installed modules.

- ▶ Type the text you are looking for into the **Filter** text box.

The table in the **Module Registry** tab now shows just entries that contain text that matches your entered text. To view all entries again, just clear the text control by deleting its content.

I.	Id	Label	Type	Category	Layer	Component	SWVer	Sp
	NvM_TS_TxDxM6IOR0	NvM (V6.0.0, AS4.0.2)	NvM	Memory	Memory S...	ECUC	6.0.0	3.0
	Fls_TS_T0D1M4I1R0	Fls (V4.1.0, AS4.0.2)	Fls	Stub	Stub	ECUC	4.1.0	3.0
	Gpt_TS_T0D1M4I1R0	Gpt (V4.1.0, AS4.0.2)	Gpt	Stub	Stub	ECUC	4.1.0	3.0
	IpduM_TS_T0D1M4I1R0	IpduM (V4.1.0, AS4.0.2)	IpduM	Stub	Stub	ECUC	4.1.0	2.0
	Ts5Atl_TS_TxDxM2IOR0	Ts5Atl (V2.0.0, AS4.0.2)	Ts5Atl	Test	Service	ECUC	2.0.0	0.0
	WdgM_TS_T0D1M2IOR0	WdgM (V2.0.2, AS3.1.0)	WdgM	Stub	Stub	ECUC	2.0.2	1.0
	PduR_TS_T0D1M4I1R0	PduR (V4.1.0, AS4.0.2)	PduR	Stub	Stub	ECUC	4.1.0	3.0
	Cdd2_TS_T0D1M4I1R0	Cdd2 (V4.1.0, AS4.0.2)	Cdd2	Stub	Stub	ECUC	4.1.0	3.0
	Can_TS_T19D1M2IOR0	Can (V2.0.1, AS4.0.2)	Can	CAN	Communi...	ECUC	2.0.1	3.0
	FrNm_TS_TxDxM5IOR0	FrNm (V5.0.0, AS4.0.2)	FrNm	FlexRay	Communi...	ECUC	5.0.0	4.0
	IpduM_TS_TxDxM3IOR0	IpduM (V3.0.3, AS4.0.2)	IpduM	Com Ser...	Communi...	ECUC	3.0.3	2.0
	Wdg_TS_T0D1M2IOR0	Wdg (V2.0.2, AS3.1.0)	Wdg	Stub	Stub	ECUC	2.0.2	2.0

6.2.1.6. Finding all installed plug-ins

To find out if a specific plug-in is part of your EB tresos Studio installation and if it is working properly:

- ▶ Select the **Bundle Registry** tab.

The list below the tab displays all installed plug-ins.

- ▶ Type the text you are looking for into the **Filter** text box.

The table in the **Bundle Registry** tab now shows just entries that contain text that matches your entered text. To view all entries again, just clear the text control by deleting its content.



EB tresos Details

Module Registry					
State	SymbolicName	Bundle-Name	Bundle-Ven...	Bundle-Version	Bundle-Desc...
ACTIVE	org.eclipse.osgi	OSGi System Bundle	Eclipse.org ...	3.8.0.v20120529-1548	OSGi System
ACTIVE	org.eclipse.equinox.simple...	Simple Configurator	Eclipse.org ...	1.0.300.v20110815-1744	
ACTIVE	com.ibm.icu	International Compo...	IBM Corpor...	4.4.2.v20110823	
ACTIVE	dreisoft.tresos.application.l...	Launcher2 Applicatio...	Elektrobit A...	13.0.0.b120703-1246	
ACTIVE	dreisoft.tresos.autosar2.api...	tresos Studio autosar...	Elektrobit A...	13.0.0.b120703-1246	
ACTIVE	dreisoft.tresos.autosar2.plu...	tresos Studio autosar...	Elektrobit A...	13.0.0.b120703-1246	
ACTIVE	dreisoft.tresos.core.plugin	EB tresos Studio Core...	Elektrobit A...	14.1.0.b130607-1700	
ACTIVE	dreisoft.tresos.datamodel2....	tresos Studio datamo...	Elektrobit A...	13.0.0.b120703-1246	
ACTIVE	dreisoft.tresos.launcher2.ap...	tresos Studio launche...	Elektrobit A...	13.0.0.b120703-1246	
ACTIVE	dreisoft.tresos.lib2.api.plugin	EB tresos Studio lib2 ...	Elektrobit A...	13.0.0.b120815-1015	
ACTIVE	dreisoft.tresos.lib2.plugin	EB tresos Studio lib2	Elektrobit A...	13.0.0.b120815-1015	
ACTIVE	dreisoft.tresos.sysimporte...	Tresos Studio System...	Elektrobit A...	13.0.0.b120703-1246	

Filter:

OK

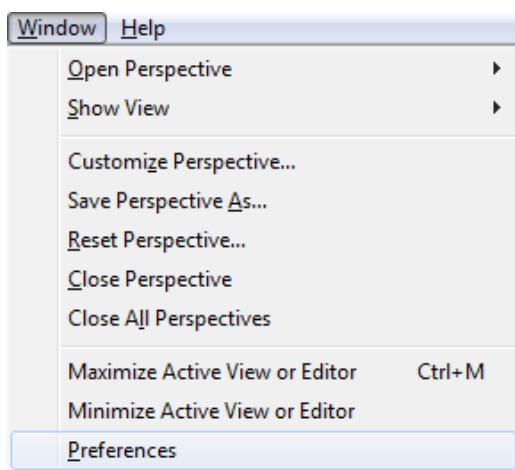
NOTE

This list shows all plugins from the Eclipse framework upon which EB tresos Studio is built as well as EB tresos Studio core libraries. If a plugin could not be loaded properly, the line is colored red.

6.2.2. Setting preferences

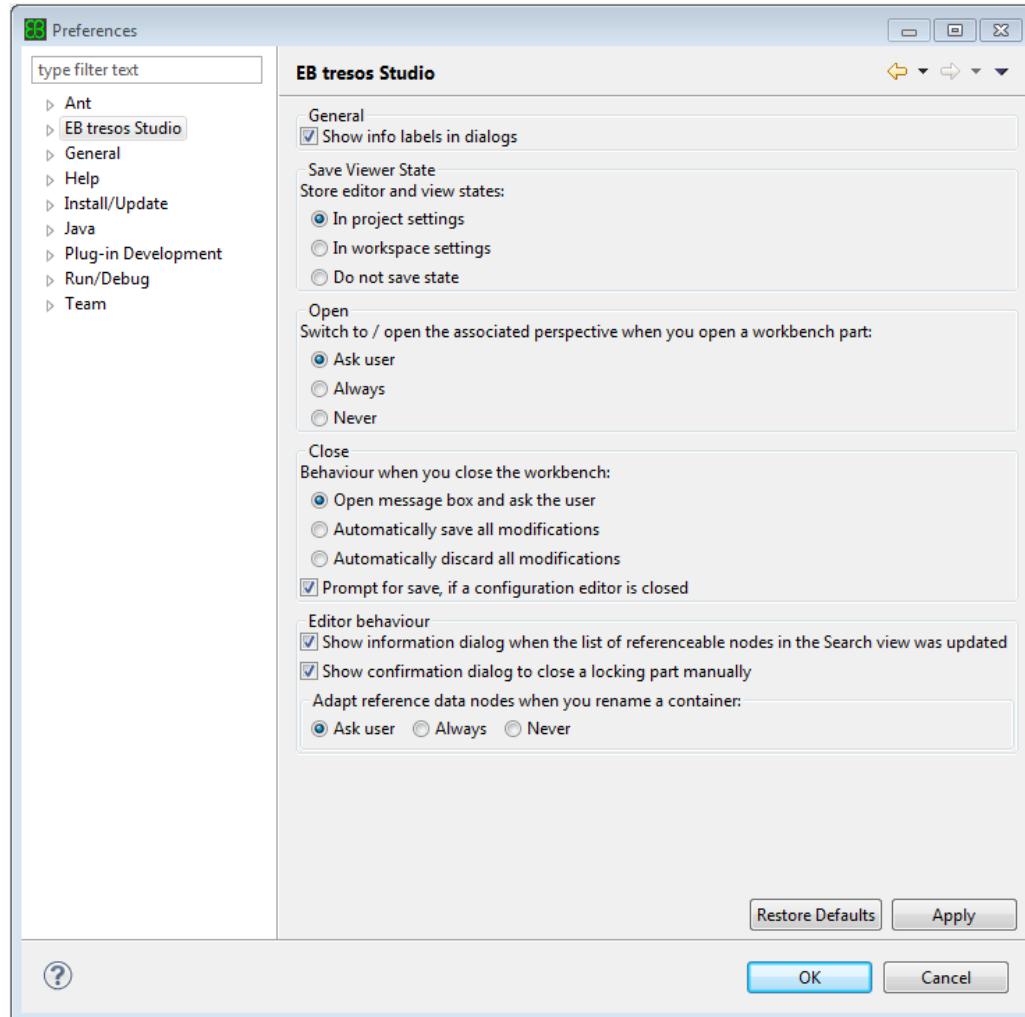
To globally store settings for a workspace set your preferences in the Window menu.

- ▶ Select **Window**.
- ▶ Select **Preferences** from the menu.



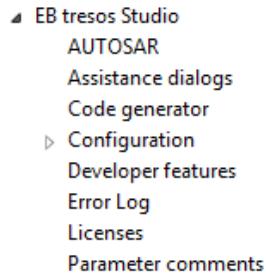


The *Preferences* window opens up.



- ▶ Select **EB tresos Studio** from the tree view on the left of the window.
- ▶ Double-click **EB tresos Studio**.

A subtree opens up underneath the tree item **EB tresos Studio**.



The following subsection will explain the preference pages related to items of the subtree. On any page, you can press the **Restore Defaults** button to restore the default values of the page. You can press the **Apply**

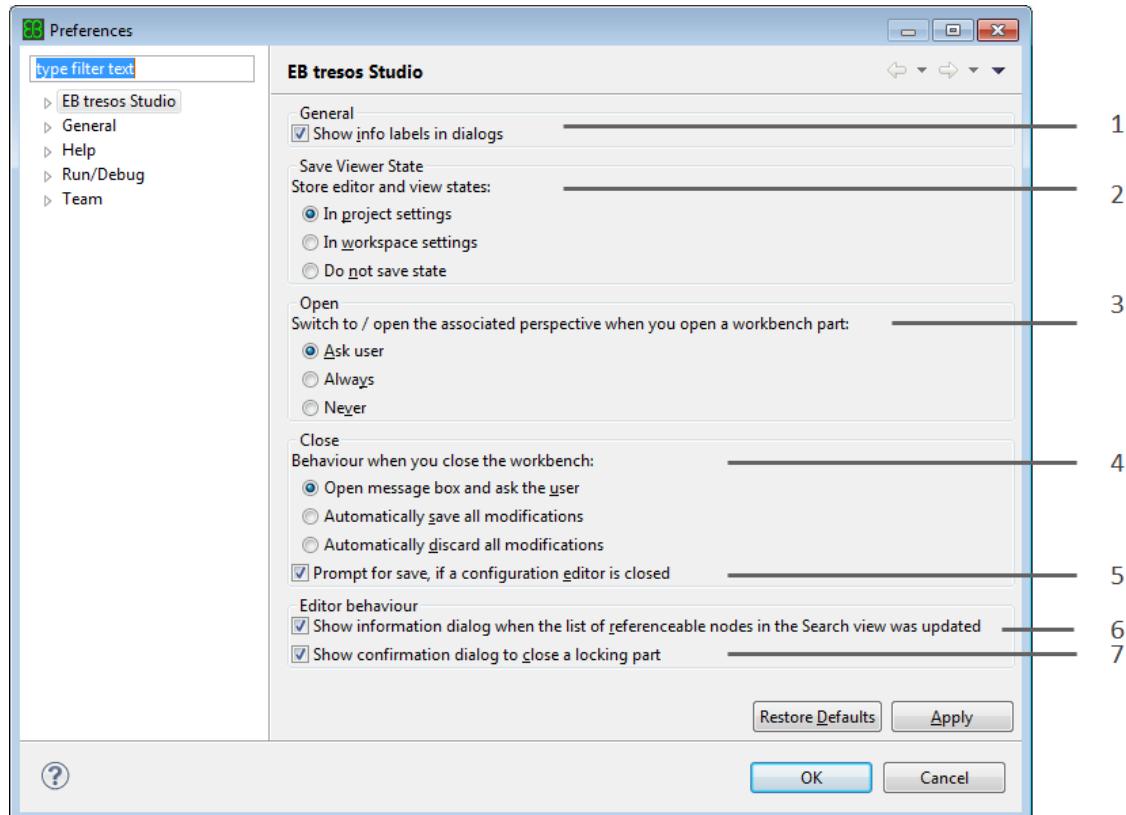


button to store the changes you made so far. To save the changes and close the complete dialog, click the **Ok** button. To cancel any changes you made and to close the dialog, click the **Cancel** button.

For more information on the other preference pages, see the [Eclipse Workbench User Guide](#). This Eclipse document is included in the EB tresos Studio onscreen help, which opens up when you press **F1**.

6.2.2.1. Setting general preferences

To set preferences about the general behavior of EB tresos Studio, use the following settings on the **Preferences EB tresos Studio** window:



Number	check box/Radiobutton	Instruction
1	Show info labels in dialogs	Select to enable help text for dialogs and wizards.
2	Store editor and view states:	<p>The configuration editors of EB tresos Studio are able to preserve their current state, e.g. columns shown in tables, column width, number format selected between editing sessions, by storing the state to disk when the editor is closed.</p> <p>The preference Store editor and view states allows to configure where this information is saved to.</p>



Number	check box/Radiobutton	Instruction
		<p>In project settings</p> <p>The editor state is saved inside the EB tresos Studio project the editor belongs to. The state is written to files in a directory <code>.settings</code> beneath the project directory of the project.</p> <p>In workspace settings</p> <p>The state is saved into the meta-data area of the workspace currently opened. The meta-data area is stored in a hidden directory called <code>.metadata</code> beneath the workspace root directory.</p> <p>Do not save state</p> <p>The editor does not write or load any state.</p>
3	Switch to / open the associated perspective when you open a workbench part	<p>When you open a workbench part such as a guided configuration editor for the first time, EB tresos Studio might try to switch to another perspective. This might not be intended by you and this behavior can be configured here.</p> <p>Ask user</p> <p>If a workbench part wants to switch to another perspective, a dialog is opened to let you decide to switch or stay in the current perspective.</p> <p>Always</p> <p>A workbench part switches to its preferred perspective as soon as you open it.</p> <p>Never</p> <p>Any workbench part opens in the currently active perspective.</p>
4	Behaviour when you close the workbench	<p>Open message box and ask the user</p> <p>Select to be asked every time you close the workbench whether you want to save changes. This will prevent you from closing the program with unsaved projects.</p> <p>Automatically save all modification</p> <p>Select if you want to save changes to your projects automatically upon closing. No dialog window will appear.</p> <p>Automatically discard all modifications</p> <p>Select if you do not want to save changes to your projects automatically and do not want to be reminded to save by a dialog window.</p>

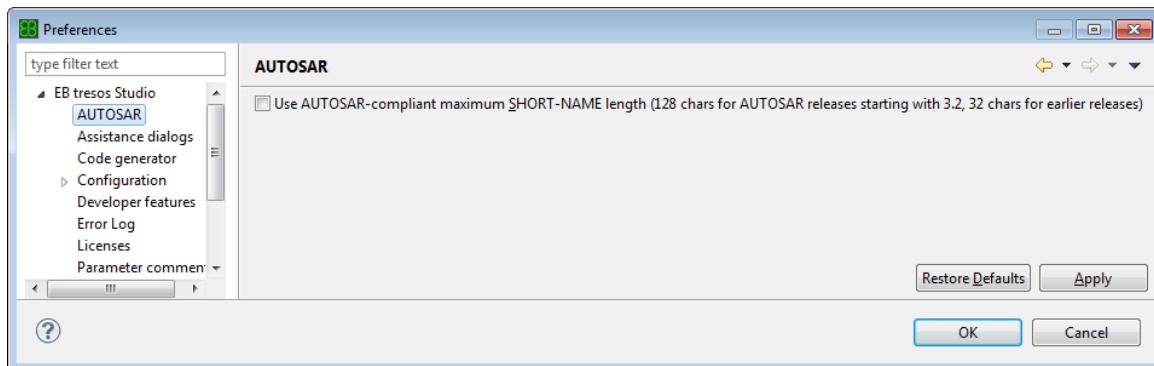


Number	check box/Radiobutton	Instruction
5	Prompt for save...	Select if you want the configuration editor to remind you to save your settings before closing.
6	Show information dialog when the list of referenceable nodes in the Search view was updated	You can select List referenceable nodes from the icon menu of a parameter within the generic editor. Select if you want an information dialog that the result is shown in the Search view.
7	Show confirmation dialog to close a locking part	If selected, a confirmation dialog is shown any time a function is chosen while a guided configuration dialog blocks the project.
8	Adapt references when you rename a container	<p>When you rename a container, you can decide to automatically adapt the references which refer to or into this container. You can configure and save this decision here.</p> <p>Ask user If you rename a container, a dialog opens to let you decide to adapt the references or not.</p> <p>Always If you rename a container, all references are adapted automatically.</p> <p>Never If you rename a container, no references are adapted automatically.</p>

6.2.2.2. Setting AUTOSAR preferences

To set AUTOSAR preferences:

- ▶ Select the **Preferences EB tresos Studio** subpage **AUTOSAR** and use the following settings:





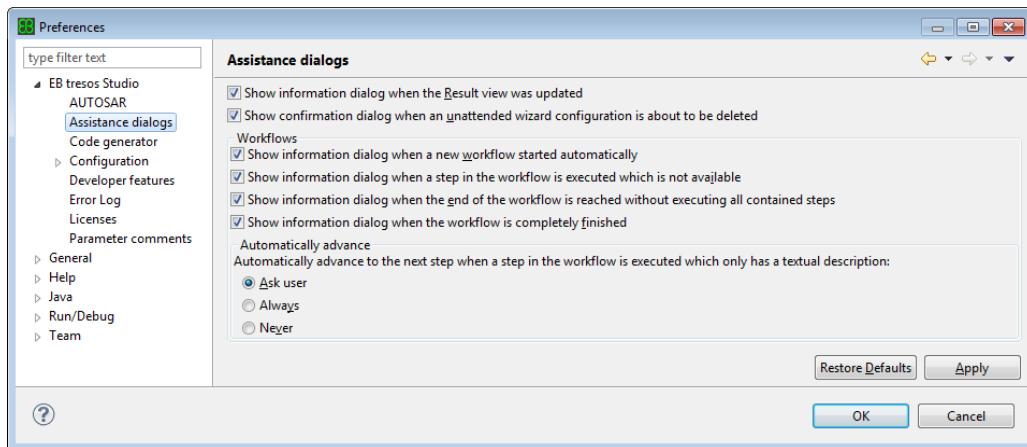
- ▶ Select **Use AUTOSAR-compliant maximum SHORT-NAME length (128 chars for AUTOSAR releases starting with 3.2, 32 chars for earlier releases)** to restrict the SHORT-NAME length to
 - ▶ 128 chars according to the AUTOSAR specification for AUTOSAR releases starting with 3.2
 - ▶ 32 chars for earlier AUTOSAR releases

If unchecked, longer names are allowed within EB tresos Studio. This also applies for the import and export of configuration data. Problems may occur if other tools try to import such exported configuration.

6.2.2.3. Setting Assistance dialog preferences

To set Assistance dialog preferences:

- ▶ select the **Preferences EB tresos Studio** subpage **Assistance dialogs** and use the following settings:



- ▶ Select **Show information dialog when the Result view was updated** to get informed about new results in the **Results** view.

If checked, an information dialog is shown each time after running unattended wizards or assistance dialogs to inform you about new results in the **Results** view.

- ▶ Select **Show confirmation dialog when an unattended wizard is about to be deleted** to decide whether you really want to delete the unattended wizard configuration.

If checked, a confirmation dialog is shown each time when deleting an unattended wizard configuration in the **Unattended Wizards** dialog.

- ▶ Select **Show information dialog when a new workflow started automatically** to get informed if the current workflow automatically started a new workflow.

If checked, an information dialog is shown if a new workflow has been started automatically in the **Workflow** view.



- ▶ Select **Show information dialog when a step in the workflow is executed which is not available** to get informed about unavailability of a workflow step.

If checked, an information dialog is shown when you try to execute a workflow step which is not available due to some reasons.

- ▶ Select **Show information dialog when the end of the workflow is reached without executing all contained steps** to get informed about skipped workflow steps which maybe still need to be performed.

If checked, an information dialog is shown when the end of the workflow is reached but some of the steps have not been executed yet (e.g. marked as TODO).

- ▶ Select **Show information dialog when the workflow is completely finished** to get informed when all steps have been performed.
- ▶ **Automatically advance to the next step when a step in the workflow is executed which only has a textual description:**

Enable **Ask user** if you want to be asked whether to directly navigate to the next step in the workflow or to stay on the current step to be able to perform the instructions in the textual description.

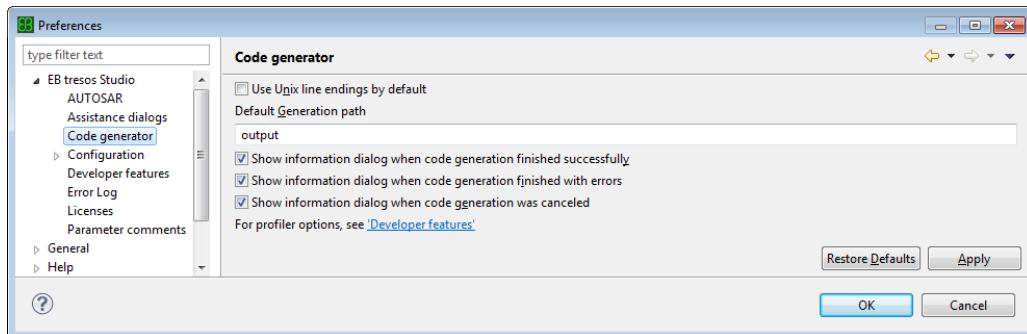
Enable **Always** if you want to always directly navigate to the next step.

Enable **Never** if you always want the remain on the current workflow step to be able to follow the instructions in the description area.

6.2.2.4. Setting Code generator preferences

To set Code generator preferences:

- ▶ select the **Preferences EB tresos Studio** subpage **Code generator** and use the following settings:



- ▶ Select **Use Unix line endings by default** to use Unix line endings in your generated code as default. You may override this workspace-wide default setting in the project preferences for every project individually. For more details, see [Section 6.3.7, “Viewing and changing project properties”](#).



- ▶ The **Default Generation path** contains the default value for the code generator's output path. If you change this value, the **New Project** wizard uses this new value as default for the project's **Generation path** text box. Additionally, the legacy command line mode also uses this setting.
- ▶ Select **Show information dialog when code generation finished successfully** to show an information dialog after successful code generation.
- ▶ Select **Show information dialog when code generation finished with errors** to show an information dialog when code generation finished with errors.
- ▶ Select **Show information dialog when code generation was canceled** to show an information dialog after code generation was canceled.

6.2.2.5. Setting configuration preferences

To set configuration preferences:

- ▶ select the **Preferences EB tresos Studio** subpage **Configuration** and use the following settings:

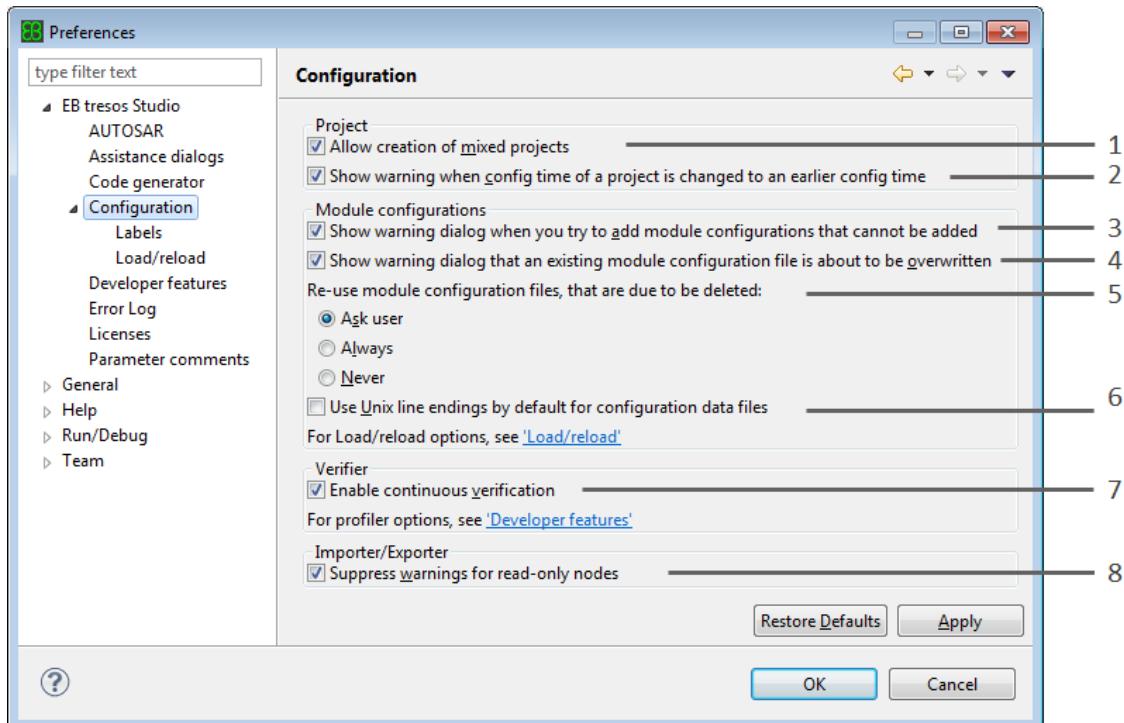


Figure 6.1. Preferences subpage *Configuration*

Number	Label	Instruction
1	Allow creation of mixed projects	Select to configure modules with different release versions. A mixed project may contain module configurations with different release versions. If you enable this feature, the Release Version combo box in the project wiz-



Number	Label	Instruction
		ard (see Section 6.3.1, "Creating a new project") contains the entry mixed .
2	Show warning when config time of a project is changed to an earlier config time	Select to get a warning that describes the consequences as soon as you change the config time of your project to an earlier config time.
3	Show warning dialog when trying to add module configurations that can not be added	If checked a warning dialog is shown any time you try to add a module configuration that can not be added, e.g. if there is already a configuration for that module and it does not allow multiple configurations.
4	Show warning dialog that an existing module configuration file is about to be overwritten	If checked a warning dialog is shown any time you uncheck the "Load existing file" option when adding a new module configuration.
5	Re-use module configuration files, that are due to be deleted:	<p>Ask user Select to be asked when you add a module and there is already an existing configuration data file whether you want to re-use or overwrite it.</p> <p>Always Select to automatically re-use existing configuration data files when you add a module.</p> <p>Never Select to always overwrite existing configuration data files when you add a module.</p>
6	Use Unix line endings by default for configuration files	Select to use Unix line endings instead of platform line endings in configuration data files. This is a project-specific configuration. You may change it for each project in the preferences. The configuration data files are located inside the config directory of the project directory. The exporters also use this setting.
7	Enable continuous verification	Select to enable/disable continuous verification.

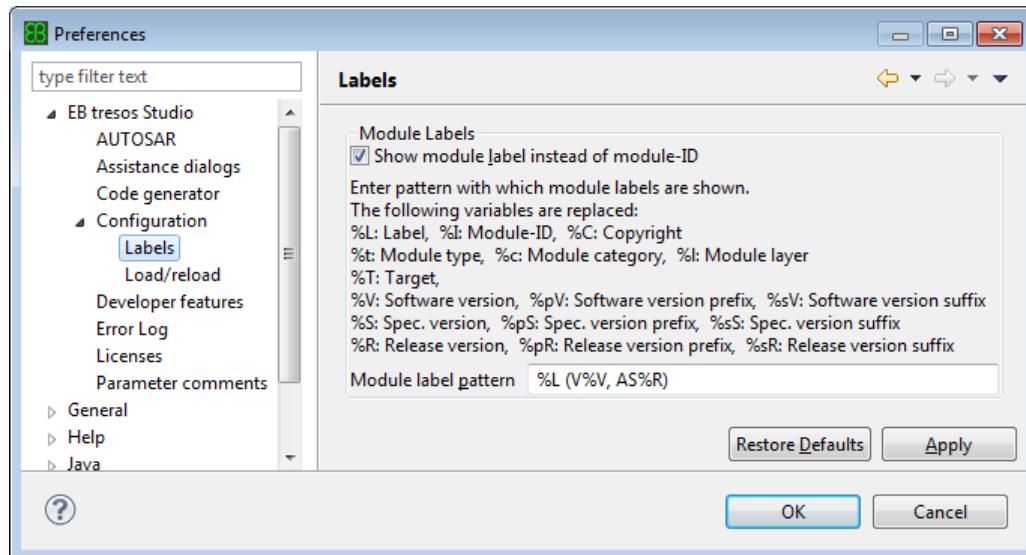


Number	Label	Instruction
		<p>NOTE</p>  <p>If enabled, the verification of the configuration will run continuously in the background for all loaded projects. Detected problems will be displayed in the Problems View.</p>
		<p>WARNING</p>  <p>Even if you disable continuous verification, EB tresos Studio will still verify the configuration, when you start the code generation. No code will be generated if the verification detects errors in the configuration.</p>
8	Suppress warnings for readonly nodes	Select if you want to suppress warnings for readonly nodes while running importer or exporter.

6.2.2.6. Setting Labels preferences

To set Labels preferences:

- ▶ Select the **Preferences EB tresos Studio** subpage **Labels** and use the following settings:



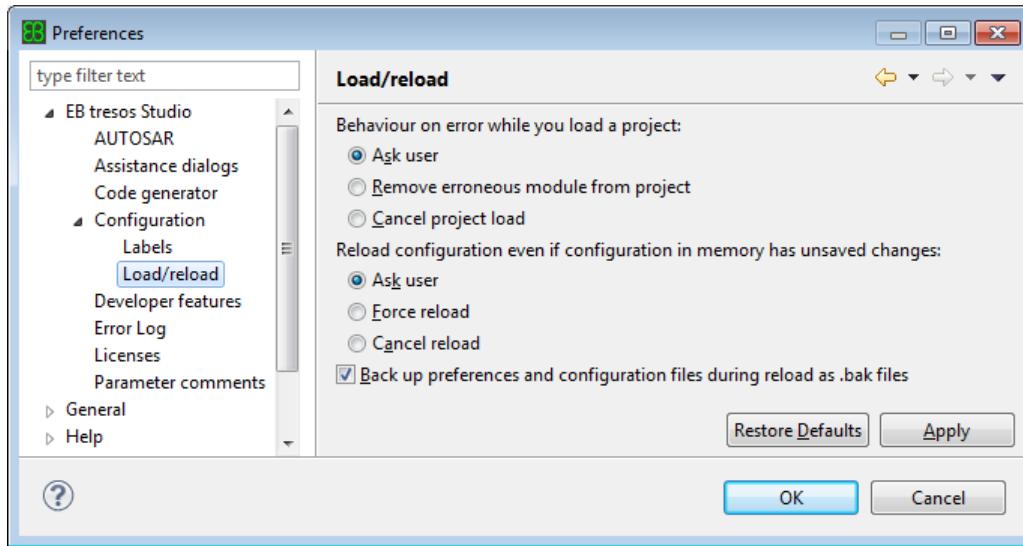
- ▶ Select **Show module label instead of module-ID** to display a module in the GUI with a pattern defined below. This setting is visible e.g. in the **Project Browser** or the **Add Module** dialog.
- ▶ Enter a pattern in **Module label pattern** that defines what the module label should look like in the GUI.



6.2.2.7. Setting Load/reload preferences

To set Load/reload preferences:

- ▶ Select the **Preferences EB tresos Studio** subpage **Load/reload** and use the following settings:



- ▶ **Behavior on error while you load a project:**

Select **Ask user** to be asked to do one of the following actions, when you try to load a project and an error occurs while a module is loading:

- ▶ Remove this module from the project and continue loading.
- ▶ Cancel the load process.

Select **Remove erroneous module from project** to remove the erroneous module persistently from the project, when you try to load a project and an error occurs while a module is loading. The project load continues without this module.

Select **Cancel project load** to cancel the load process, when you try to load a project and an error occurs while a module is loading. After that, you can resolve the problem manually and try loading the project again.

- ▶ **Reload configuration even if configuration in memory has unsaved changes:**

Select **Ask user** to be asked whether you want to discard your changes or cancel the reload process, when you are about to reload a project with unsaved changes.

Enable **Force reload** to always discard changes without further notice, when you are about to reload a project with unsaved changes.



Enable **Cancel reload** to always cancel the reload process, when you are about to reload a project with unsaved changes. For further information on reloading a project, see [Section 6.3.6.4, “Reloading a project”](#).

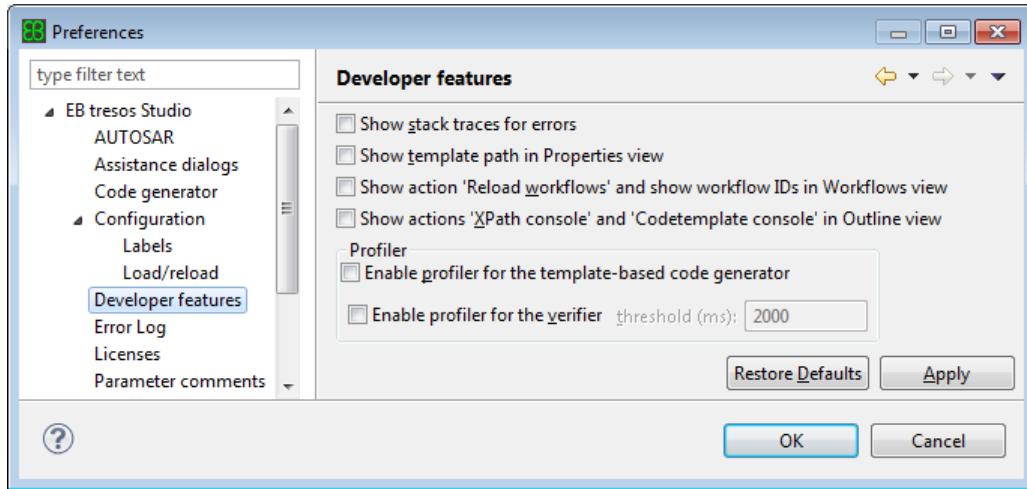
► **Back up preferences and configuration files during reload as .bak files**

Select this preference to backup the preferences and configuration files, when you reload an already loaded project. This backup then contains the current in-memory representation, including any unsaved changes. The backup files are stored in the same directories as the original files with a `.bak` file extension.

6.2.2.8. Setting Developer features preferences

To set Developer features preferences:

► Select the **Preferences EB tresos Studio** subpage **Developer features** and use the following settings:



- Select **Show stack traces for errors** to additionally show the stack trace in the **Error Log Entry Details** dialog.
- Select **Show template path in Properties view** to show the path of the selected element in the **Properties** view as it can be used within code templates.
- Select **Show action 'Reload workflows' and show workflow IDs in Workflows view** to show the action 'Reload workflows' in the **Workflow** view and to show the workflow ID at every place the workflow name is shown.
- Select **Show actions 'XPath console' and 'Codetemplate console' in Outline view** to show the mentioned consoles in the context menu of the **Outline** view.
- Select **Enable Profiler for the template-based code generator** to create a `.profile` file next to each file generated by the template-based code generator. Those files contain information about the time required by each executed command within the template.



- ▶ Select **Enable Profiler for the verifier** to create a file that contains information about the time required to verify each parameter every time the verifier finishes. This file contains only parameters for which the verification took more than the time configured in **threshold (ms)**.

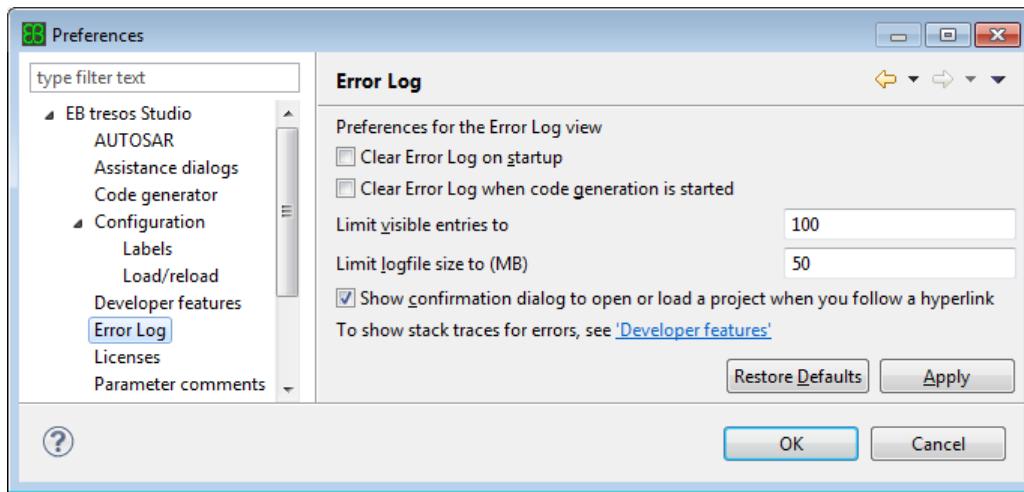
An entry is shown in the **Error Log** view every time a profile file is written that contains the location of the file.

Each entry written to the logfile represents the summed up timing information for all configuration parameters, which belong to the same parameter definition. Each entry also mentions the amount of these configuration parameters and the path to the parameter definition where the verification rules are usually configured.

6.2.2.9. Setting Error Log preferences

To set Error Log preferences:

- ▶ Select the **Preferences EB tresos Studio** subpage *Error Log*



- ▶ Select **Clear Error Log on startup** if you want the **Error Log** view to contain no messages when you start EB tresos Studio.
- ▶ Select **Clear Error Log when code generation is started** if you want to remove all messages from the error log. You can do this e.g. before generating code, to clearly see which messages are logged during code generation.
- ▶ To limit the number of shown entries in the **Error Log** enter the respective number of entries you wish to be displayed in the **Limit visible entries to** text field. Entering big numbers may slow down the GUI.
- ▶ All errors are written to a logfile called `.tresoslog`. With increasing size of the logfile, the GUI may become slower and the memory usage might increase. The parameter **Limit logfile size to (MB)** provides an upper bound after which the content of the log is discarded and written to a backup file.



- ▶ Some **Error Log** entries may contain hyperlinks to a parameter. When you select **Show confirmation dialog to open or load a project when you follow a hyperlink**, a dialog is shown that lets you decide whether to automatically open and load the project or to not follow the hyperlink.

6.2.2.10. Setting License preferences

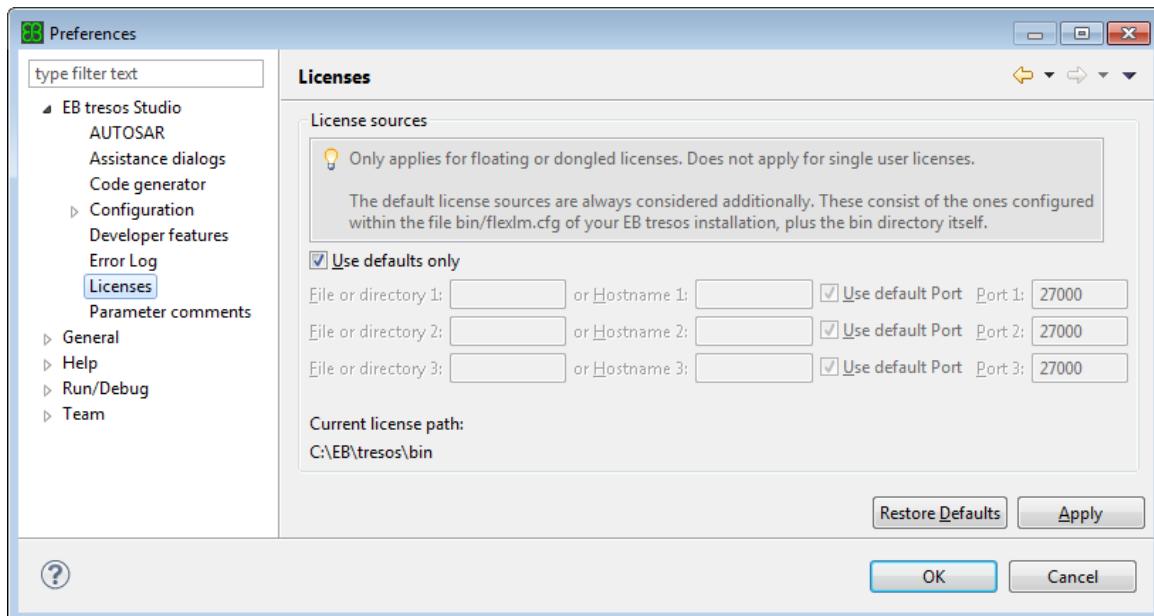
NOTE
License preferences only apply for floating licenses or dongled licenses


Setting the license preferences may only be relevant if you use a license server or have received one or more FlexLM license files, e.g. together with an USB dongle. It is not applicable if you use activation-based single user licenses.

For more information about using single user licenses, see the EB tresos installation guide.

To configure the location of your FlexLM license file or the hostname and port of your FlexLM license server:

- ▶ Select the **Preferences EB tresos Studio** subpage *Licenses*



- ▶ Uncheck the **Use defaults only** checkbox.
- ▶ Fill in the form to define up to three FlexLM license files or folders that contain license files, or server hostnames and their ports as sources of licenses. It is recommended to use the default-port, which automatically checks several ports.

In addition to the license sources you configure here, some default license sources are added to the license path. These consist of the ones which you may configure within the `flexlm.cfg` file, as well as the `bin` directory of your EB tresos Studio installation.



The `flexlm.cfg` file must be located within the `bin` directory of your EB tresos Studio installation. It contains the following information:

```
# Configure up to 3 FlexLM license sources, one per line.
# These are either server hostnames and ports for floating
# licenses in the form [<port>]@<hostname>. Or they are
# absolute paths to directories where your *.lic files reside,
# or absolute paths to your license files directly.
# E.g.
# @your.license.server
# 27000@your.license.server
# C:\path\to\your\license\files
# C:\path\to\your\licenseFile.lic
```

- ▶ The resulting **Current license path**: is updated after you click the **Apply** button.

6.2.2.11. Setting preferences for parameter comments

To set preferences for parameter comments:

- ▶ Select **Parameter comments** from the **EB tresos Studio** preferences.

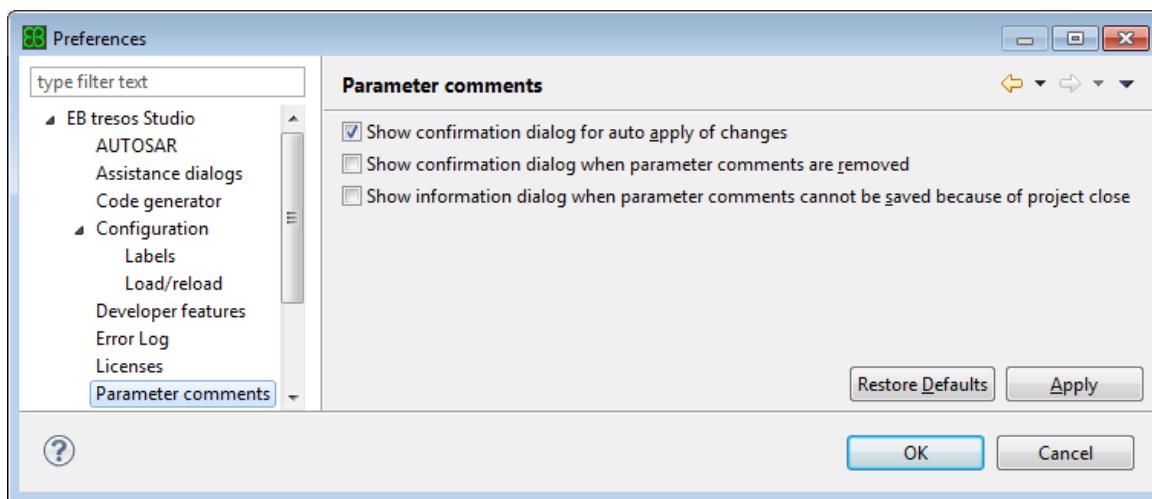


Figure 6.2. Setting up preferences for parameter comments

- ▶ Select **Show confirmation dialog for auto apply changes** to show a confirmation dialog each time your changes to a parameter comment are not saved in the **Comments for <parameter name>** dialog.
- ▶ Select **Show confirmation dialog when parameter comments are removed** to show a confirmation dialog each time a parameter comment is removed.



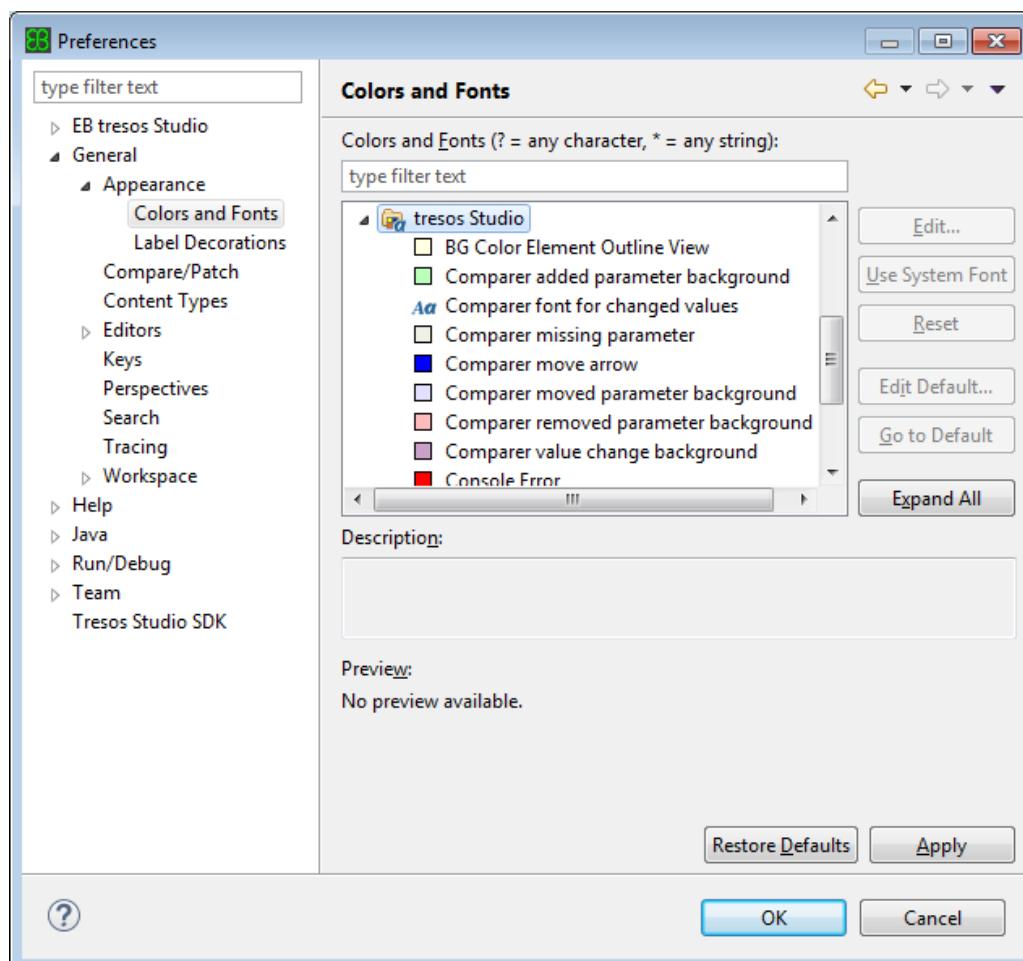
- ▶ Select **Show information dialog when parameter comments cannot be saved because of project close or shutdown EB tresos Studio** to show an information dialog each time an unapplied parameter comment cannot be saved, e.g. during application shutdown or project close.

6.2.2.12. Setting colors and fonts

To set colors and fonts:

- ▶ Open the **Preferences** window and select **General** tree view on the left side.
- ▶ Open the tree view, select **Appearance**.
- ▶ Open the tree view under **Appearance** and select **Colors and Fonts**.

You see a categorized list of colors and fonts used in EB tresos Studio.



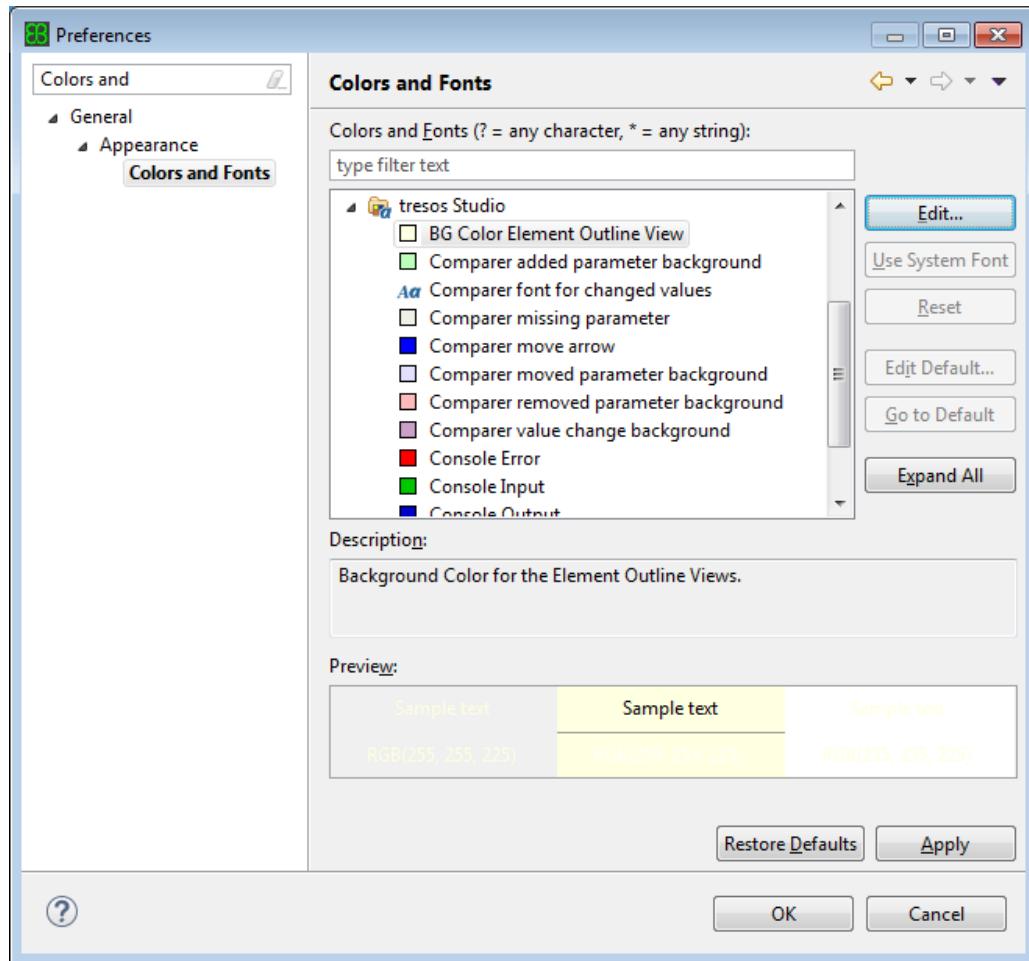
- ▶ Select any of the items to retrieve more information about it in the **Description:** field.

To change any of the colors:



- ▶ Select a color in the **Colors and Fonts** view on the right side.

The color will appear on a button left to a **Reset** button.



- ▶ Click the colored button.

A window with color selections opens up.

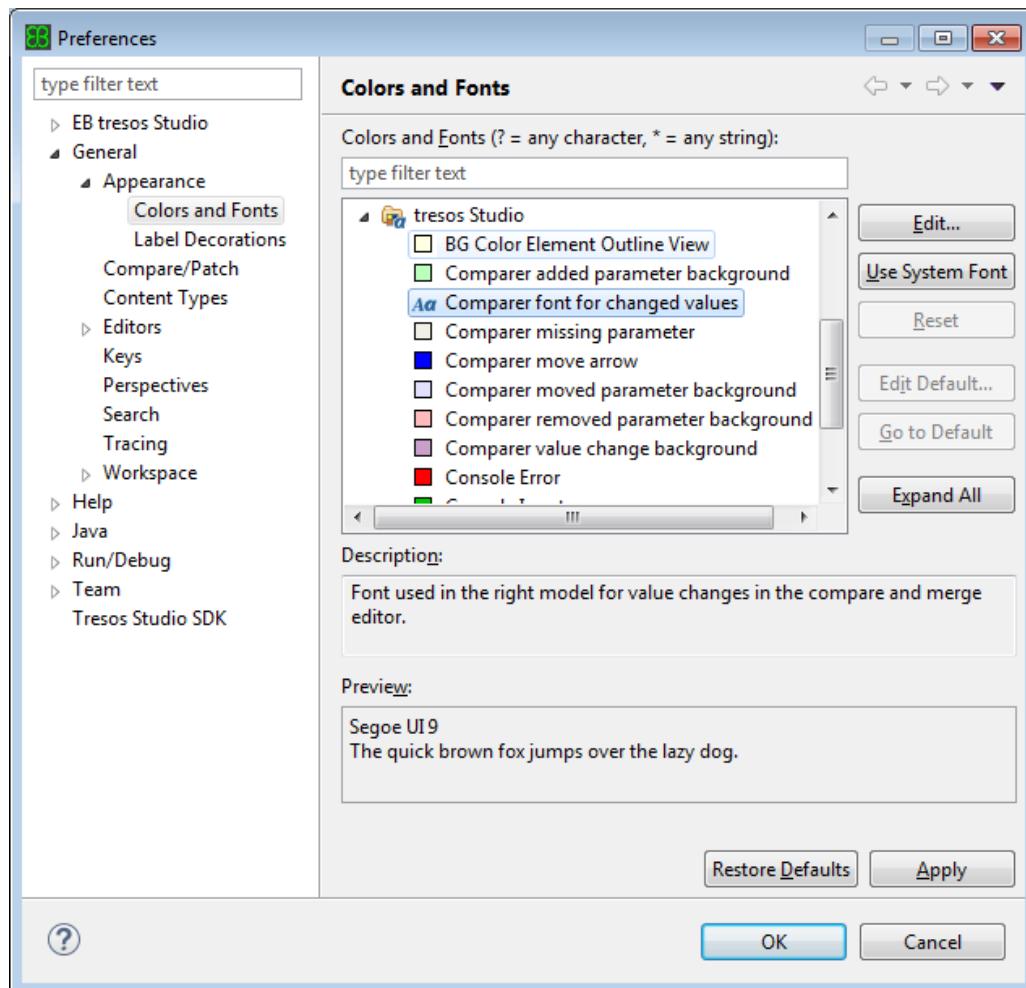
- ▶ Select a color of your liking and click **OK**.

EB tresos Studio now displays the new color in the setting you chose.

To change any of the fonts:

- ▶ Select a font in the **Colors and Fonts** view on the right side.

Buttons for adjusting the font will appear on the right.



- Click the **Change...** button.

A window to adjust the font opens up.

- Adjust the font to your likings and click **OK**.
- To reset the font to its default setting, click **Reset**.
- You can also select **Use System Font** to set the font to a reasonable value chosen by the operating system.
- Click **OK**.

EB tresos Studio now uses the font in the setting you chose.



6.2.3. Working with multiple workspaces to store different projects and settings

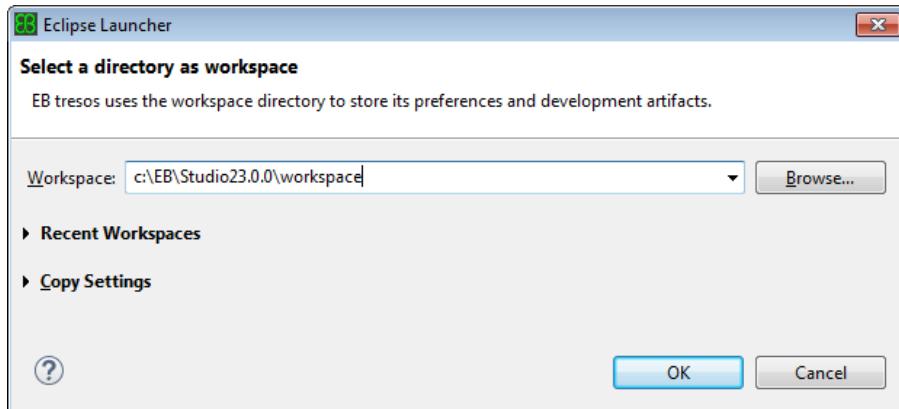
A *workspace* in EB tresos Studio contains all your projects and associated information like your preferences (see [Section 6.2.2, “Setting preferences”](#)). If you want to keep some projects completely separated from others, or if you want to work with different preferences, you can use multiple workspace and switch to the workspace you are planning to use.

NOTE

The default workspace is located directly inside your EB tresos Studio installation directory and is called `workspace`. If you did not change anything you are using this workspace.

6.2.3.1. Switching the workspace

To switch the workspace, open the **File** menu and select **Switch workspace** and then **Other**. The **Eclipse Launcher** dialog opens:



Use the text box labeled **Workspace** to type in your new workspace location. You can also use the **Browse** button to open up a Windows explorer to select the new location. If you have already used the switch workspace feature before, the directories used previously are available by clicking the drop-down arrow of the text box.

If you click on **Copy settings** the dialog expands and you get more options (as depicted in the screenshot). Select **Workbench layout** if you have modified the view layout of EB tresos Studio and want to use the same layout in your new workspace. If you do not choose this option, the default layout will be used and EB tresos Studio looks like when you started it for the first time.



6.2.3.2. Finding out which workspace directory is used

If you want to check which workspace directory EB tresos Studio is currently using, open the **EB tresos Details** dialog (see [Section 6.2.1.4, “Finding the location of the installation and the workspace”](#)). In the list on the **General** tab under **Workspace location** you can see the directory of the current workspace.

6.2.4. Placing an EB tresos Studio installation under version control

It is possible to place a complete EB tresos Studio installation under version control. You can submit the complete EB tresos Studio installation directory to your version control system, except the following files and directories:

- ▶ `workspace/`,
- ▶ `configuration/.settings/`, and
- ▶ `configuration/org.eclipse.*`.

Some version control systems mark files as read-only when you check them out. In this case you need to make all files from the EB tresos Studio installation directory writable by the current user.

6.2.5. Adding new features and extensions to EB tresos Studio via external folders

If you want to extend EB tresos Studio with new features or with an application that is compatible with the Eclipse version used, you can add the plug-ins of these features or extensions to external folders. After you set a link to these folders from the EB tresos Studio installation, the plug-ins of the features and extensions are loaded into EB tresos Studio when you start EB tresos Studio. To remove the feature or application and have a clean EB tresos Studio installation again, you can simply remove this link.

Eclipse requires a specific directory structure for additional plug-in folders. The directory structure is similar to a normal Eclipse installation: plug-ins are stored in an `eclipse\plugins` folder, which may be anywhere on your PC. Links are stored in a `links` folder in your EB tresos Studio installation, which is referred to as `$TRESOS_BASE`.

To add an extension or new feature to your EB tresos Studio installation:

- ▶ Name the folder in which you have stored your new feature plug-ins `eclipse\plugins`, e.g. `c:\development\extensions\eclipse\plugins`.



- ▶ To link to this folder with your new feature plug-ins from EB tresos Studio, create a folder called `links` in your `$TRESOS_BASE` directory: `$TRESOS_BASE\links`.
- ▶ In the `$TRESOS_BASE\links` folder, create a text file with the suffix `.link`, e.g. `additional.link`. The name of the file extension is important, but you can freely choose the name of the file itself.
- ▶ Open your `.link` file in a text editor and add the path to the folder with the new feature plug-ins without the `eclipse/plugins` folders in this path. E.g. `write path=c:/development/extensions` into your `.link` file.

NOTE**Write the path to your external folder with normal slashes**

You must write the path to your external folder in the `.link` file with normal slashes, i.e. with `/`, not with backslashes (`\`).

-
- ▶ Save your `.link` file.
 - ▶ To apply your changes, restart EB tresos Studio.
 - ▶ To check whether your plug-ins are loaded correctly, select **EB tresos Details** from the **Help** menu. If your new feature appears in the **EB tresos Details** dialog, the link and all paths are correct.

If your new feature does not appear in the **EB tresos Details** dialog, check and correct all paths of the files and folders.

6.3. Working with projects

This section describes how to work with projects in EB tresos Studio. A project contains all the configurations for one ECU. Before being able to edit module configurations, or to generate code, you need to create a project first and add modules to the project. The project also stores meta-information about the ECU like e.g. the ECU ID or the target and derivate setting.

6.3.1. Creating a new project

To create a new configuration project, EB tresos Studio provides the **Project Wizard**. The Project Wizard guides you through the creation of a new project via several dialog pages. You proceed from one page to the next by clicking the **Next** button. You can always return to a previous page and change your settings by pressing the **Back** button. Only when you click the **Finish** button your settings are executed and a new project is created.



NOTE Most of the wizard's buttons stay disabled until you have entered all required data.



6.3.1.1. Step 1: Starting the Project Wizard

To start the Project Wizard, there are two alternatives:

- ▶ Alternative 1: Right-click within the Project Explorer (see [Section 5.3.1, “Project Explorer view”](#)).

A context menu opens up.

- ▶ Select **New**.

A context menu opens up.

- ▶ Select **Project**.

The Project Wizard **Select a wizard** page opens up.

- ▶ Select **Configuration Project** from the list.

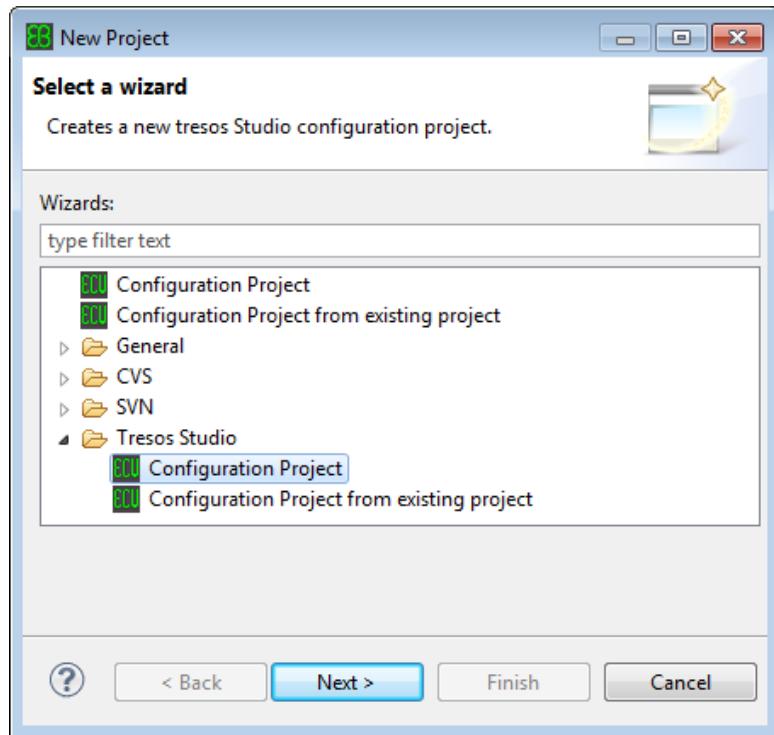
NOTE

The **Configuration Project from existing project** option is used to upgrade an existing project to a newer AUTOSAR version. You can find details about this in [Section 6.7.1, “Converting a project to a new AUTOSAR version”](#).



- ▶ Click **Next**.

The **New Configuration Project** page opens up.



Alternative 2: to start the Project Wizard,

- ▶ Select the **File** menu.
A context menu opens up.
- ▶ Select **New**.
A context menu opens up.
- ▶ Select **Configuration Project** from the list.

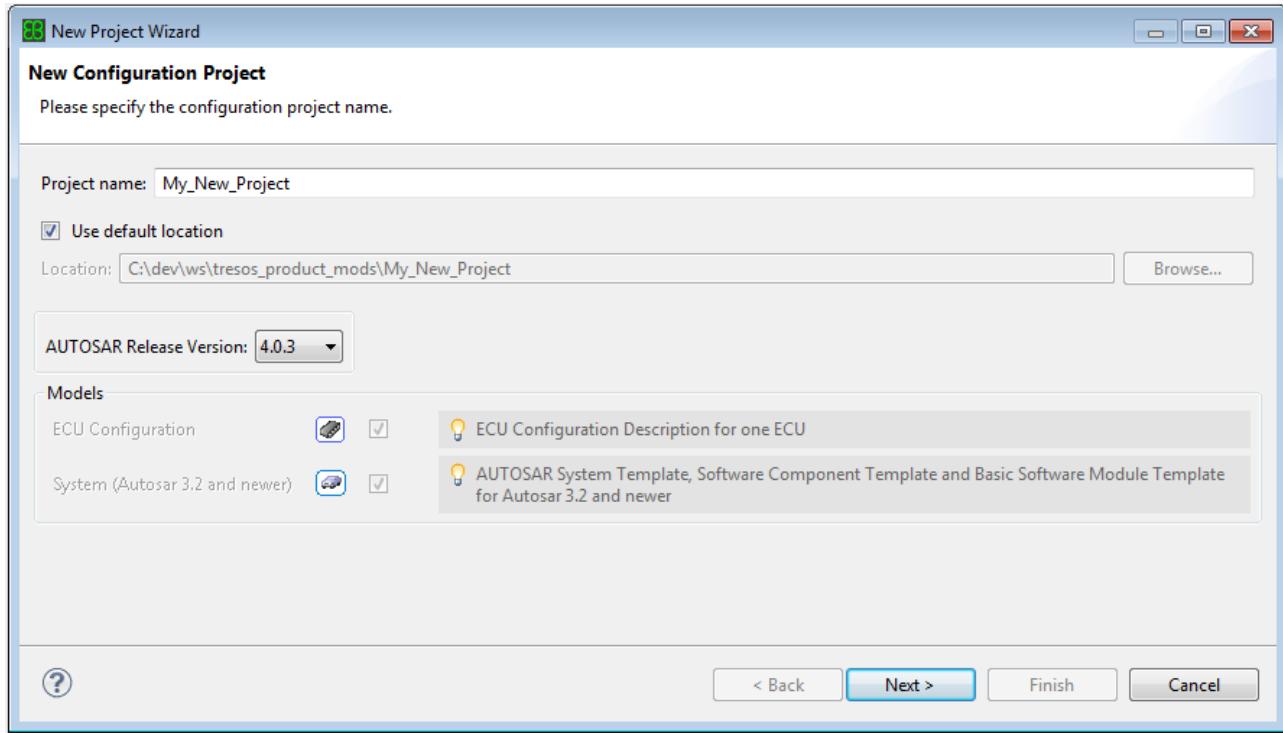
NOTE

The **Configuration Project from existing project** option is used to upgrade an existing project to a newer AUTOSAR version. Details about this can be found in [Section 6.7.1, "Converting a project to a new AUTOSAR version"](#).

-
- ▶ The **New Configuration Project** page opens up.



6.3.1.2. Step 2: Creating a new project



- ▶ Enter a project name in the **Project name** text box.

NOTE

The project name must be a valid AUTOSAR shortname.



- ▶ To store the project in the workspace folder, check the box **Use default location**.
- ▶ To store the project in another location, uncheck the check box and select a path.
- ▶ Fill in the **AUTOSAR Release Version**. Use the drop-down list box to select the version of your project.

For example if you select **4.0.2**, you can add only modules with AUTOSAR release version 4.0.2 to your project. If you select **Mixed**, you may add all available modules for the selected target/derivative and configure module configurations with different release versions.

NOTE

This choice affects the Exporter



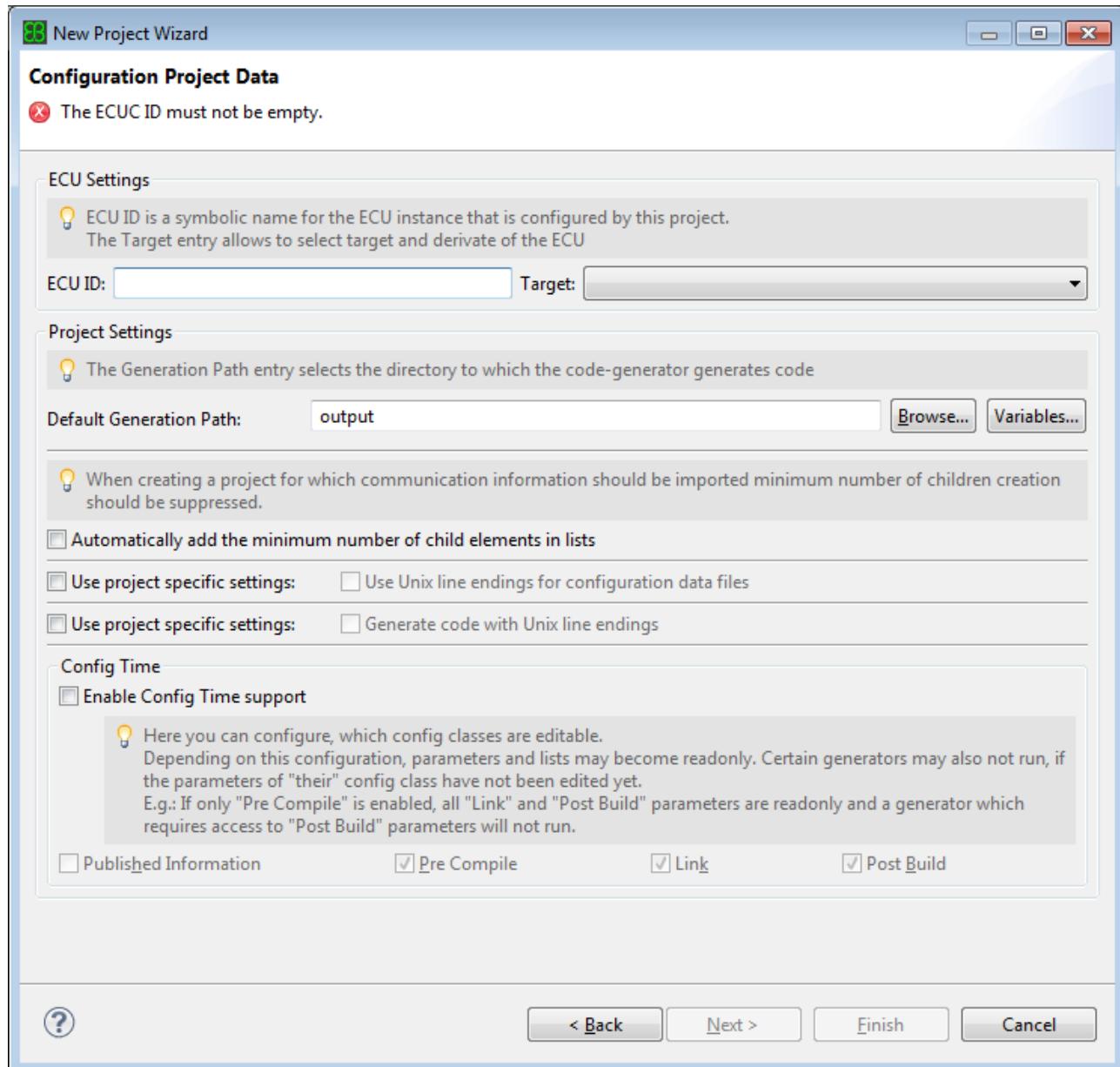
This choice affects the Exporter and restricts the possible target content types (see [Section 6.10.2.2, “Importing and exporting configuration data from and to the AUTOSAR format”](#)).

- ▶ Click **Next**.



The **Configuration Project Data** page opens up.

6.3.1.3. Step 3: Configuring project properties



On the **Configuration Project Data** page:

- ▶ Fill in the **ECU ID**.

Most likely this ID is provided by the OEM as part of a system description that describes a vehicle as a whole.



- ▶ Fill in the **Target**. Use the drop-down list box to select the target/derivate of your ECU.
- ▶ Select the **Project Settings** according to your preferences.
 - ▶ To automatically add the minimum number of child elements in lists select the check box **Automatically add....**
 - ▶ To use project-specific settings, check the check box **Use project-specific settings**.
 - ▶ *Line endings of configuration data files:* By default, the type of line endings which is used for the files which store the configuration data is controlled by a global workspace setting (see [Section 6.2.2.5, “Setting configuration preferences”](#)). If you check the check box **Use project-specific settings**, the local setting takes effect for the project which you are just creating. You can then check or uncheck the check box **User Unix line endings** The value you choose will only be active for this project.

NOTE

You can change this setting later after the project is created via the project preferences (see [Section 6.3.7.2, “Editing the Code Generator settings”](#)).

- ▶ *Line endings of generated code:* By default, the type of line endings of the files generated by the code generator (see [Section 6.9, “Generating code for projects”](#)) is controlled by a global workspace setting (see [Section 6.2.2.5, “Setting configuration preferences”](#)). If you check the check box **Use project-specific settings**, the local setting takes effect for the project which you are just creating. You can then check or uncheck the check box **Generate code with Unix line endings** and this setting will only be active for this project.

NOTE

You can change this setting later after the project is created via the project properties (see [Section 6.3.7.2, “Editing the Code Generator settings”](#)).

- ▶ Select **Enable Config Time Support** to allow only the configuration of parameters and references of specific configuration classes. Parameters and references that refer to one of the selected configuration class can then be edited by the user. By default this setting is deactivated and all parameters are editable (see [Section 3.5.2.1, “Configuration time of projects”](#))

- ▶ *Published Information:* This check box is never enabled by default, because parameters of the configuration class *Published Information* are fixed even before the pre-compile stage. These parameters can be seen as constants.
- ▶ *PreCompile:* If you select this check box, parameters of the configuration class *PreCompile* are editable.
- ▶ *Link:* If you select this check box, parameters of the configuration class *Link* are editable.
- ▶ *Post Build:* Makes parameters of the configuration class *Post Build* editable.

**NOTE****Dependency of configuration classes to code generators**

If you limit the configuration to certain configuration classes, this might affect the code generators. If a code generator requires access to parameters of a not selected or not editable configuration class, this code generator will not run. For detailed information on code generators, see [Section 6.9.2, "Generating a project"](#).

TIP**You may change these settings later on as well**

You may also change these settings later on in the project properties, after the project has been created. For instructions on setting the project properties, see [Section 6.3.7.3, "Editing the ECU Configuration settings"](#).

- ▶ Click **Next**.

The **Module Configurations** page opens up.

6.3.1.4. Step 4: Defining module configurations

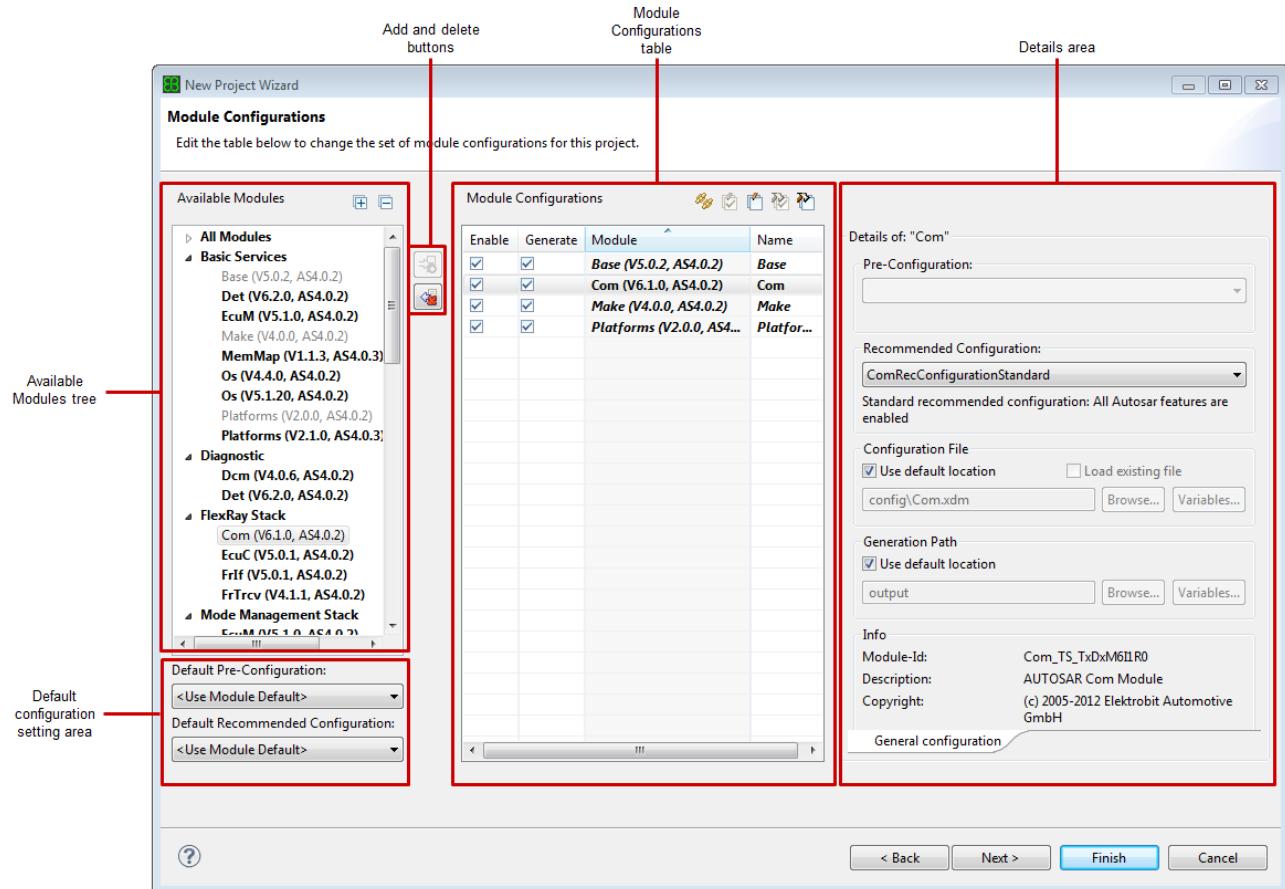


Figure 6.3. The **Module Configurations** page



On the **Module Configurations** page you may add or remove the module configurations to/from the project you want to create and you may edit the configuration settings of these module configurations. Some modules are mandatory for your architecture or AUTOSAR release version. These modules are added automatically and are already displayed in the **Module Configurations** table when you open the **Module Configurations** page. It is possible to remove them, but it is not recommended.

TIP**You may add module configurations later to your project**

You may also create a project which initially contains no module configurations and add them later. For instructions for adding or editing module configurations to/in your project, see [Section 6.6, “Editing module configurations”](#).

When you are done setting up your initial module configurations, click the **Next** button and continue reading at [Section 6.3.1.5, “Step 5: Importing data into the project \(optional\)”](#).

6.3.1.4.1. Adding module configurations to your project

You may add single module configurations, several module configurations at once or clusters of module configurations to your project.

NOTE**You can recognize whether a module has been added by its font**

Modules that are still available and modules that already have been added to your project are displayed in different font styles in the **Available Modules** tree:

- ▶ Modules that have not been added to your project and thus are still available are displayed in bold font.
- ▶ Modules that already have been added to your project are displayed in standard font. The modules that allow multiple configurations are displayed in black. All others are grayed-out and cannot be added to your project anymore.

To add new module configurations to your project:

- ▶ In the **Available Modules** tree, select all modules/clusters of modules for which you want to add configurations for.
- ▶ Click the **Add** button .

TIP**Add single modules with a double-click**

Module configurations for single modules can also be added by double-clicking the required module.

- ▶ If the required module configurations cannot be added to your project, the following dialog opens up:

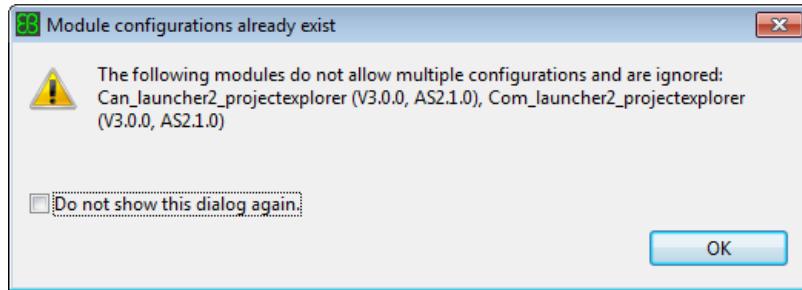


Figure 6.4. Some modules may be added only once to your project

The selected modules are added to your project as module configurations with default settings.

NOTE



Only one module configuration is added per selected module

Since modules can be part of more than one cluster, it is possible that a selection in the **Available Modules** tree refers several times to the same module. Note that only one configuration is added in this case, even if the module allows multiple configurations.

As long as the recently added module configurations have not been stored into your project yet, you may edit the (default) settings of the module configuration. Mind that depending on the module, there might be more tab pages next to the **General configuration** tab on the bottom of the **Details** area.

6.3.1.4.2. Removing module configurations from your project

You may remove single module configurations or several module configurations at once from your project.

NOTE



You can recognize the state of a module by its font

Module configurations that previously had been added to your project, mandatory module configurations and recently added module configurations are displayed in different fonts in the **Modules Configuration** table:

- ▶ Mandatory module configurations are displayed in italic font. It is not recommended to remove these module configurations from your project.
- ▶ Recently added module configurations are displayed in bold font. You may edit them.
- ▶ Module configurations that had been added to your project before are displayed in standard font.

To remove module configurations from your project:

- ▶ In the **Module Configurations** table, select the module configuration(s) you want to remove.

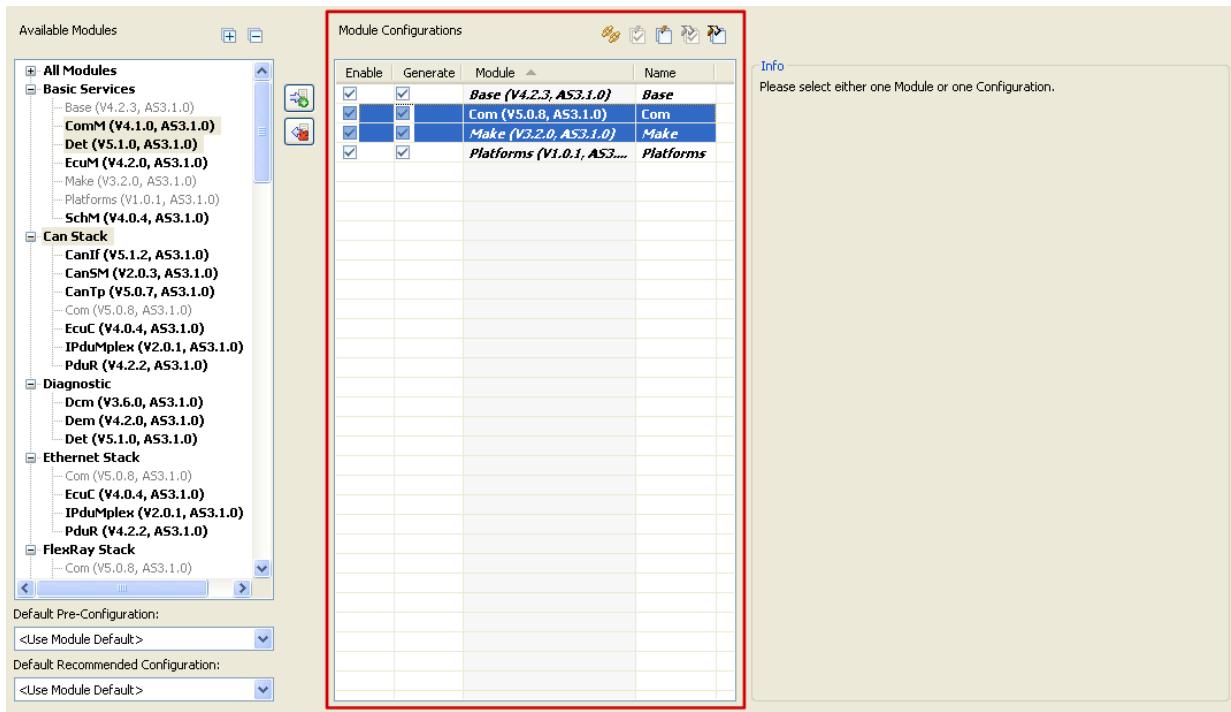


Figure 6.5. Removing module configurations

- ▶ Click the **Delete** button .
- ▶ If none of the selected module configurations refers to an already existing configuration file, the items selected are removed from the **Module Configurations** table without further notice. Otherwise the **Delete Module Configurations** dialog opens up

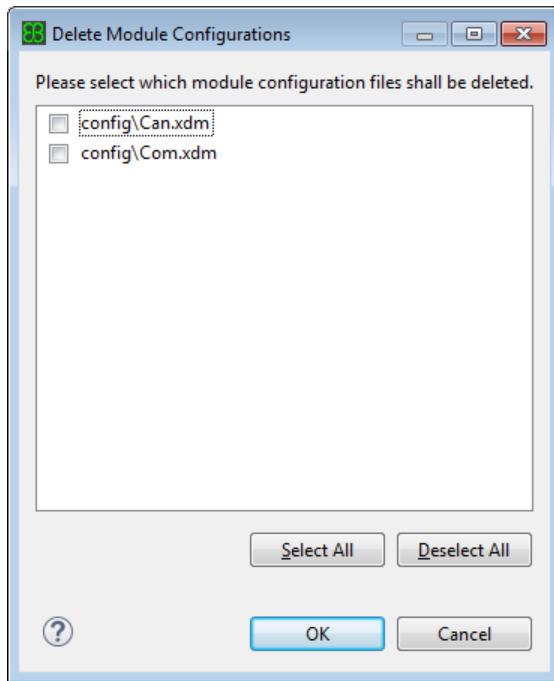


Figure 6.6. Confirming which module configuration files you want to remove

- ▶ Select the check box of the configuration files you want to delete.

NOTE**Data loss**

The selected files will be deleted from the file system when the changed module configurations are applied. After that, you cannot restore these files anymore.

- ▶ To confirm the deletion of the configuration files, click **OK**.

The items selected are removed from the **Module Configurations** table.

- ▶ To cancel the deletion of the **Module Configurations** and their referred configuration files, click **Cancel**.

6.3.1.4.3. Enabling and disabling module configurations

In the **Module Configurations** table you may (temporarily) enable or disable the module configurations you have added to your project. Disabled configurations are treated as if they do not exist, which means you cannot edit or verify them, and code generators cannot access them. To enable module configurations:

- ▶ To enable only one module configuration, select the **Enable** check box of the module configuration you want to enable in the **Module Configurations** table.



Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.7. Enabling one module configuration

- ▶ To enable several module configuration at once:
 - ▶ In the **Module Configurations** table, select the module configurations you want to enable.
 - ▶ In the tool bar of the **Module Configurations** table, click the **Enable module configuration** button .

To disable module configurations:

- ▶ To disable only one module configuration in the **Module Configurations** dialog/page, clear the **Enable** check box of the module configuration you want to disable in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.8. Disabling one module configuration

To disable several module configuration at once:

- ▶ In the **Module Configurations** table, select the module configurations you want to disable.
- ▶ In the tool bar of the **Module Configurations** page, click the **Disable module configuration** button .

6.3.1.4.4. Enabling and disabling code generation for module configurations

In the **Module Configurations** table you may enable or disable code generation for the module configurations of your project.

**WARNING****Only partly generated code leads to incorrectly working ECUs**

If you only generate the code of a part of the module configurations you later want to implement on an ECU, depending on which modules you choose, the code might not work correctly. This might lead to incorrectly working ECUs.

To avoid incorrectly working ECUs, generate the code of all module configurations in your project before deploying them on an ECU.

To enable code generation for module configurations:

- ▶ To enable code generation for only one module configuration, select the **Generate** check box of the module configuration for which you want to enable code generation in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.9. Enabling code generation for one module configuration

- ▶ To enable code generation for several module configurations at once:
 - ▶ In the **Module Configurations** table, select the module configurations, for which you want to enable code generation.
 - ▶ In the tool bar of the **Module Configurations** page, click the **Enable code generation** button

To disable code generation for module configurations:

- ▶ To disable code generation for only one module, clear the **Generate** check box of the module configuration for which you want to disable code generation in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input type="checkbox"/>	<input type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.10. Disabling code generation for one module configuration

- ▶ To disable code generation for several module configurations at once:



- ▶ In the **Module Configurations** table, select the module configurations, for which you want to disable code generation.
- ▶ In the tool bar of the **Module Configurations** page, click the **Disable code generation** button

6.3.1.4.5. Naming Module Configurations

When you add new module configurations to your project, the name of the new module configuration is set automatically to a name that is unique within the project using either the default configuration name provided by the module or the module name itself.

Instead of using the default name you may want to use another name for the module configuration.

NOTE
Only in recently added module configurations the name can be changed


You can only change the name of a module configuration if you added this module configuration during the same session of the dialog.

To edit the name of a module configuration:

- ▶ Add the required module configuration to your project.
- ▶ In the **Module Configurations** table, click into the **Name** cell of the module configuration whose name you want to change.
- ▶ Type the new name.



Figure 6.11. Editing the name of a new module configuration

NOTE

Name constraints

Mind that the module configuration name must meet the following constraints:

- ▶ The name must be unique within the project.
- ▶ The name must start with a letter.
- ▶ The name must contain only letters and digits plus the underscore character ('_').
- ▶ The name must not be empty.
- ▶ The name must not exceed 255 characters.

- ▶ Press **ENTER**.

The name of the module configuration is changed.



6.3.1.4.6. Choosing Pre-Configurations

Module suppliers optionally can equip modules with one or more preconfigurations that contain values for configuration parameters. If a module configuration of your project provides one or more preconfiguration(s), you must assign this module configuration to such a preconfiguration when adding it to your project. The parameters covered by the chosen preconfiguration are then fixed in the module configuration and cannot be edited by the user.

In the **Default configuration setting** area, you can set the **Default Pre-Configuration** which is used as a default value for all new module configurations. To support a more consistent selection, this preconfiguration setting is persisted. You can also edit the **Pre-Configuration** of an individual module configuration.

6.3.1.4.6.1. Setting the Default Pre-Configuration

In the **Default configuration setting** area of the **Module Configurations** page, you may edit the **Default Pre-Configuration** for new module configurations.

NOTE**The value set is a union set of all preconfigurations provided by all modules**

The preconfiguration names to choose from derive from the union set of all names provided by all modules that match the project's target architecture and AUTOSAR release version. If the chosen name is not valid for a specific module, the module's default preconfiguration is taken instead when you add it to your project.

To change the default preconfiguration, select the required preconfiguration in the **Default Pre-Configuration** drop-down list box.

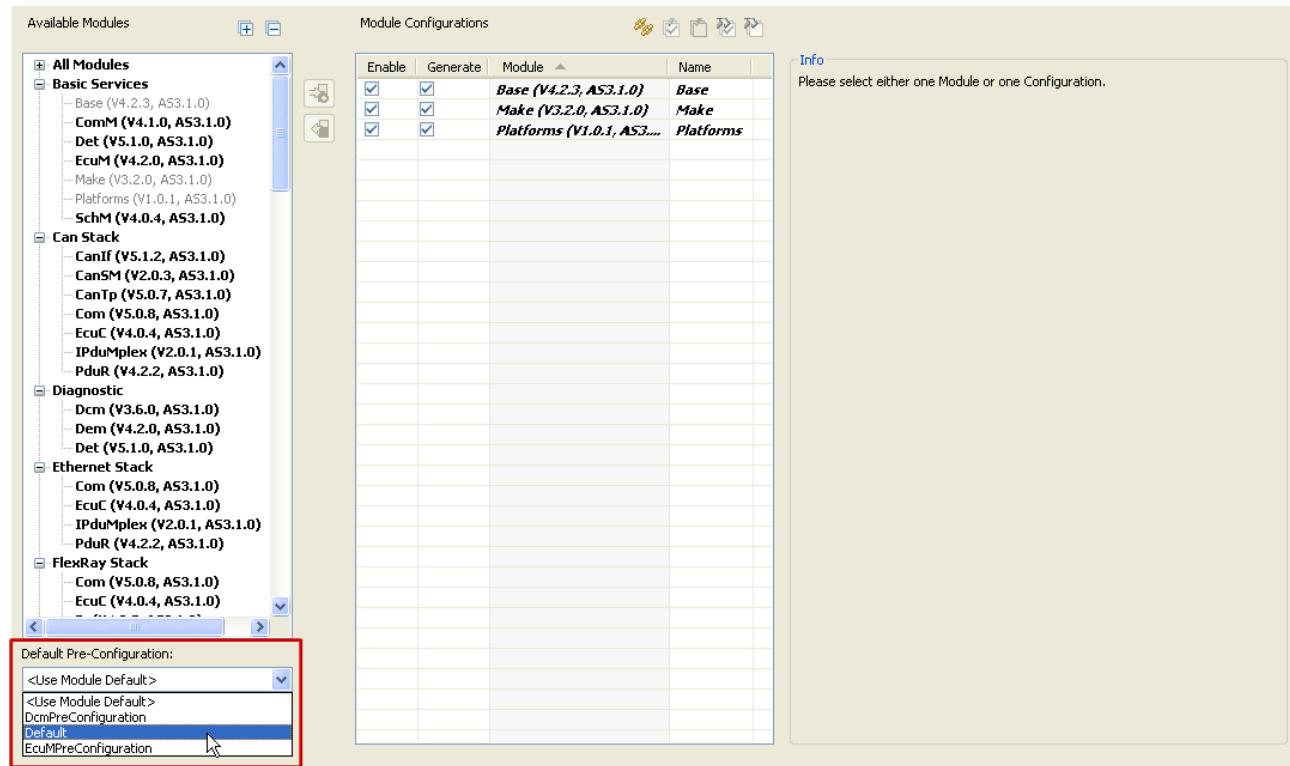


Figure 6.12. Selecting the default preconfiguration for new module configurations

- ▶ To use the respective default preconfiguration for new module configurations, select <Use Module Default>.

NOTE**Non-compilable code**

If you select <Use Module Default>, every module will be added to your project with a different preconfiguration. This might lead to incompatibilities of the module configurations and thus to non-compilable code.

- ▶ The selected default preconfiguration is saved, whenever you actually add new module configurations to your project.

6.3.1.4.6.2. Changing the preconfiguration of one specific module configuration

To change the preconfiguration of one specific module configuration:

- ▶ Add the required module configuration to your project.

**NOTE****Only in recently added module configurations the preconfiguration can be changed**

You can only change the preconfiguration of a specific module configuration if you added this module configuration during the same session of the dialog.

- ▶ In the **Module Configurations** table, mark the module configuration whose preconfiguration you want to change.

In the **Details** area of the **Module Configurations** page, detailed information about the specific module is displayed.

The screenshot shows the 'Module Configurations' dialog. On the left, the 'Available Modules' tree view is expanded to show categories like 'All Modules', 'Basic Services', 'Diagnostic', 'FlexRay Stack', 'Mode Management Stack', 'OS/RTE', and 'Stubs'. Under 'All Modules', several sub-modules are listed with their versions (e.g., Base V5.0.2, AS4.0.2; Com V6.1.0, AS4.0.2). On the right, a table lists 'Module Configurations' with columns for 'Enable', 'Generate', 'Module', and 'Name'. One row is selected, highlighting 'Com (V6.1.0, AS4.0.2)' under the 'Module' column. To the right of the table is the 'Details' area, which is highlighted with a red border. This area contains tabs for 'Pre-Configuration' (set to 'EcuMPreConfiguration'), 'Recommended Configuration' (set to 'ComRecConfigurationStandard'), 'Configuration File' (checkboxes for 'Use default location' and 'Load existing file' with 'config\Com.xdm'), 'Generation Path' (checkbox for 'Use default location' with 'output'), and 'Info' (displaying module details like 'Module-Id: Com_TS_TxDxM6I1R0', 'Description: AUTOSAR Com Module', etc.).

Figure 6.13. The **Details** area in the **Module Configurations** page

- ▶ In the **Pre-Configuration** drop-down list box in the **Details** area, select the required preconfiguration for the selected module configuration.

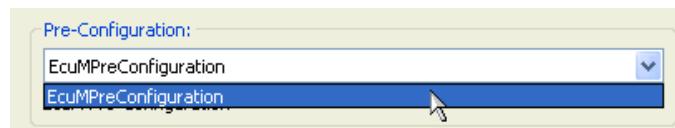


Figure 6.14. Changing the preconfiguration of one specific module



NOTE**Grayed-out preconfigurations are not available**

If the **Pre-Configuration** drop-down list is grayed-out, the selected module does not provide a preconfiguration of its own.

6.3.1.4.7. Applying recommended configurations

Module suppliers optionally can equip modules with one or more recommended configurations that contain values for configuration parameters. If a module configuration of your project provides one or more recommended configurations, you can choose one when you add a configuration for this module to your project. The parameter values covered by the chosen recommended configuration can be considered as a guideline. They may be edited by the user later.

In the **Default configuration setting area**, you can set the **Default Recommended Configuration** which is used as a default value for all new module configurations. To support a more consistent selection, this setting is persisted. You can also edit the **Recommended Configuration** of an individual module configuration.

6.3.1.4.7.1. Setting the Default Recommended Configuration

In the **Default configuration setting area** of the **Module Configurations** page, you may edit the **Default Recommended Configuration** for new module configurations.

NOTE**The value set is a union set of all recommended configurations provided by all modules**

The names of the recommended configurations to choose from derive from the union set of all names provided by all modules that match the project's target architecture and AUTOSAR release version. If the chosen name is not valid for a specific module, the module's default configuration is taken instead when you add it to your project. If the module does not provide a default configuration, no recommended configuration is applied.

To change the default recommended configuration, select the required recommended configuration in the **Default Recommended Configuration** drop-down list box.

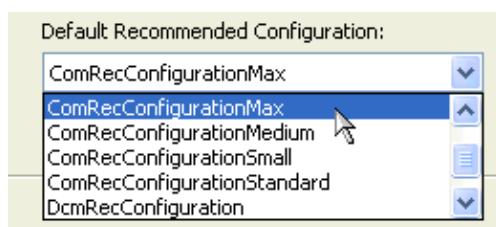


Figure 6.15. Selecting the default recommended configuration for new module configurations



- ▶ If you don't want to apply any of the recommended configurations, select <None>.
- ▶ To use the respective default recommended configuration for new module configurations, select <Use Module Default>.

NOTE**Non-compilable code**

If you select <Use Module Default>, every module will be added to your project with a different recommended configuration. This might lead to incompatibilities of the module configurations and thus to non-compilable code.

- ▶ The selected default recommended configuration is saved, when you actually add new module configurations to your project.

6.3.1.4.7.2. Changing the recommended configuration of one specific module configuration

To change the recommended configuration of one specific module configuration:

- ▶ Add the required module configuration to your project.

**Only in recently added module configurations the recommended configuration can be changed**

You can only change the recommended configuration of a specific module configuration if you have added this module configuration during the same session of the dialog.

- ▶ Mark the module configuration whose recommended configuration you want to change.

In the **Details** area of the **Module Configurations** page, detailed information about the specific module is displayed.



Enable	Generate	Module	Name
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V5.0.2, AS4.0.2)	Base
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Com (V6.1.0, AS4.0.2)	Com
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	FrTrcv (V4.1.1, AS4.0.2)	FrTrcv
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Make (V4.0.0, AS4.0.2)	Make
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Os (V4.1.0, AS4.0.2)	Os

Default Pre-Configuration: DcmPreConfigurationDflt

Default Recommended Configuration: <Use Module Default>

Details of: "Com"

Pre-Configuration:

Recommended Configuration: ComRecConfigurationStandard

Standard recommended configuration: All Autosar features are enabled

Configuration File: Use default location config\Com.xdm Load existing file Browse... Variables...

Generation Path: Use default location output Browse... Variables...

Info

Module-Id: Com_TS_TxDxM6IIR0

Description: AUTOSAR Com Module

Copyright: (c) 2005-2012 Elektrobit Automotive GmbH

Clusters: FlexRay Stack

Allows multiple configurations: No

Mandatory: General configuration

Figure 6.16. The Details area in the Module Configurations page

- In the **Recommended Configuration** drop-down list box, select the required recommended configuration for the selected module configuration.

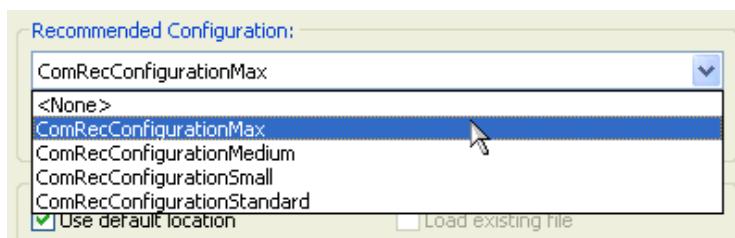


Figure 6.17. Changing the recommended configuration of one specific module

**NOTE****Grayed-out recommended configurations are not available**

If the **Recommended Configuration** drop-down list is grayed-out, the selected module does not provide a recommended configuration of its own.

6.3.1.4.8. Editing the configuration file settings

The configuration file stores all parameters for a specific module configuration. For detailed information on how to edit those parameters, e.g. by using the Generic Configuration Editor, see [Section 6.8, “Editing parameters of a module configuration”](#).

When you add new module configurations to your project, per default the configuration file name is set to config/<moduleconfigurationname>.xdm, relative to your project folder. This is also referred to as the default location.

Instead of using the default location you can use another location or name for the configuration file. To use another location or file name:

- ▶ Add the required module configuration to your project.

NOTE**Only in recently added module configurations the configuration file can be changed**

You can only change the configuration file of a module configuration if you have added this module configuration during the same session of the dialog.

- ▶ In the **Module Configurations** table, mark the module configuration whose configuration file you want to change.
- ▶ In the **Configuration File** panel of the **Details** area, clear the check box **Use default location**.

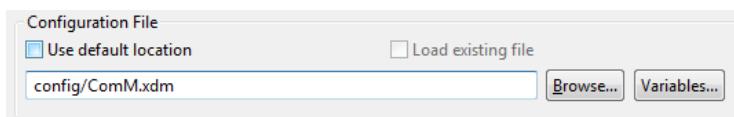


Figure 6.18. Editing the configuration file settings

- ▶ Enter or choose the new file name. Either use absolute path names or path names relative to your project. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#)

**NOTE****The file extension .xdm is mandatory**

You can only use file names ending with the extension `.xdm`. When you choose an existing file, you must make sure that it matches the targeted module configuration schema. Otherwise the new module configuration will fail to load when applied.

6.3.1.4.8.1. Using existing configuration files

Instead of configuring the module configuration from scratch you may also use an already existing configuration file.

NOTE**You cannot set preconfigurations and recommended configurations when you load an existing configuration file**

If you use an existing configuration file, you cannot apply any preconfiguration or recommended configuration.

To load an already existing configuration file into your project:

- ▶ In the **Configuration File** panel, browse to the file you want to load.

NOTE**The selected configuration file will be modified**

The selected configuration file will be modified while you edit the parameters of the module configuration. If you want to preserve the file as it is, copy the file to your project before loading it to your project.

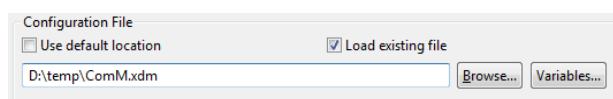


Figure 6.19. Using an existing configuration file

Whenever you point the **Configuration File** panel to an existing file in the file system, the **Load existing file** check box is marked.

NOTE**Data loss**

If you clear the **Load existing file** check box, the existing file will be overwritten and cannot be restored again.

6.3.1.4.9. Editing the code generation output path settings

In **Generation Path**, specify the directory into which EB tresos Studio writes the generated code files for this module configuration.



When you add new module configurations to your project, the **Generation Path** is set to the **Default Generation Path** of the project. This is also referred to as the default location. For more information about the **Default Generation Path** of the project, see [Section 6.3.7.2, “Editing the Code Generator settings”](#).

Instead of using the default location you can use another location for the generated code of this module configuration. To use another location:

- ▶ In the **Module Configurations** table, mark the module configuration whose generation path you want to change.
- ▶ In the **Generation Path** panel of the **Details** area, clear the check box **Use default location**.



Figure 6.20. Editing the generation path settings

- ▶ Enter or choose the new path name. Either use absolute path names or path names relative to your project. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#).

6.3.1.4.10. Upgrading module configurations

To upgrade module configurations from older software versions of the modules to newer ones:

- ▶ Click the **Upgrade** button  in the tool bar of the **Module Configurations** table.

The **Upgrade** dialog opens up.

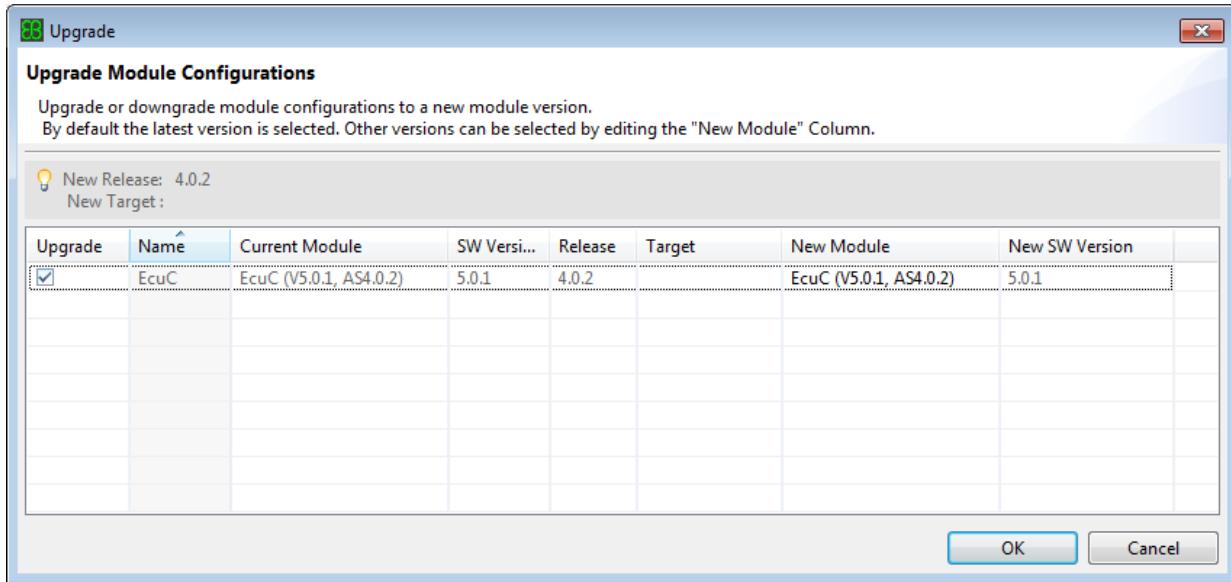
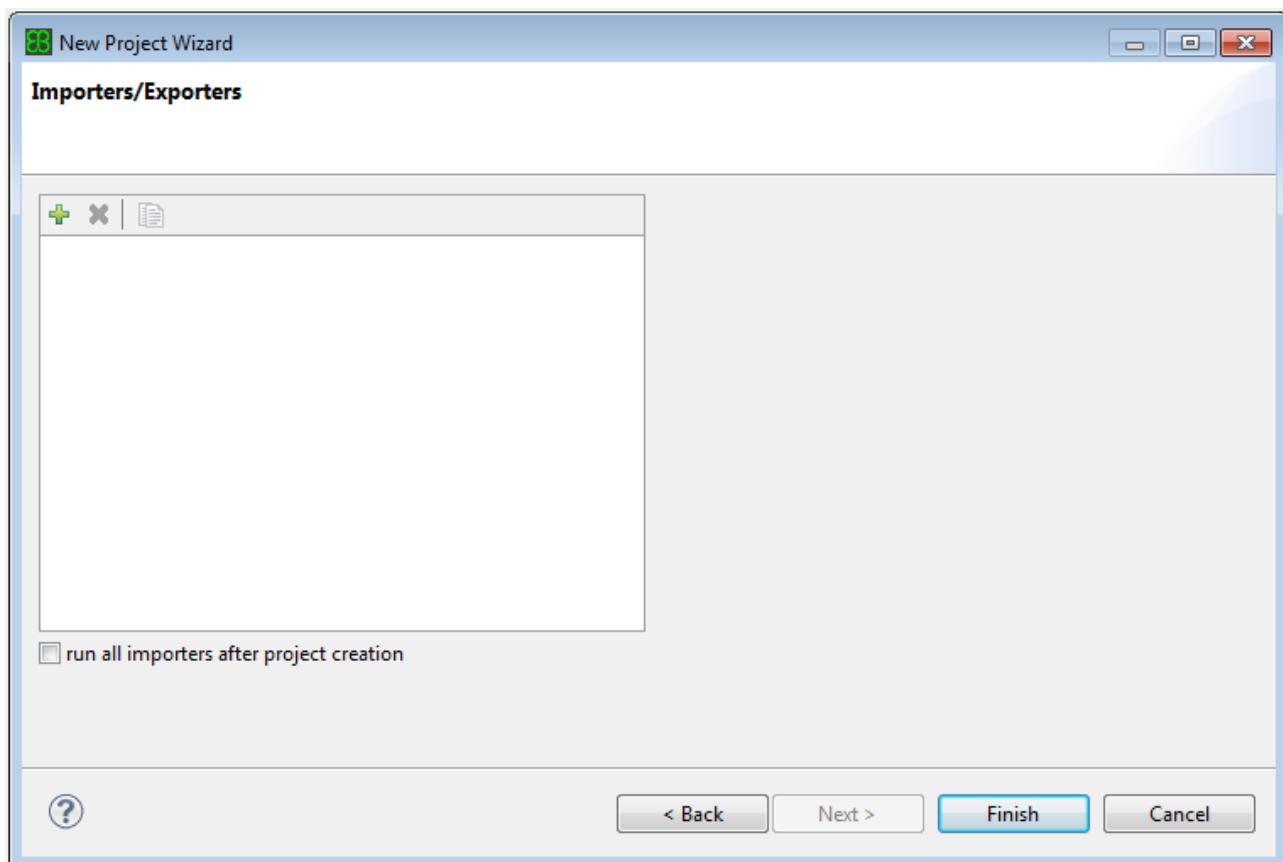


Figure 6.21. Upgrading module configurations in the **Upgrade** dialog

For detailed information on the **Upgrade** dialog, see [Section 6.7, “Upgrading projects and configurations”](#).

6.3.1.5. Step 5: Importing data into the project (optional)

It is possible to import configuration data into the project directly when it is created. This step is optional, you can also import the data later, or not at all. See [Section 6.10.2, “Importing and exporting configuration data”](#) for details on how to do this. If you want to skip this part, continue directly to the next section.



6.3.1.6. Step 6: Finishing set-up of the new project

To finish setting up the new project:

- ▶ Click the **Finish** button. The wizard closes and you see the main window again. If you are creating a project with a lot of modules or if you are importing big files the creation may take a while and a dialog with a progress bar may appear. If the work is completed, the progress bar dialog closes.

The new project is now visible in the **Project Explorer**. It is automatically loaded and you can start editing the module configurations.

6.3.2. Saving a project

If you have changed a project, e.g. by editing one of the module configurations, you need to save the changes to disk. Each module configuration is saved to a separate file. The location of the file has been set when the module was added to the project.

To save a project



- ▶ Select the project node in the **Project Explorer**.
- ▶ Right-click on the node.
A context menu opens up.
- ▶ Select **Save**

6.3.3. Renaming a project

If you want to change the name of a project after it has been created, you can rename it.

To rename a project

- ▶ Select the project node in the **Project Explorer**.
- ▶ Right-click on the node.
A context menu opens up.
- ▶ Select **Rename** and type in a new name.
The project now appears in the project explorer under its new name.

6.3.4. Copying a project

To copy a project:

- ▶ Right-click on the project node you want to copy.
A context menu opens up.
- ▶ Select the **Copy** entry
The project is copied to the clipboard.

To paste the project under a new name:

- ▶ Right-click in the **Project Explorer**.
A context menu opens up.
- ▶ Select the **Paste** entry.

A dialog opens up and prompts you for a name of the new project. Since the project name must be unique you have to decide on a new name. After entering the name, click the **OK** button.

The dialog closes and the new project is visible in the **Project Explorer**.

**WARNING**

If you manually changed any paths of the project to absolute paths, both projects now work on the same directories. This may be the case if you changed e.g. the output directory, or the location of the config files during project creation or in the project properties. If you did not change anything, only directories relative to the project directory are used and both project are completely separated from another.

6.3.5. Deleting a project

It is possible to delete a project. Deleted projects cannot be opened again.

To delete a project

- ▶ Select the project node in the **Project Explorer**.
- ▶ Right-click on the node.
A context menu opens up.
- ▶ Select **Delete**. A dialog opens up asking you if you really want to delete the project. Additionally, there are two radio buttons in the dialog which let you make a choice:
 - ▶ if you decide to **also delete contents ...**, the project will completely be removed from disk. The disk space will be freed, but you will have no way of restoring the project.
 - ▶ if you decide to **do not delete contents**, all associations to the project inside EB tresos Studio are removed, but the project remains on disk (inside the `workspace` directory). If you want to restore the project you can do so by importing it again into EB tresos Studio (see [Section 6.10.1.1, “Importing a project”](#)).

6.3.6. Closing and opening projects

By default the projects are always open in EB tresos Studio. If you are working with several very big projects and if you want to reduce the memory footprint and increase the performance of EB tresos Studio you can close the projects that you currently do not need.

Closed projects only use very few resources, its content is not loaded and you cannot edit their configurations or generate code for them. A closed project needs to be opened and then loaded to be fully functional. More information about the state of a project can be found in [Section 5.3.1.1, “Project and ECU configuration states”](#).

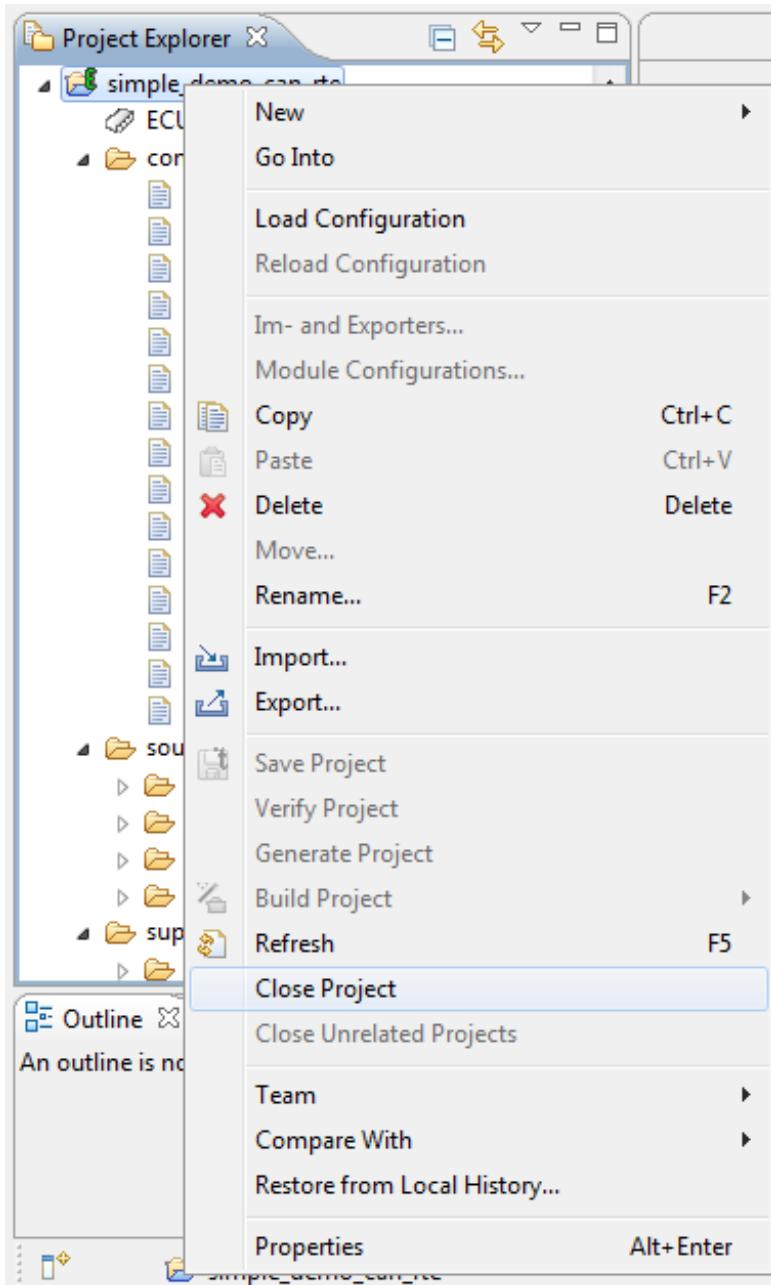
6.3.6.1. Closing projects

To close a project:



- ▶ Select the project node in the **Project Explorer**.
- ▶ Right-click on the project node.

A context menu opens up.



- ▶ Select **Close Project**

The project selected is being closed.

To close all projects that are not referenced by your current project:



- ▶ Select the project node you want to keep open in the Project Explorer.
- ▶ Right-click on the node.

A context menu opens up.

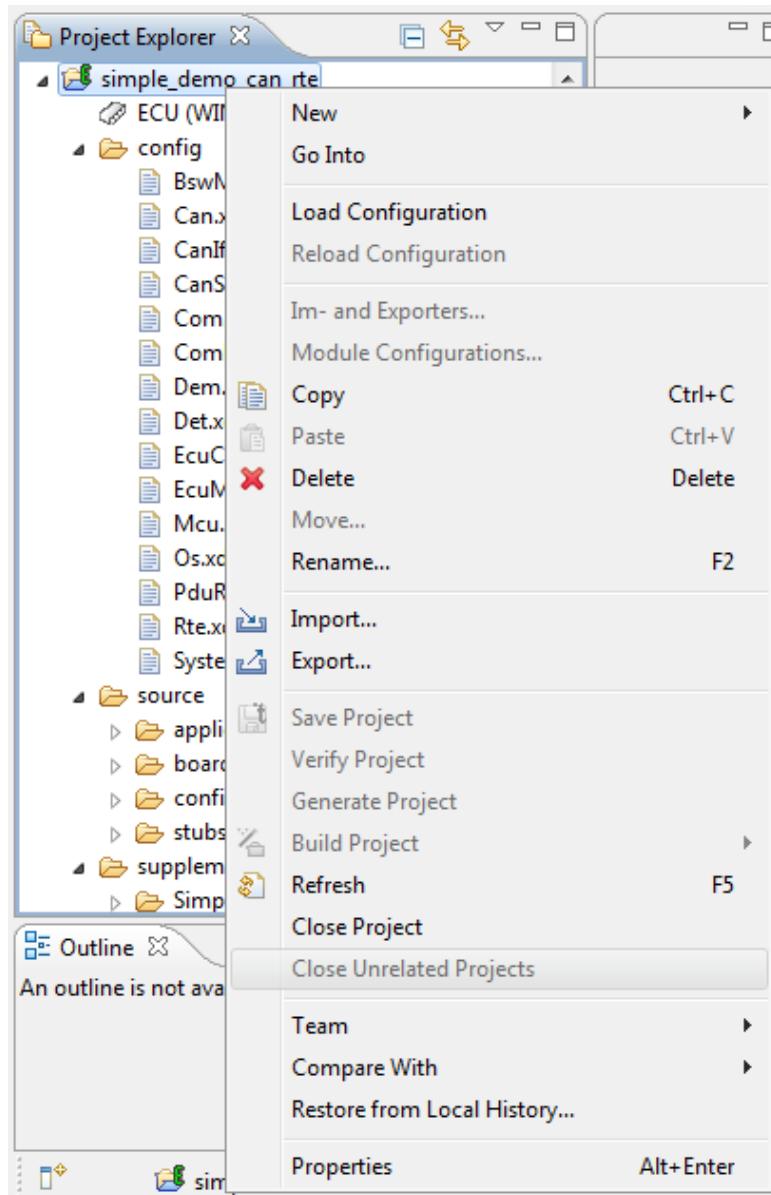


Figure 6.22. Closing all projects that are not referenced by the current project

- ▶ Select **Close Unrelated Projects**.

All projects that are not referenced by your current project are closed.



6.3.6.2. Opening a project

To open a closed project:

- ▶ Right-click on the closed project to open the **Project Explorer**'s context menu.
- ▶ Select **Open Project**.

The project is now open but yet unloaded.

6.3.6.3. Loading a project

To load an open project take one of the following two steps:

- ▶ Alternative 1: Double-click on the ECU configuration node

The project starts loading its configuration.

- ▶ Alternative 2: Right-click on the ECU configuration node to open the Project Explorer's context menu, then select **Load Configuration**

The project starts loading its configuration.

6.3.6.4. Reloading a project

If you modify any XDM file of a loaded project, e.g. by using the team collaboration support feature (see [Section 6.4, "Using team collaboration support"](#)), you must reload the project.

WARNING**Data loss**

It is strongly recommended to modify XDM files only if the project is *not* loaded yet.

If you do edit XDM files in loaded projects, you must reload the project.

Otherwise all intermediate changes on the XDM files will be lost, when you save the configuration project the next time.

6.3.6.4.1. Reloading projects automatically

EB tresos Studio automatically reloads a loaded project in the following cases:

- ▶ If you modify any XDM file of a loaded project, e.g. by using the team collaboration support (see [Section 6.4, "Using team collaboration support"](#)) or by using the integrated text based file editor of EB tresos Studio.



- ▶ If you modify any XDM file of a loaded project from outside of EB tresos Studio, e.g. by using any text editor, and if you have set the EB tresos Studio preference **General -> Workspace -> Refresh automatically**. For further information on preferences settings, see [Section 6.2.2, “Setting preferences”](#).

6.3.6.4.2. Reloading projects manually

In some cases it is useful to manually reload an already loaded project, for instance after you have canceled an automatic reload, because there are unsaved changes in your ECU configuration project.

To manually reload a loaded project:

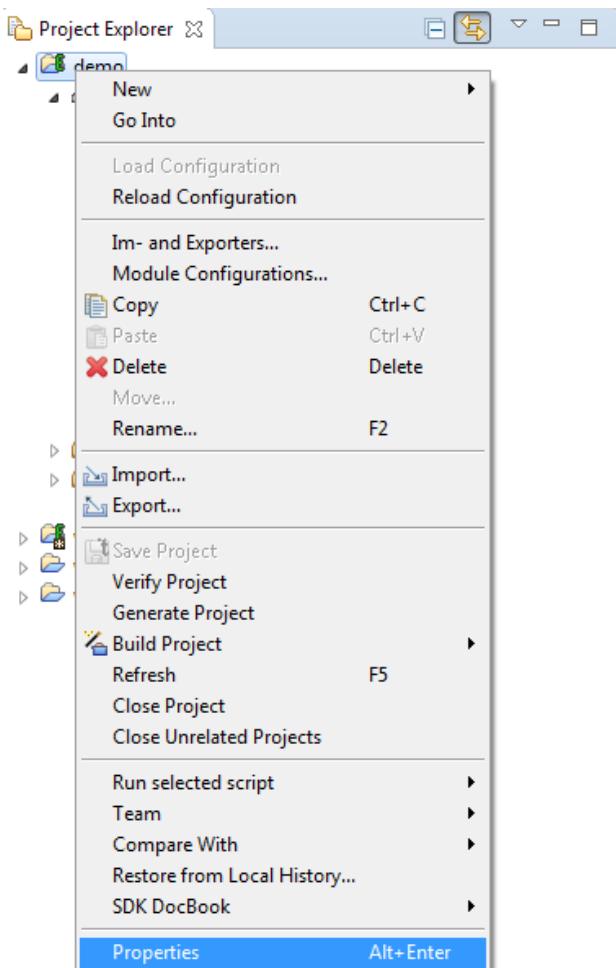
- ▶ Right-click the ECU configuration in the **Project Explorer** view. A context menu opens up.
- ▶ In the context menu, select **Reload Configuration**.

The project starts reloading its configuration.

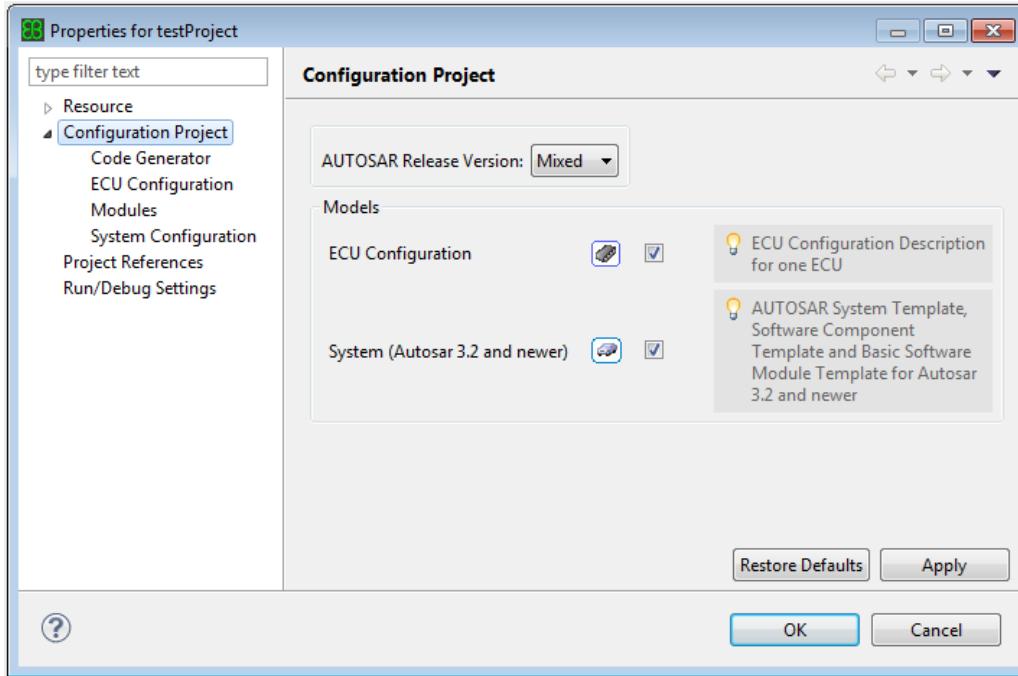
6.3.7. Viewing and changing project properties

The data you have entered in the **New Project** wizard can be viewed and changed. This is done in the **Properties for project name** dialog. To open this dialog:

- ▶ Select the project in the Project Explorer.
- ▶ Right-click on the project.
A context menu opens up.
- ▶ Select **Properties** from the menu.



The **Properties** for project name dialog opens up.

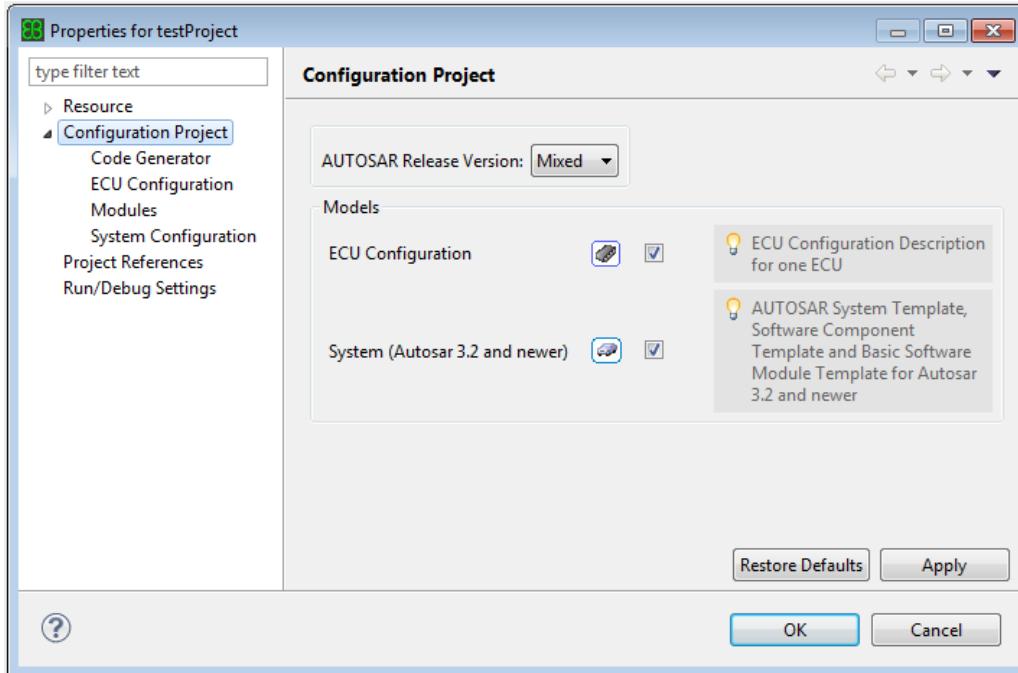


The **Properties for project name** dialog consists of a tree on the left and a main area. The available properties are divided into different categories which make up the items of the tree on the left. Some categories contain subcategories. If you choose a category by selecting an item of the tree, the properties of that category are displayed in the main area of the dialog. The categories which are relevant for the work with EB tresos Studio are grouped below **Configuration Project**. You can expand this tree item to see more categories.

6.3.7.1. Listing the supported models of the project

On the **Configuration Project** page of the project properties you can see which models are enabled in this project. The **ECU Configuration** model is always enabled. The **System** is automatically enabled when an **Rte** module is present in your project. You can also enable the **System** manually without an **Rte** module present, but currently this does not have any effect. In the **Models** section you can choose the **System** matching to the **AUTOSAR Release Version**.

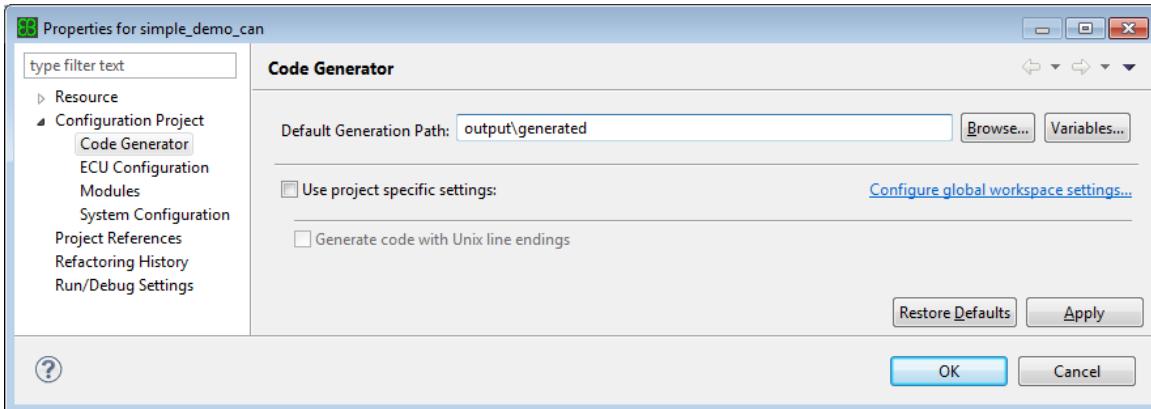
- ▶ In the **AUTOSAR Release Version** drop-down list box you may change the AUTOSAR version of your project.



6.3.7.2. Editing the Code Generator settings

The properties on the **Code Generator** page affect the behavior of code generators. To open this page:

- ▶ Select **Code Generator** from the tree view on the left.



- ▶ In the **Default Generation Path** text box, specify the path into which the generated code files are written to by default, i.e. unless you choose specific generation paths for your module configurations. To specify generation paths per module, see [Section 6.3.7.4.9, “Editing the code generation output path settings”](#).

If the value is a relative path, the path is considered to be relative to the project's directory. If you did not change the generation path during the creation of the project, the value is `output`. If you want to change the value, you may enter an absolute or a relative path. You can also use the predefined variables that



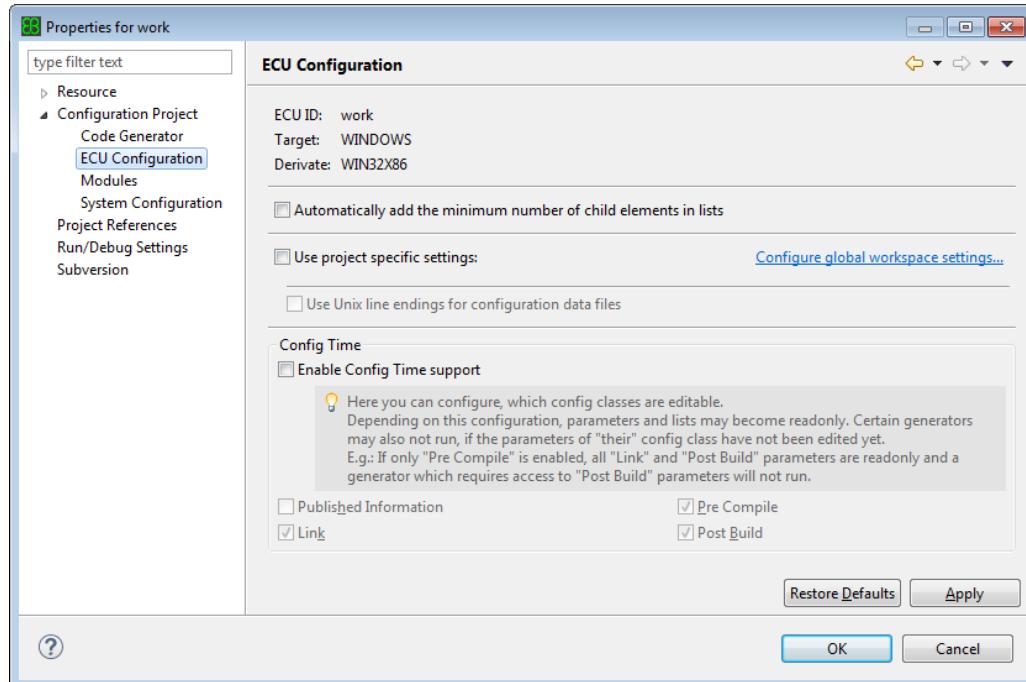
are available by clicking the **Variables** button. There are variables defining the workspace directory, the tresos installation directory and so on.

- ▶ If the **Use project specific settings** check box is not set, the global workspace-wide value for this property is used. You can view and change this setting by clicking on the link **Configure global workspace settings** (see [Section 6.2.2.5, “Setting configuration preferences”](#)).
- ▶ If you want to change the unix line endings for generated code just for this project:
 - ▶ Select **Code Generator** from the tree view.
 - ▶ Check the **Use project specific settings** check box.
 - ▶ Check the **Generate code with Unix line endings** check box if you want to have Unix line endings in your generated files. Uncheck the check box if you want to have Windows line endings in your files.

6.3.7.3. Editing the ECU Configuration settings

The **ECU Configuration** page allows you to change settings that influence the way how ECU configurations are handled by the tool. To open this page:

- ▶ Select **ECU Configuration** from the tree view on the left.



The main part of the dialog provides you with several options:

- ▶ You can view the **ECU ID**, **Target** and **Derivate** settings. These values have been set when the project was generated (see [Section 6.3.1.3, “Step 3: Configuring project properties”](#)) and cannot be changed.



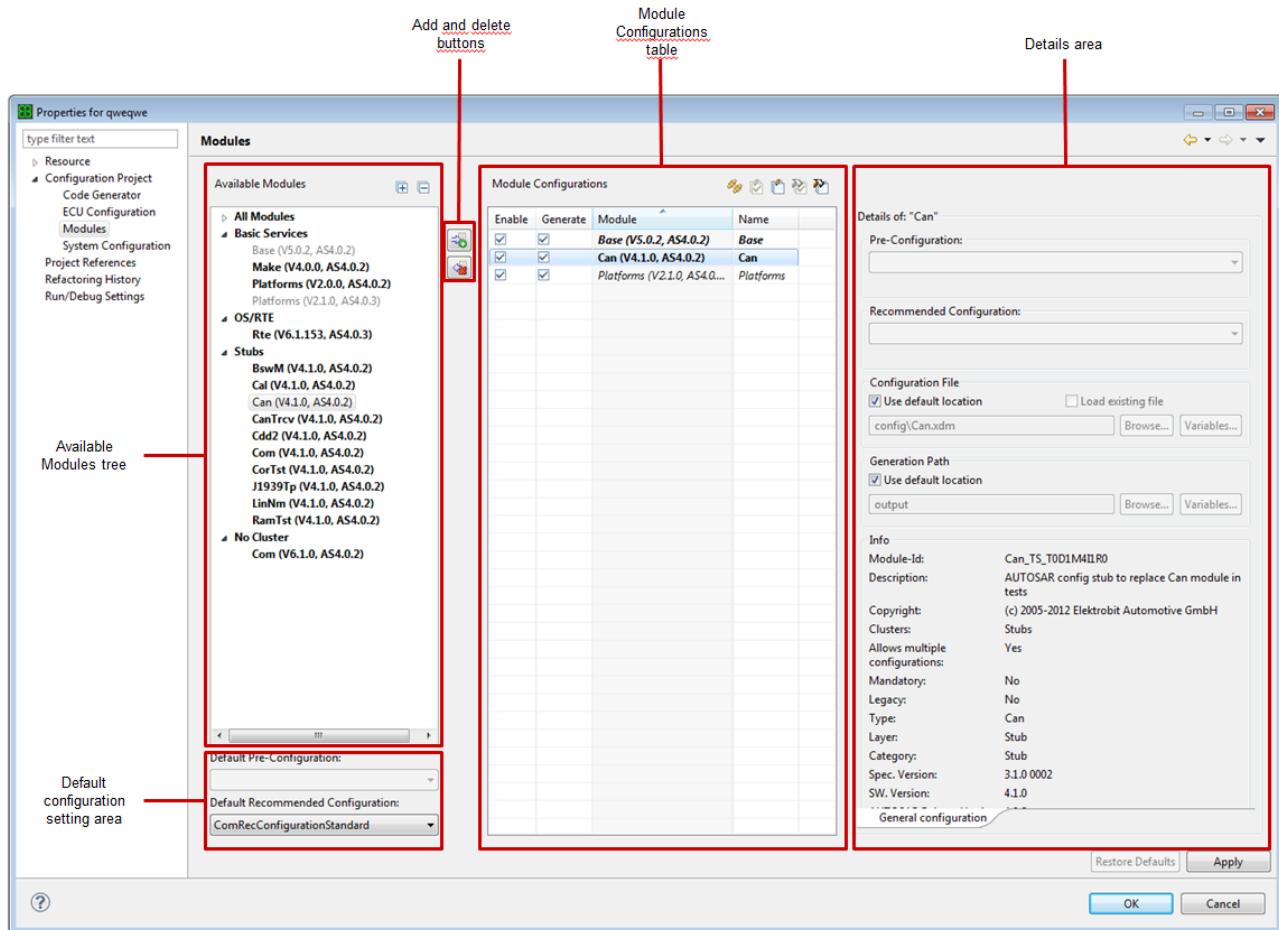
- ▶ If you want lists to be filled with default entries, select the **Automatically add...** check box. If the check box is cleared, the list elements will not be added automatically.
- ▶ You can change which line endings the configuration data files use. If the **Use project specific settings** check box is not set, the global workspace-wide value for this property is used. You can view and change this setting by clicking on the link **Configure global workspace settings** (see [Section 6.2.2.5, "Setting configuration preferences"](#)).

To change the setting for Unix line endings for configuration data files in this project independently of the global settings in EB tresos Studio, check **Use project specific settings**. Then change **Use Unix line endings for configuration data files** to the value you would like to have in this project.

- ▶ You can change the settings of the configuration time support. These values have been initially set when the project was created and can also be changed here (see [Section 6.3.1.3, "Step 3: Configuring project properties"](#)).

6.3.7.4. Viewing information about the project's modules

To view and edit the module configurations of your project, select **Modules** from the tree view at the left. The **Modules** page opens up.

Figure 6.23. Editing module configurations on the **Modules** page of the project properties

6.3.7.4.1. Adding module configurations to your project

You may add single module configurations, several module configurations at once or clusters of module configurations to your project.

NOTE



You can recognize whether a module has been added by its font

Modules that are still available and modules that already have been added to your project are displayed in different font styles in the **Available Modules** tree:

- ▶ Modules that have not been added to your project and thus are still available are displayed in bold font.
- ▶ Modules that already have been added to your project are displayed in standard font. The modules that allow multiple configurations are displayed in black. All others are grayed-out and cannot be added to your project anymore.

To add new module configurations to your project:



- ▶ In the **Available Modules** tree, select all modules/clusters of modules for which you want to add configurations for.
- ▶ Click the **Add** button .

TIP**Add single modules with a double-click**

Module configurations for single modules can also be added by double-clicking the required module.

- ▶ If the required module configurations cannot be added to your project, the following dialog opens up:

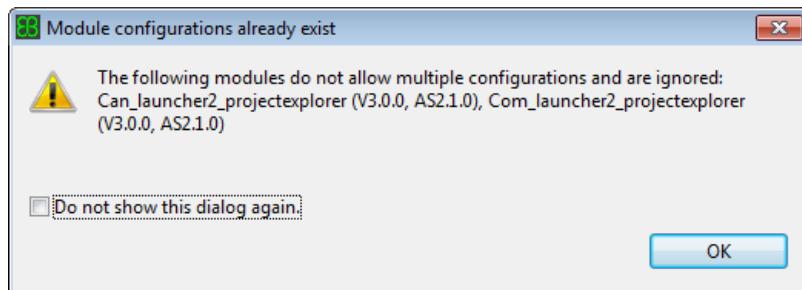


Figure 6.24. Some modules may be added only once to your project

The selected modules are added to your project as module configurations with default settings.

NOTE**Only one module configuration is added per selected module**

Since modules can be part of more than one cluster, it is possible that a selection in the **Available Modules** tree refers several times to the same module. Note that only one configuration is added in this case, even if the module allows multiple configurations.

As long as the recently added module configurations have not been stored into your project yet, you may edit the (default) settings of the module configuration. Mind that depending on the module, there might be more tab pages next to the **General configuration** tab on the bottom of the **Details** area.

6.3.7.4.2. Removing module configurations from your project

You may remove single module configurations or several module configurations at once from your project.

**NOTE****You can recognize the state of a module by its font**

Module configurations that previously had been added to your project, mandatory module configurations and recently added module configurations are displayed in different fonts in the **Modules Configuration** table:

- ▶ Mandatory module configurations are displayed in italic font. It is not recommended to remove these module configurations from your project.
- ▶ Recently added module configurations are displayed in bold font. You may edit them.
- ▶ Module configurations that had been added to your project before are displayed in standard font.

To remove module configurations from your project:

- ▶ In the **Module Configurations** table, select the module configuration(s) you want to remove.

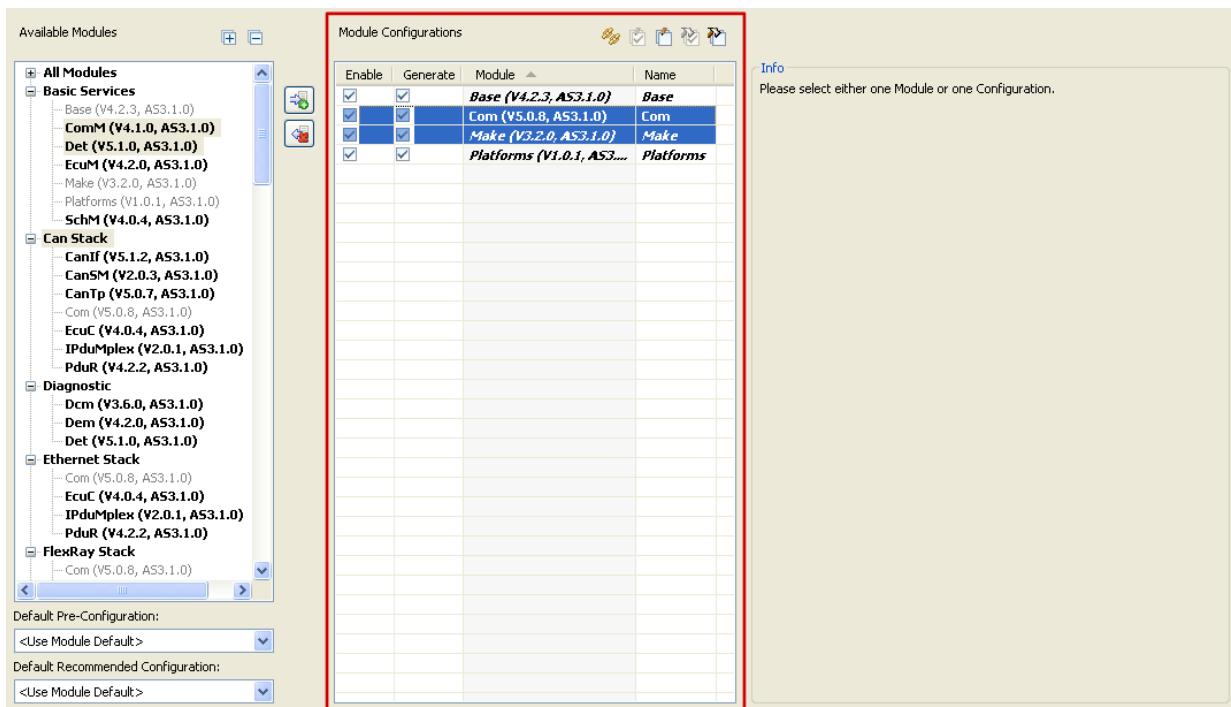


Figure 6.25. Removing module configurations

- ▶ Click the **Delete** button .
- ▶ If none of the selected module configurations refers to an already existing configuration file, the items selected are removed from the **Module Configurations** table without further notice. Otherwise the **Delete Module Configurations** dialog opens up

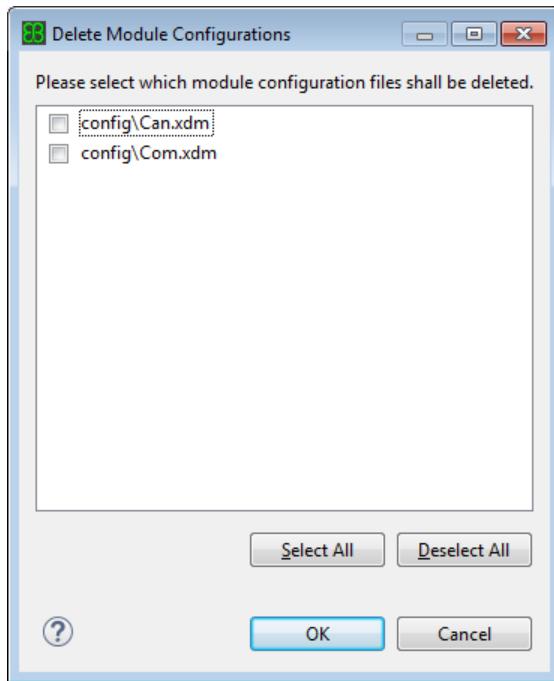


Figure 6.26. Confirming which module configuration files you want to remove

- ▶ Select the check box of the configuration files you want to delete.

NOTE**Data loss**

The selected files will be deleted from the file system when the changed module configurations are applied. After that, you cannot restore these files anymore.

- ▶ To confirm the deletion of the configuration files, click **OK**.

The items selected are removed from the **Module Configurations** table.

- ▶ To cancel the deletion of the **Module Configurations** and their referred configuration files, click **Cancel**.

6.3.7.4.3. Enabling and disabling module configurations

In the **Module Configurations** table you may (temporarily) enable or disable the module configurations you have added to your project. Disabled configurations are treated as if they do not exist, which means you cannot edit or verify them, and code generators cannot access them. To enable module configurations:

- ▶ To enable only one module configuration, select the **Enable** check box of the module configuration you want to enable in the **Module Configurations** table.



Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.27. Enabling one module configuration

- ▶ To enable several module configuration at once:
 - ▶ In the **Module Configurations** table, select the module configurations you want to enable.
 - ▶ In the tool bar of the **Module Configurations** table, click the **Enable module configuration** button .

To disable module configurations:

- ▶ To disable only one module configuration in the **Module Configurations** dialog/page, clear the **Enable** check box of the module configuration you want to disable in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.28. Disabling one module configuration

To disable several module configuration at once:

- ▶ In the **Module Configurations** table, select the module configurations you want to disable.
- ▶ In the tool bar of the **Module Configurations** page, click the **Disable module configuration** button .

6.3.7.4.4. Enabling and disabling code generation for module configurations

In the **Module Configurations** table you may enable or disable code generation for the module configurations of your project.

**WARNING****Only partly generated code leads to incorrectly working ECUs**

If you only generate the code of a part of the module configurations you later want to implement on an ECU, depending on which modules you choose, the code might not work correctly. This might lead to incorrectly working ECUs.

To avoid incorrectly working ECUs, generate the code of all module configurations in your project before deploying them on an ECU.

To enable code generation for module configurations:

- ▶ To enable code generation for only one module configuration, select the **Generate** check box of the module configuration for which you want to enable code generation in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.29. Enabling code generation for one module configuration

- ▶ To enable code generation for several module configurations at once:
 - ▶ In the **Module Configurations** table, select the module configurations, for which you want to enable code generation.
 - ▶ In the tool bar of the **Module Configurations** page, click the **Enable code generation** button

To disable code generation for module configurations:

- ▶ To disable code generation for only one module, clear the **Generate** check box of the module configuration for which you want to disable code generation in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input type="checkbox"/>	<input type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.30. Disabling code generation for one module configuration

- ▶ To disable code generation for several module configurations at once:



- ▶ In the **Module Configurations** table, select the module configurations, for which you want to disable code generation.
- ▶ In the tool bar of the **Module Configurations** page, click the **Disable code generation** button

6.3.7.4.5. Naming Module Configurations

When you add new module configurations to your project, the name of the new module configuration is set automatically to a name that is unique within the project using either the default configuration name provided by the module or the module name itself.

Instead of using the default name you may want to use another name for the module configuration.

NOTE
Only in recently added module configurations the name can be changed


You can only change the name of a module configuration if you added this module configuration during the same session of the dialog.

To edit the name of a module configuration:

- ▶ Add the required module configuration to your project.
- ▶ In the **Module Configurations** table, click into the **Name** cell of the module configuration whose name you want to change.
- ▶ Type the new name.



Figure 6.31. Editing the name of a new module configuration

NOTE

Name constraints

Mind that the module configuration name must meet the following constraints:

- ▶ The name must be unique within the project.
- ▶ The name must start with a letter.
- ▶ The name must contain only letters and digits plus the underscore character ('_').
- ▶ The name must not be empty.
- ▶ The name must not exceed 255 characters.

- ▶ Press **ENTER**.

The name of the module configuration is changed.



6.3.7.4.6. Choosing Pre-Configurations

Module suppliers optionally can equip modules with one or more preconfigurations that contain values for configuration parameters. If a module configuration of your project provides one or more preconfiguration(s), you must assign this module configuration to such a preconfiguration when adding it to your project. The parameters covered by the chosen preconfiguration are then fixed in the module configuration and cannot be edited by the user.

In the **Default configuration setting** area, you can set the **Default Pre-Configuration** which is used as a default value for all new module configurations. To support a more consistent selection, this preconfiguration setting is persisted. You can also edit the **Pre-Configuration** of an individual module configuration.

6.3.7.4.6.1. Setting the Default Pre-Configuration

In the **Default configuration setting** area of the **Module Configurations** page, you may edit the **Default Pre-Configuration** for new module configurations.

NOTE**The value set is a union set of all preconfigurations provided by all modules**

The preconfiguration names to choose from derive from the union set of all names provided by all modules that match the project's target architecture and AUTOSAR release version. If the chosen name is not valid for a specific module, the module's default preconfiguration is taken instead when you add it to your project.

To change the default preconfiguration, select the required preconfiguration in the **Default Pre-Configuration** drop-down list box.

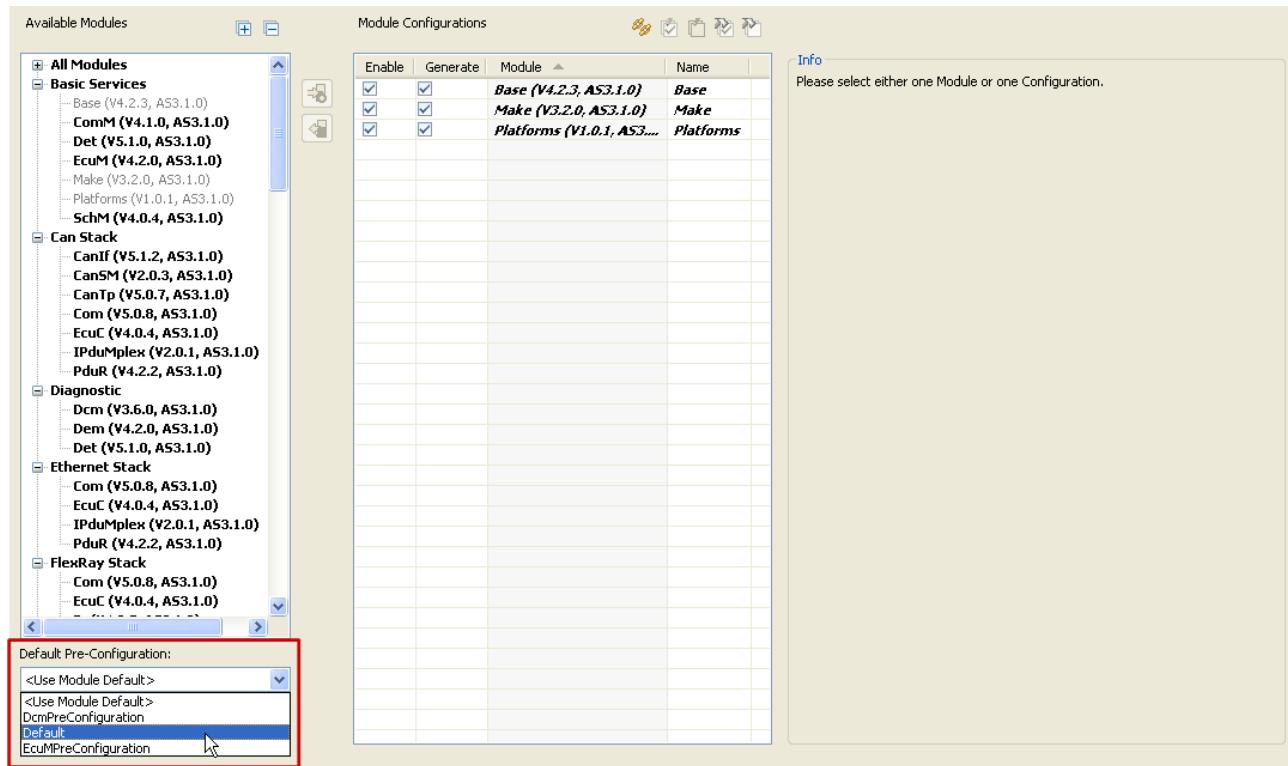


Figure 6.32. Selecting the default preconfiguration for new module configurations

- ▶ To use the respective default preconfiguration for new module configurations, select <Use Module Default>.

NOTE**Non-compilable code**

If you select <Use Module Default>, every module will be added to your project with a different preconfiguration. This might lead to incompatibilities of the module configurations and thus to non-compilable code.

- ▶ The selected default preconfiguration is saved, whenever you actually add new module configurations to your project.

6.3.7.4.6.2. Changing the preconfiguration of one specific module configuration

To change the preconfiguration of one specific module configuration:

- ▶ Add the required module configuration to your project.

**NOTE****Only in recently added module configurations the preconfiguration can be changed**

You can only change the preconfiguration of a specific module configuration if you added this module configuration during the same session of the dialog.

- ▶ In the **Module Configurations** table, mark the module configuration whose preconfiguration you want to change.

In the **Details** area of the **Module Configurations** page, detailed information about the specific module is displayed.

The screenshot shows the 'Module Configurations' dialog. On the left, the 'Available Modules' tree view is expanded to show categories like 'All Modules', 'Basic Services', 'Diagnostic', 'FlexRay Stack', 'Mode Management Stack', 'OS/RTE', and 'Stubs'. Under 'All Modules', several sub-modules are listed with their versions (e.g., Base (V5.0.2, AS4.0.2), Com (V6.1.0, AS4.0.2)). On the right, a table lists 'Module Configurations' with columns for 'Enable', 'Generate', 'Module', and 'Name'. The 'Com (V6.1.0, AS4.0.2)' row is selected. A red box highlights the 'Details' area on the right, which contains detailed information for the selected module:

Details of: "Com"	
Pre-Configuration:	<input type="text"/>
Recommended Configuration:	ComRecConfigurationStandard
Standard recommended configuration: All Autosar features are enabled	
Configuration File:	<input checked="" type="checkbox"/> Use default location <input type="checkbox"/> Load existing file <input type="text"/> config\Com.xdm <input type="button"/> Browse... <input type="button"/> Variables...
Generation Path:	<input checked="" type="checkbox"/> Use default location <input type="checkbox"/> Load existing file <input type="text"/> output <input type="button"/> Browse... <input type="button"/> Variables...
Info	
Module-Id:	Com_TS_TxDxM6I1R0
Description:	AUTOSAR Com Module
Copyright:	(c) 2005-2012 Elektrobit Automotive GmbH
Clusters:	FlexRay Stack
Allows multiple configurations:	No
Mandatory:	No
General configuration	

Figure 6.33. The **Details** area in the **Module Configurations** page

- ▶ In the **Pre-Configuration** drop-down list box in the **Details** area, select the required preconfiguration for the selected module configuration.

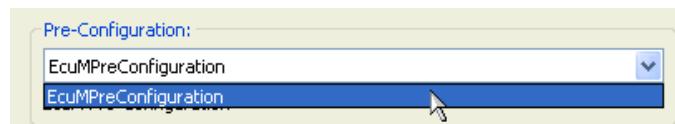


Figure 6.34. Changing the preconfiguration of one specific module

**NOTE****Grayed-out preconfigurations are not available**

If the **Pre-Configuration** drop-down list is grayed-out, the selected module does not provide a preconfiguration of its own.

6.3.7.4.7. Applying recommended configurations

Module suppliers optionally can equip modules with one or more recommended configurations that contain values for configuration parameters. If a module configuration of your project provides one or more recommended configurations, you can choose one when you add a configuration for this module to your project. The parameter values covered by the chosen recommended configuration can be considered as a guideline. They may be edited by the user later.

In the **Default configuration setting area**, you can set the **Default Recommended Configuration** which is used as a default value for all new module configurations. To support a more consistent selection, this setting is persisted. You can also edit the **Recommended Configuration** of an individual module configuration.

6.3.7.4.7.1. Setting the Default Recommended Configuration

In the **Default configuration setting area** of the **Module Configurations** page, you may edit the **Default Recommended Configuration** for new module configurations.

NOTE**The value set is a union set of all recommended configurations provided by all modules**

The names of the recommended configurations to choose from derive from the union set of all names provided by all modules that match the project's target architecture and AUTOSAR release version. If the chosen name is not valid for a specific module, the module's default configuration is taken instead when you add it to your project. If the module does not provide a default configuration, no recommended configuration is applied.

To change the default recommended configuration, select the required recommended configuration in the **Default Recommended Configuration** drop-down list box.

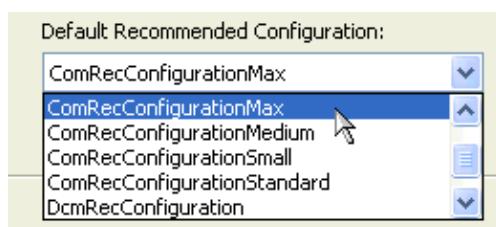


Figure 6.35. Selecting the default recommended configuration for new module configurations



- ▶ If you don't want to apply any of the recommended configurations, select <None>.
- ▶ To use the respective default recommended configuration for new module configurations, select <Use Module Default>.

NOTE**Non-compilable code**

If you select <Use Module Default>, every module will be added to your project with a different recommended configuration. This might lead to incompatibilities of the module configurations and thus to non-compilable code.

- ▶ The selected default recommended configuration is saved, when you actually add new module configurations to your project.

6.3.7.4.7.2. Changing the recommended configuration of one specific module configuration

To change the recommended configuration of one specific module configuration:

- ▶ Add the required module configuration to your project.

NOTE**Only in recently added module configurations the recommended configuration can be changed**

You can only change the recommended configuration of a specific module configuration if you have added this module configuration during the same session of the dialog.

- ▶ Mark the module configuration whose recommended configuration you want to change.

In the **Details** area of the **Module Configurations** page, detailed information about the specific module is displayed.



Figure 6.36. The Details area in the Module Configurations page

- In the **Recommended Configuration** drop-down list box, select the required recommended configuration for the selected module configuration.

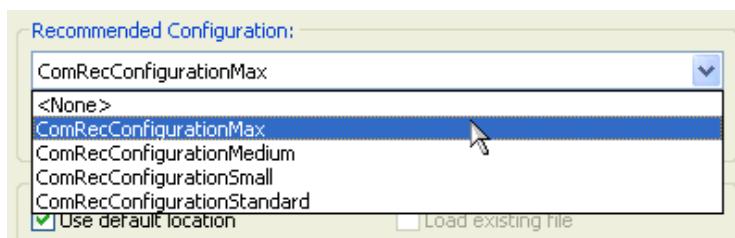


Figure 6.37. Changing the recommended configuration of one specific module

**NOTE****Grayed-out recommended configurations are not available**

If the **Recommended Configuration** drop-down list is grayed-out, the selected module does not provide a recommended configuration of its own.

6.3.7.4.8. Editing the configuration file settings

The configuration file stores all parameters for a specific module configuration. For detailed information on how to edit those parameters, e.g. by using the Generic Configuration Editor, see [Section 6.8, “Editing parameters of a module configuration”](#).

When you add new module configurations to your project, per default the configuration file name is set to config/<moduleconfigurationname>.xdm, relative to your project folder. This is also referred to as the default location.

Instead of using the default location you can use another location or name for the configuration file. To use another location or file name:

- ▶ Add the required module configuration to your project.

NOTE**Only in recently added module configurations the configuration file can be changed**

You can only change the configuration file of a module configuration if you have added this module configuration during the same session of the dialog.

- ▶ In the **Module Configurations** table, mark the module configuration whose configuration file you want to change.
- ▶ In the **Configuration File** panel of the **Details** area, clear the check box **Use default location**.

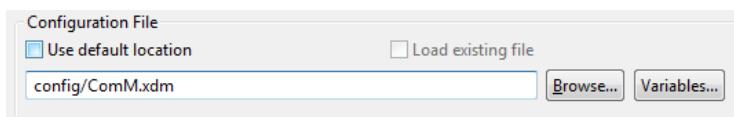


Figure 6.38. Editing the configuration file settings

- ▶ Enter or choose the new file name. Either use absolute path names or path names relative to your project. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#)

**NOTE****The file extension .xdm is mandatory**

You can only use file names ending with the extension `.xdm`. When you choose an existing file, you must make sure that it matches the targeted module configuration schema. Otherwise the new module configuration will fail to load when applied.

6.3.7.4.8.1. Using existing configuration files

Instead of configuring the module configuration from scratch you may also use an already existing configuration file.

NOTE**You cannot set preconfigurations and recommended configurations when you load an existing configuration file**

If you use an existing configuration file, you cannot apply any preconfiguration or recommended configuration.

To load an already existing configuration file into your project:

- ▶ In the **Configuration File** panel, browse to the file you want to load.

NOTE**The selected configuration file will be modified**

The selected configuration file will be modified while you edit the parameters of the module configuration. If you want to preserve the file as it is, copy the file to your project before loading it to your project.

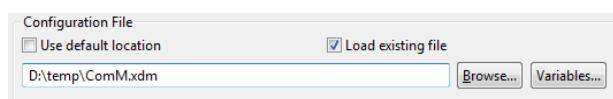


Figure 6.39. Using an existing configuration file

Whenever you point the **Configuration File** panel to an existing file in the file system, the **Load existing file** check box is marked.

NOTE**Data loss**

If you clear the **Load existing file** check box, the existing file will be overwritten and cannot be restored again.

6.3.7.4.9. Editing the code generation output path settings

In **Generation Path**, specify the directory into which EB tresos Studio writes the generated code files for this module configuration.



When you add new module configurations to your project, the **Generation Path** is set to the **Default Generation Path** of the project. This is also referred to as the default location. For more information about the **Default Generation Path** of the project, see [Section 6.3.7.2, “Editing the Code Generator settings”](#).

Instead of using the default location you can use another location for the generated code of this module configuration. To use another location:

- ▶ In the **Module Configurations** table, mark the module configuration whose generation path you want to change.
- ▶ In the **Generation Path** panel of the **Details** area, clear the check box **Use default location**.



Figure 6.40. Editing the generation path settings

- ▶ Enter or choose the new path name. Either use absolute path names or path names relative to your project. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#).

6.3.7.4.10. Upgrading module configurations

To upgrade module configurations from older software versions of the modules to newer ones:

- ▶ Click the **Upgrade** button  in the tool bar of the **Module Configurations** table.

The **Upgrade** dialog opens up.

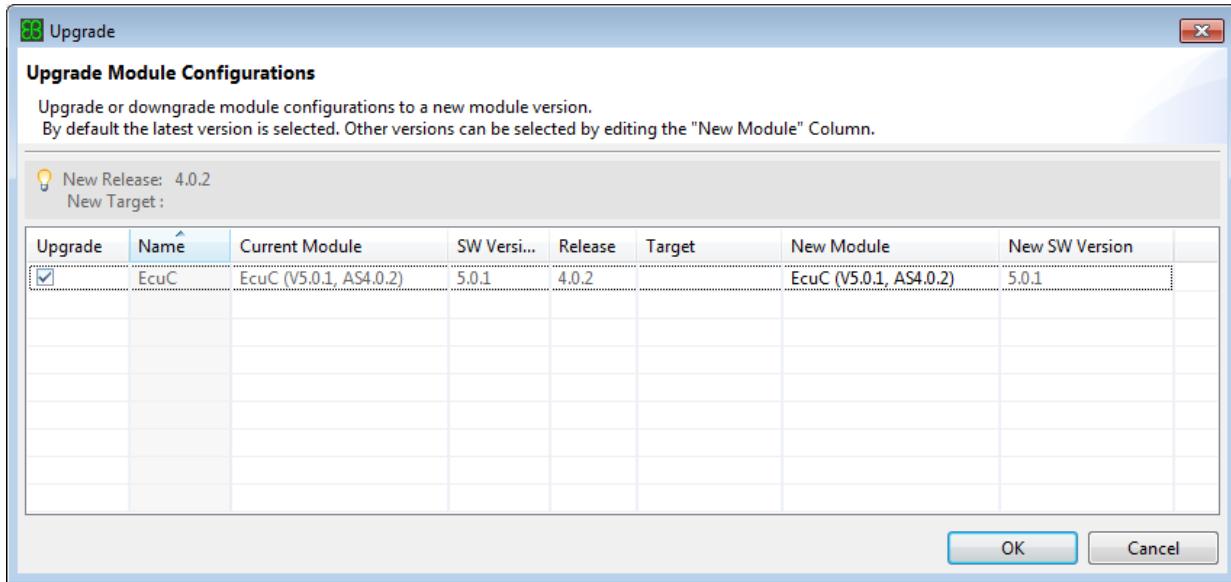


Figure 6.41. Upgrading module configurations in the **Upgrade** dialog

For detailed information on the **Upgrade** dialog, see [Section 6.7, “Upgrading projects and configurations”](#).

6.3.7.5. Editing the System Configuration settings

The **System Configuration** page allows you to configure the selected `System` and `EcuInstance` after you imported AUTOSAR system information into the project's system model. For more information on how to import AUTOSAR system information, see [Section 6.10.2.3.3, “Importing system information”](#). The selected `System` and `EcuInstance` are used e.g. for ECU Extract creation. For more information on how to create the ECU Extract, see [Section 6.11, “Creating an ECU Extract”](#).

To configure the selected `System` and `EcuInstance` of the project:

- ▶ Select **System Configuration** from the tree view on the left.

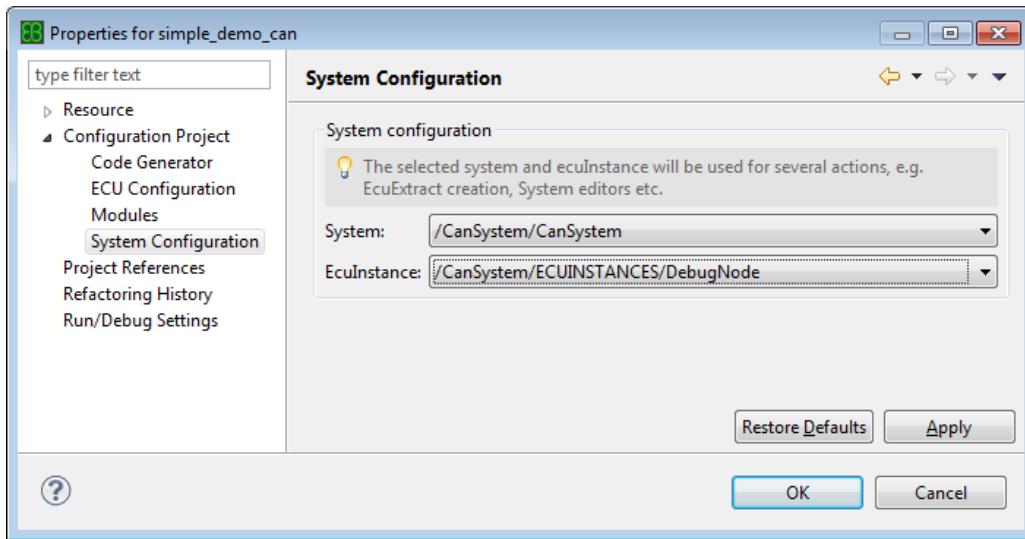


Figure 6.42. Selecting `System` and `EcuInstance` on the **System Configuration** page of the project properties

The dialog contains two drop-down list boxes. You can see which elements of `System` and `EcuInstance` are available in your system model and select one of the values.

Your system model might not provide any `System` and `EcuInstance` elements but only `TopLevelComposition` elements. In this case EB tresos Studio prompts you with a dialog when you select the **System Configuration** page.

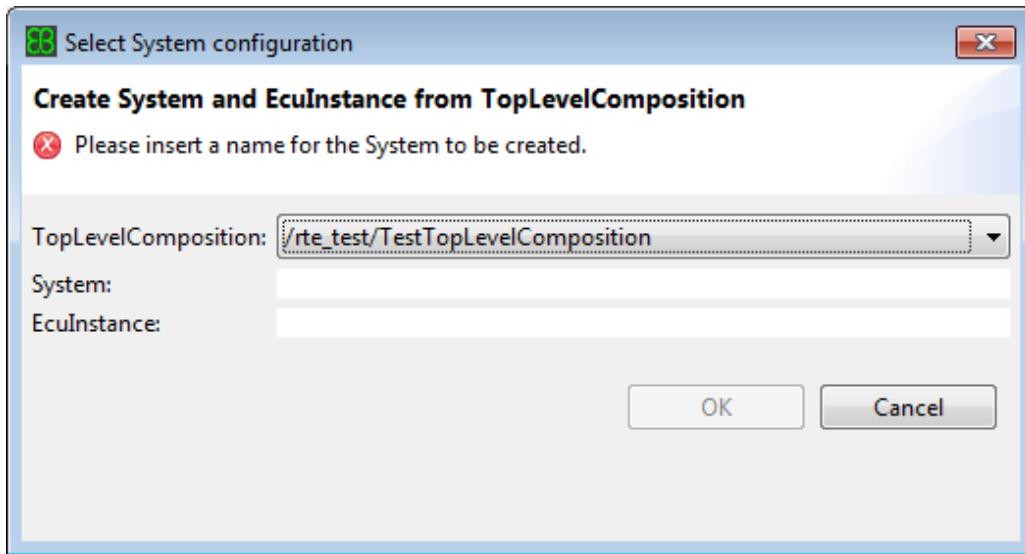


Figure 6.43. Create `System` and `EcuInstance` from a `TopLevelComposition`

You can select one `TopLevelComposition` in the **Select System configuration** dialog. From this `TopLevelComposition` EB tresos Studio derives a `System` together with an `EcuInstance` that is mapped to this `System`.



After you provide names and click the **OK** button, EB tresos Studio creates the new elements in the project's system model and automatically selects them on the **System Configuration** page.

6.4. Using team collaboration support

6.4.1. Overview

This chapter provides instructions for working as a team with EB tresos Studio. This includes instructions on:

- ▶ working with version control systems such as CVS and SVN,
- ▶ synchronizing files and file versions,

This chapter contains instructions for tasks you perform when you work with a team with EB tresos Studio. For detailed background information on the Eclipse CVS support, see the [Eclipse Workbench User Guide](#). For detailed background information on the Eclipse SVN support, see the [Subclipse - Subversion Eclipse Plugin documentation](#). Both of these Eclipse documents are included in the EB tresos Studio onscreen help, which opens up when you press **F1**.

6.4.2. Working with CVS repositories

This chapter is interesting for you if you want to connect your EB tresos Studio with your CVS repository and commit to shared EB tresos Studio projects directly from the EB tresos Studio GUI.

The state (e.g. locally modified, added) of your local files compared to the files in the CVS repository is typically indicated by overlay icons. The following overlay icons are displayed in the **Project Explorer** view:

Project Explorer overlay icons for CVS repository support



Indicates that the file or folder is under version control, with or without local modifications. Versioned files that are locally modified are additionally marked with a > character in front of the name in the **Project Explorer** view, e.g.



Indicates a new file or folder within a version controlled folder that has not yet been added to version control.



6.4.2.1. Registering your CVS repository location in EB tresos Studio

To access your CVS repositories from within EB tresos Studio you first need to register the repository location in EB tresos Studio. To register the repository location in EB tresos Studio:

- ▶ In the **Window** menu, select **Show view**. A context menu opens up.
- ▶ In the context menu that opens up, select **Other....** The **Show View** dialog opens up.
- ▶ Expand **CVS** and select **CVS Repositories**. The **CVS Repositories** view opens up as a tab in the **Properties** view in the lower right corner of the EB tresos Studio workbench.



Figure 6.44. Registering a new repository location in the **CVS Repositories** view

- ▶ To register a new repository to be accessed via EB tresos Studio, click the **Add CVS repository** button . The **Add CVS Repository** dialog opens up.

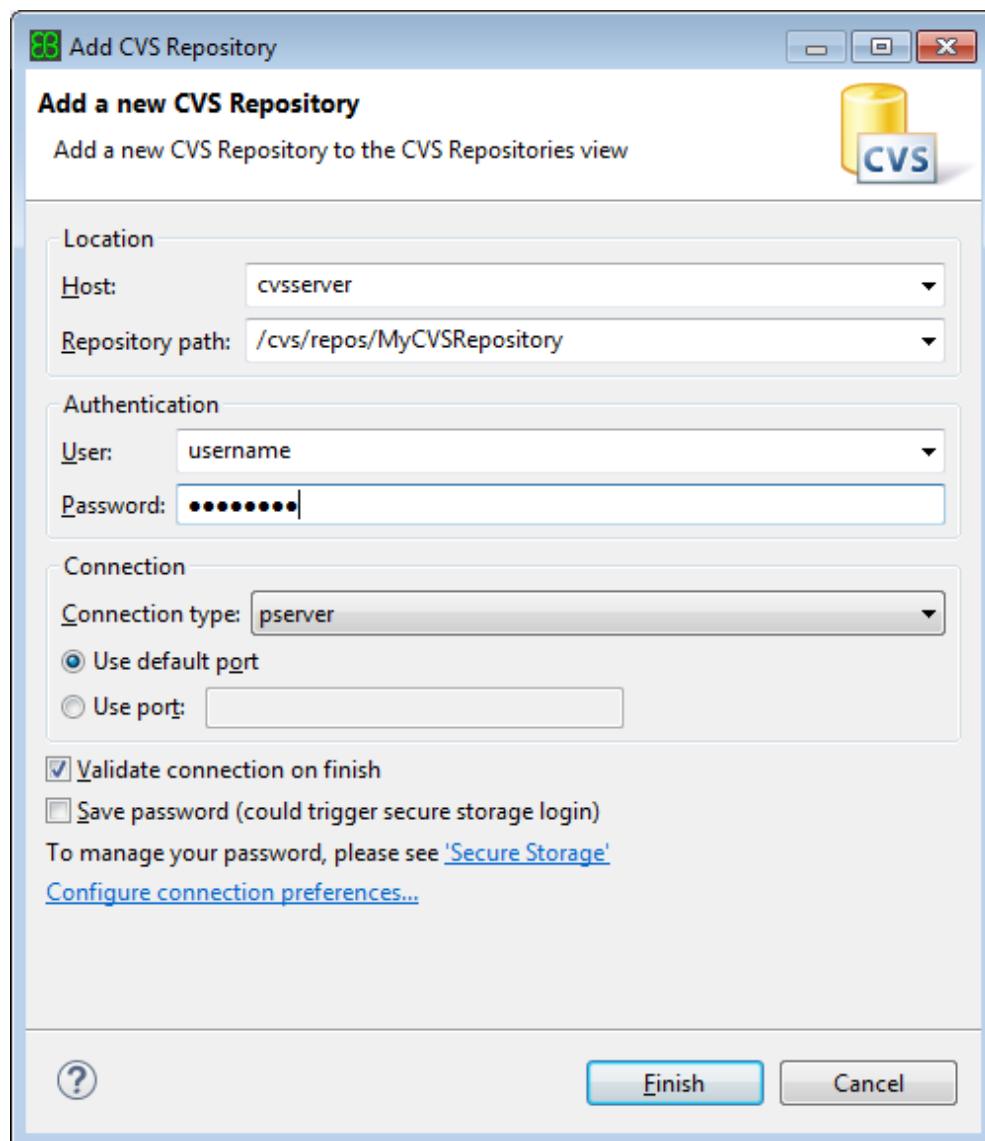
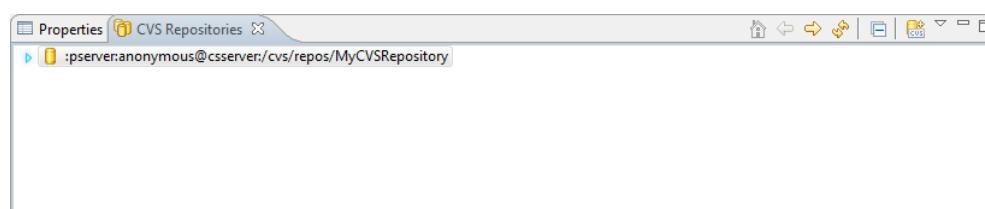


Figure 6.45. Adding a new repository location in EB tresos Studio

- ▶ Enter the connection data. If you do not know your connection data, consult your local CVS repository administrator.
- ▶ Click **Finish**. The repository is displayed in your **CVS Repositories** view.

Figure 6.46. Registered repositories in the **CVS Repositories** view



6.4.2.2. Sharing an EB tresos Studio project with other team members

In this section you will learn how to share a project with other team members. This is equivalent to importing files to a CVS repository and thus only applies to projects that have not yet been added to the repository.

NOTE**You can only share whole projects, not single files**

Adding single files to a repository is not possible from within EB tresos Studio. Thus if you want to share new files, you must share the project they reside in first.

To share a new project:

- ▶ Right-click this project in the **Project Explorer** view, select **Team** and then **Share Project...** from the context menus that open up:

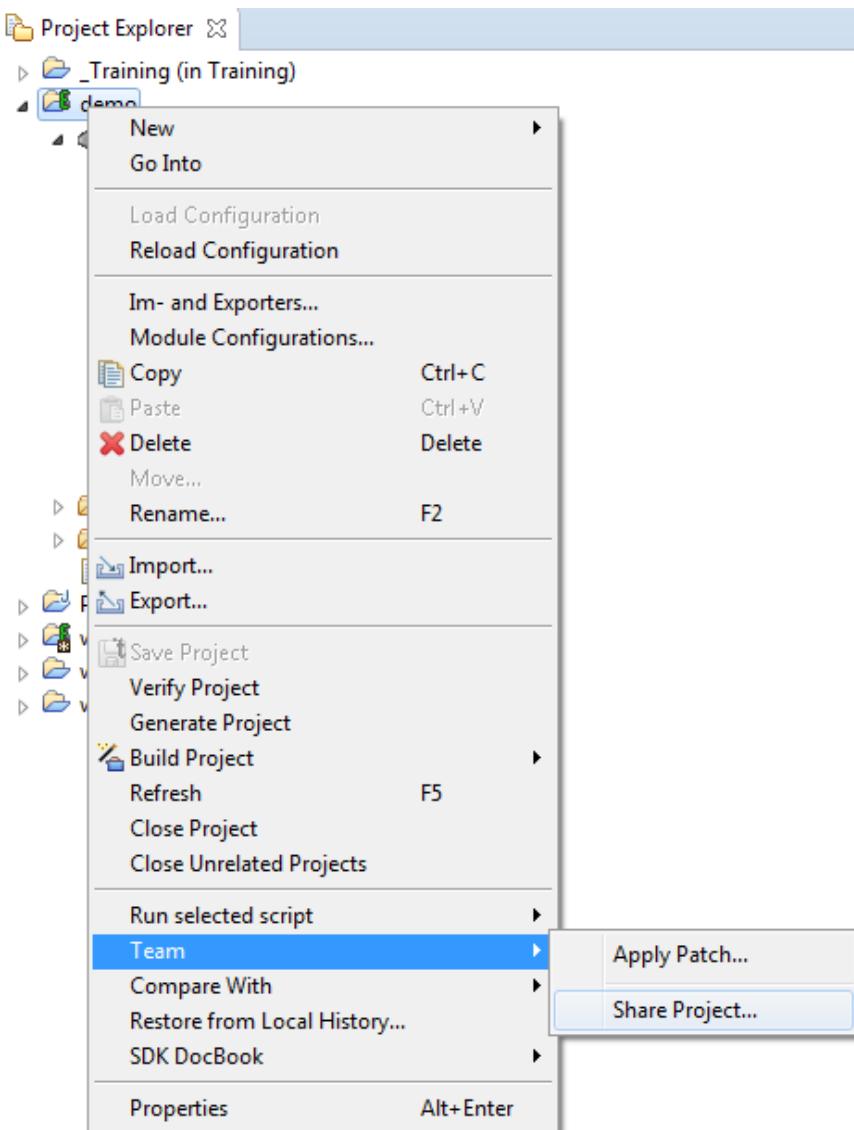


Figure 6.47. Selecting a project to share

- The **Share Project** dialog opens up.

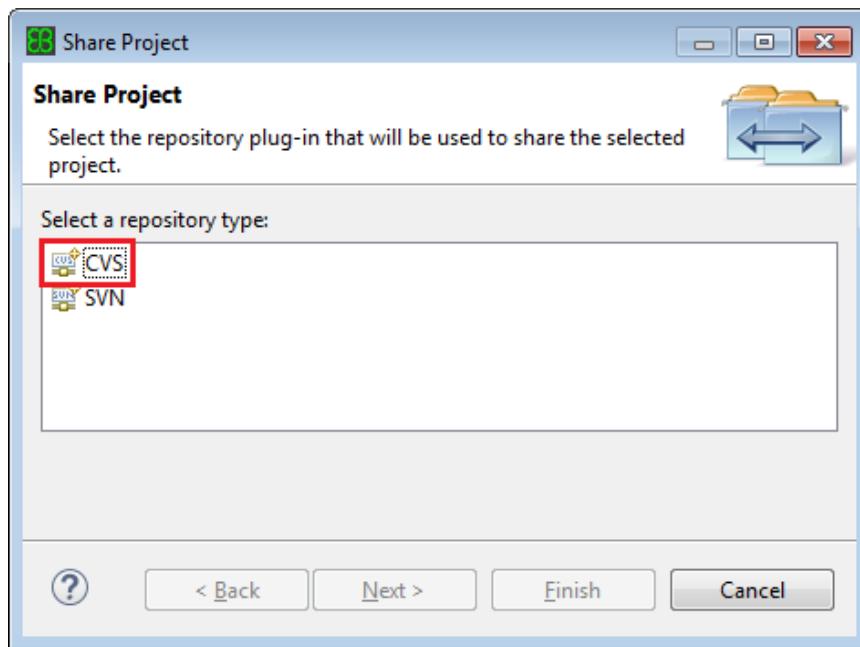


Figure 6.48. Selecting the repository type in which you want to share your project

- Select **CVS** and click **Next**.

The **Share Project with CVS Repository** page opens up

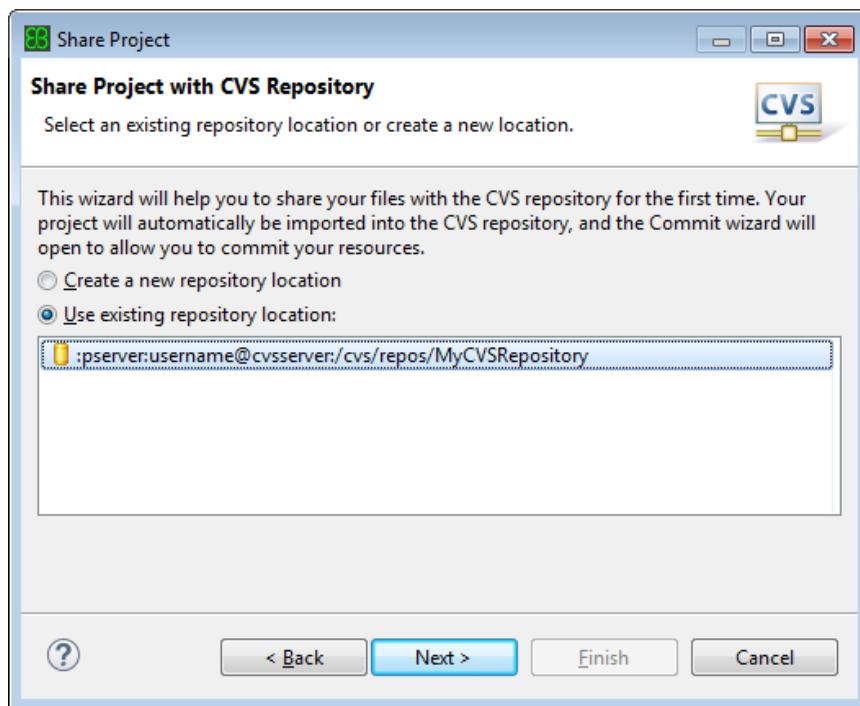


Figure 6.49. Selecting the repository location in which you want to share your project



- ▶ Select the repository you want to add your project to. You can either select an already registered repository location or register a new repository location. For information on how to register a new repository, see [Section 6.4.2.1, “Registering your CVS repository location in EB tresos Studio”](#).
- ▶ Click **Next**. The **Enter Folder Name** page opens up.

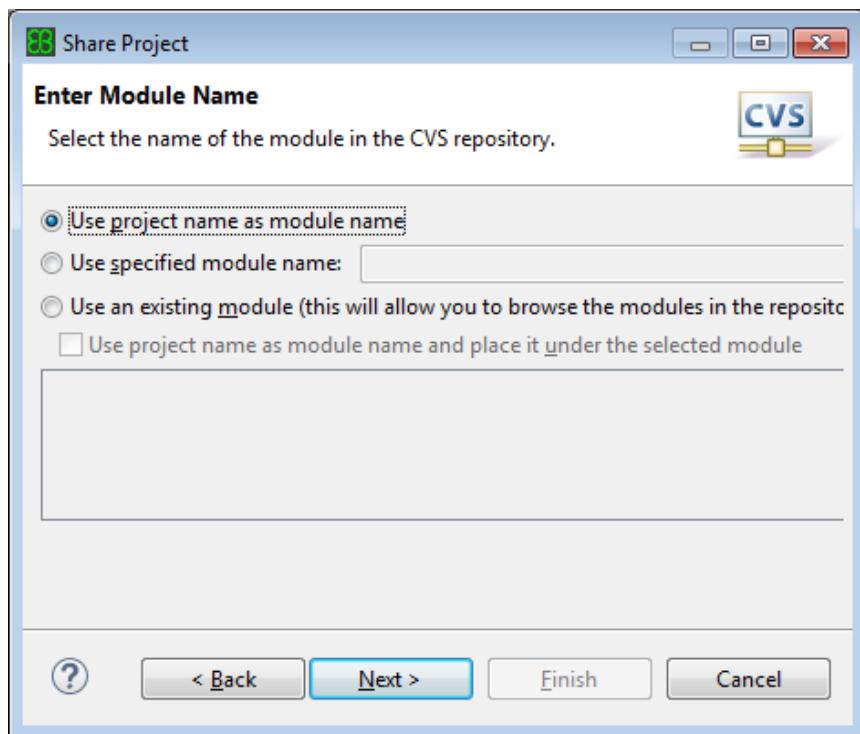


Figure 6.50. Entering a CVS module name for your project in the repository

- ▶ Enter the folder name in which you want to add your project to your CVS repository and click **Next**. The **Share Project Resources** page opens up.

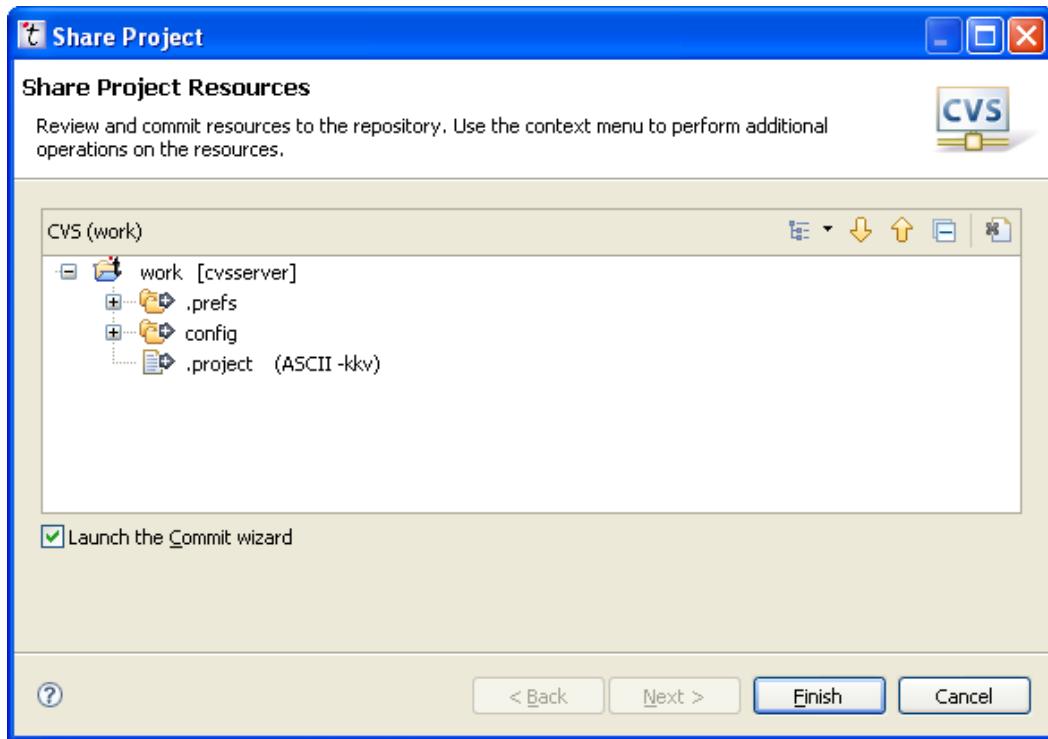


Figure 6.51. Reviewing and committing resources

- ▶ You can review the project content that will be shared and perform additional operations, such as adding files to the CVS ignore list, or excluding selected files that must not be added at this moment.
- ▶ Click **Finish**.

Your project is now under CVS control.

6.4.2.3. Checking out a shared project from your CVS repository into your EB tresos Studio workspace

In this section you will learn how to check out a CVS-managed project into your EB tresos Studio workspace. You can check out a project from a CVS repository into your EB tresos Studio workspace directly from the EB tresos Studio GUI.

To check out a project from a CVS repository into your EB tresos Studio workspace, you need to open the **Checkout Out As** dialog first. To open the **Checkout Out As** dialog:

- ▶ Right-click the shared project's folder in the **CVS Repositories** view, and select **Check Out As...** from the context menu. The **Check Out As** dialog opens up.

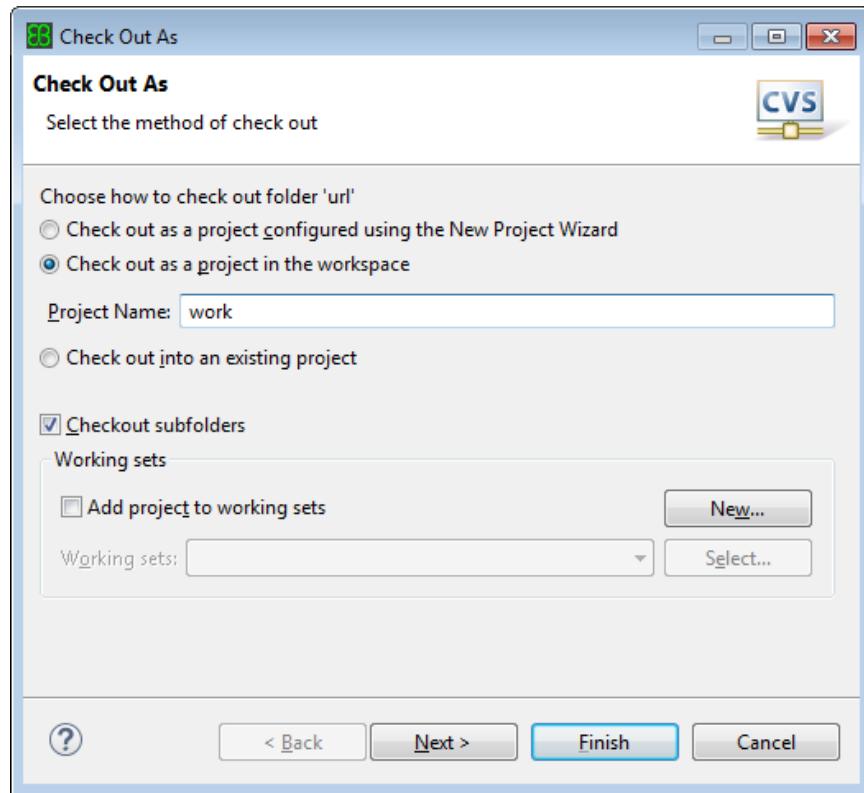


Figure 6.52. Checking out from CVS repository

- ▶ In the **Check Out As** dialog, you can specify how to check out the chosen CVS module:
 - ▶ If the selected CVS module already contains the project infrastructure file `.project`, select **Check out as project in the workspace**.
 - ▶ If the chosen module does not yet contain the project infrastructure file `.project`, select **Check out as project configured using the New Project Wizard** and click **Next**. The **New Project Wizard** opens up.

Follow the **New Project Wizard**. For instructions on creating a new project with the **New Project Wizard**, see [Section 6.3.1.2, “Step 2: Creating a new project”](#).

- ▶ To check out the selected CVS module into an already existing project, select **Check out into an existing project**.
 - ▶ Click **Next** to proceed to further wizard pages, where you can provide further options, such as where to checkout the files to, or which CVS tag or branch to use for the checkout.
- ▶ Click **Finish**.

The selected files are being checked out. As a result you see the checked out project in your **Project Explorer** view, displayed with an overlay icon:

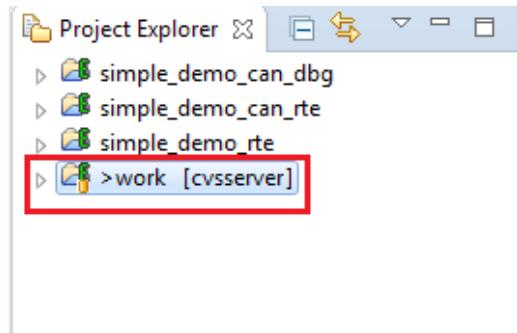


Figure 6.53. Checked out project in your **Project Explorer** view

For information on the meaning of the overlay icons, see [“Project Explorer overlay icons for CVS repository support”](#).

6.4.2.4. Updating, branching, and committing EB tresos Studio projects into your CVS repository

After you have registered and checked out your repository into EB tresos Studio, you may want to edit the files and commit your work to the repository.

Your files and folders are displayed with different overlay icons in the **Project Explorer** view. For information on the meaning of these overlay icons, see [“Project Explorer overlay icons for CVS repository support”](#).

To commit, update, branch your work to a repository or to choose another typical task for working with CVS repositories:

- ▶ Right-click the project in the **Project Explorer** view. A context menu opens up.
- ▶ In the context menu, select **Team**.
- ▶ The CVS context menu opens up.



Figure 6.54. The CVS context menu

In the CVS context menu you can select the tasks typical for working with CVS repositories, such as committing, updating, branching, tagging the repository etc.

WARNING**Reload projects after updating**

It is strongly recommended to only update files from CVS, when the configuration project is not loaded yet. After you have updated files in loaded configuration projects, the project needs to be reloaded. For further information on reloading the project, see [Section 6.3.6.4, “Reloading a project”](#).

If you want to diff files and file versions, see [Section 6.5, “Diffing files and file versions”](#).

6.4.3. Working with SVN repositories

This chapter is interesting for you if you want to connect your EB tresos Studio with your SVN repository and commit to shared EB tresos Studio projects directly from the EB tresos Studio GUI.

The state (e.g. locally modified, added) of your local files compared to the files in the SVN repository is typically indicated by overlay icons. The following overlay icons are displayed in the **Project Explorer** view:

Project Explorer overlay icons for SVN repository support



Indicates a local modification. Either a file is locally modified, or a folder contains a locally modified file.



Indicates a new file or folder within a version controlled folder that has not yet been added to version control.



Indicates that the file or folder was recently added to version control, but is not yet committed.



Indicates that the file or folder is under version control, without local modifications.

6.4.3.1. Registering your SVN repository location in EB tresos Studio

To access your SVN repositories from within EB tresos Studio, you need to register the repository location in EB tresos Studio first. To register the repository location in EB tresos Studio:

- ▶ In the **Window** menu, select **Show view**. A context menu opens up.
- ▶ In the context menu that opens up, select **Other....** The **Show View** dialog opens up.
- ▶ Expand **SVN** and select **SVN Repositories**. The **SVN Repositories** view opens up as a tab in the **Properties** view in the lower right corner of the EB tresos Studio workbench.



Figure 6.55. Registering a new repository location in the **SVN Repositories** view

- ▶ To register a new repository to be accessed via EB tresos Studio, click the button . The **Add SVN Repository** dialog opens up.

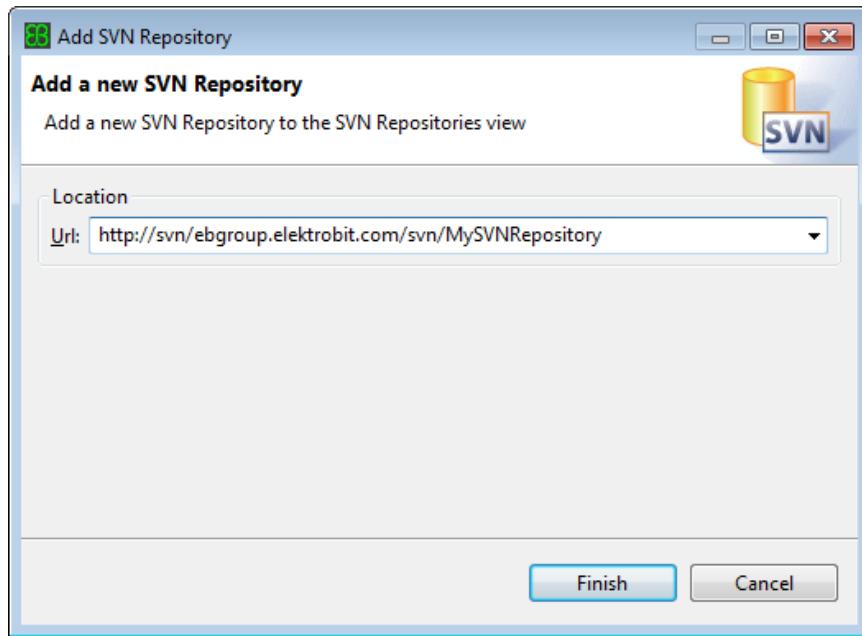
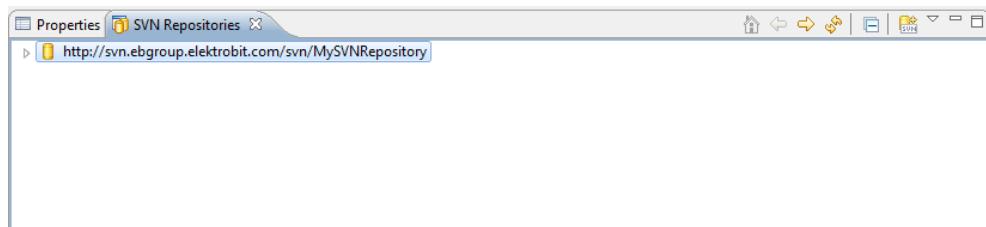


Figure 6.56. Adding a new repository location in EB tresos Studio

- ▶ In the **Location Url** text box, enter the link to your repository.
- ▶ Click **Finish**. The repository is displayed in your **SVN Repositories** view.

Figure 6.57. Registered repositories in the **SVN Repositories** view

6.4.3.2. Sharing an EB tresos Studio project with other team members

In this section you will learn how to share a project with other team members. This is equivalent to adding new files to an SVN repository and thus only applies to projects that have not yet been committed to the repository.

NOTE
You can only share whole projects, not single files


Adding single files to a repository is not possible from within EB tresos Studio. Thus if you want to share new files, you must share the project they reside in first.

To share a new project:



- Right-click this project in the **Project Explorer** view, select **Team** and then **Share Project...** from the context menus that open up:

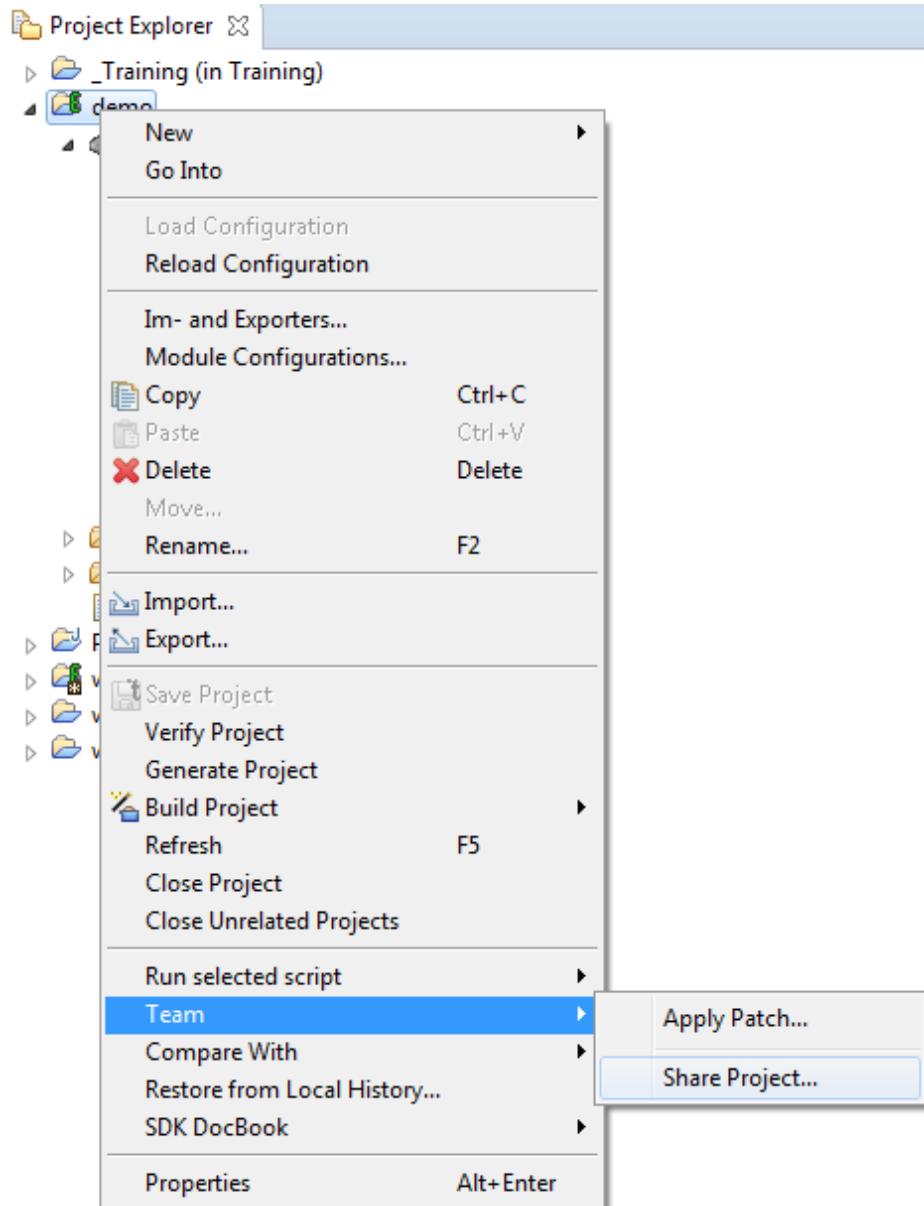


Figure 6.58. Selecting a project to share

- The **Share Project** dialog opens up.

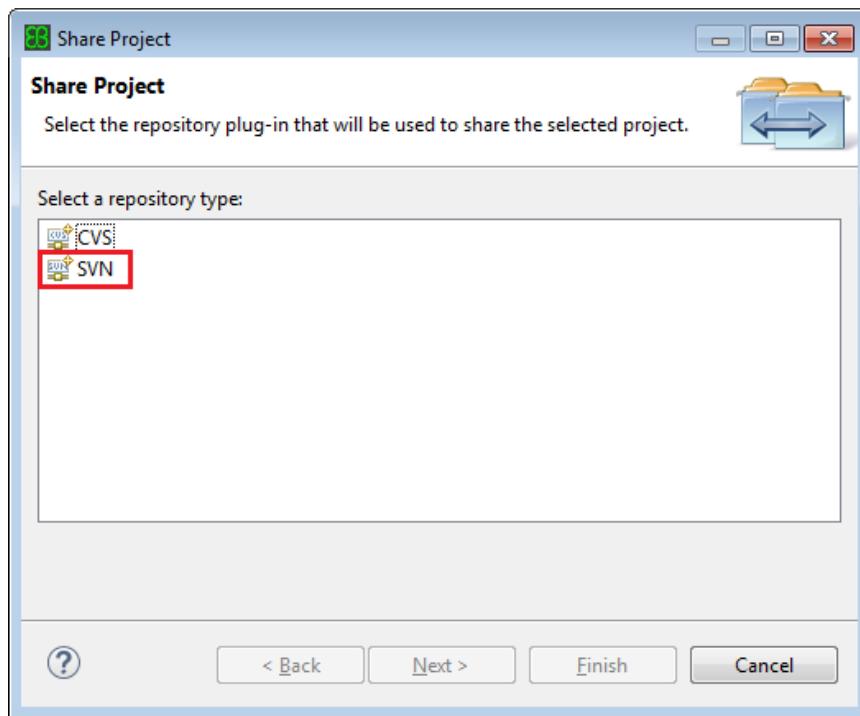


Figure 6.59. Selecting the repository type in which you want to share your project

- Select **SVN** and click **Next**.

The **Share Project with SVN Repository** page opens up

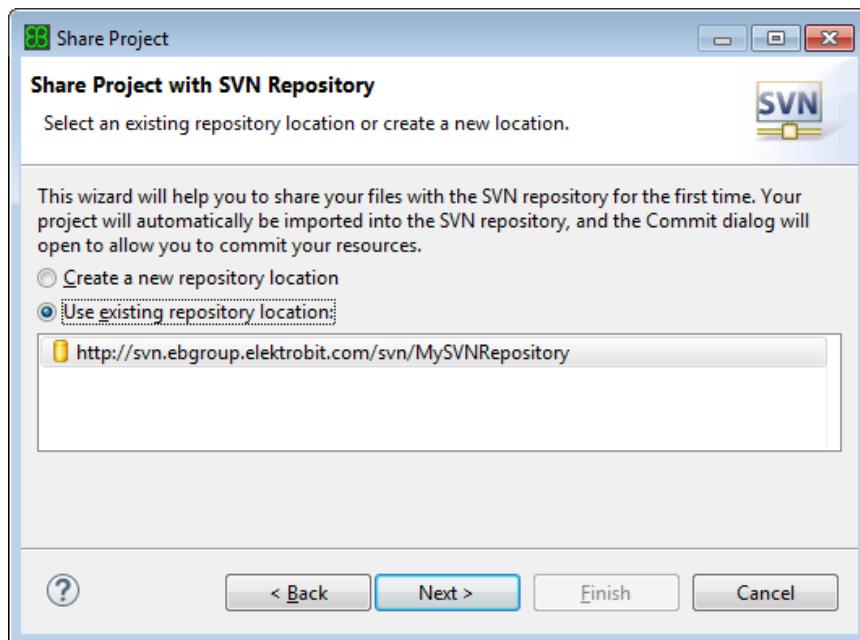


Figure 6.60. Selecting the repository location in which you want to share your project



- ▶ Select the repository you want to add your project to. You can either select an already registered repository location or register a new repository location.
- ▶ Click **Next**. The **Enter Folder Name** page opens up.

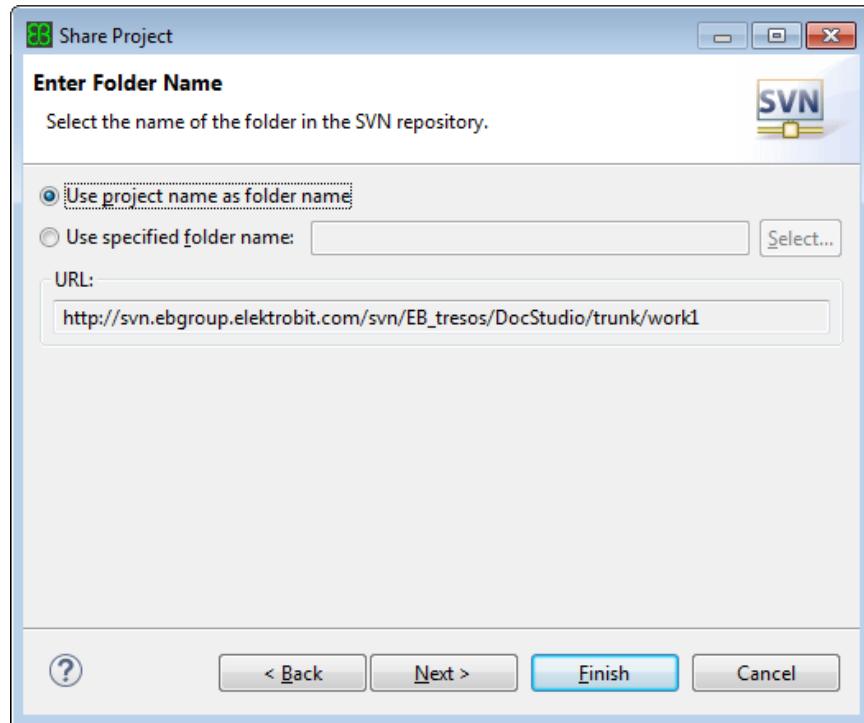


Figure 6.61. Entering a folder name for your project in the repository

- ▶ Enter the folder name in which you want to add your project to your SVN repository and click **Next**. The **Ready to Share Project** page opens up.

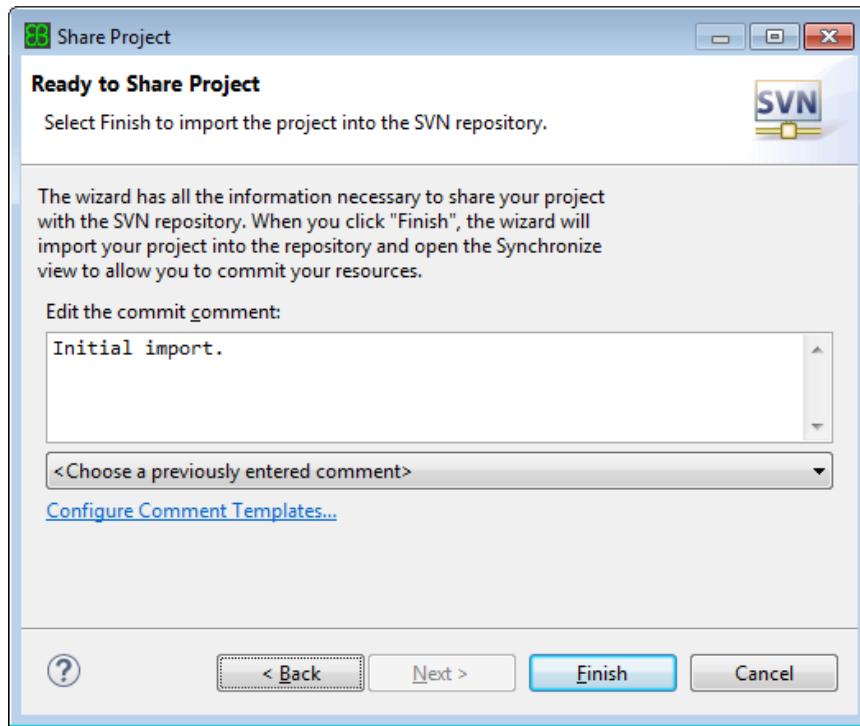


Figure 6.62. Entering a commit comment to your newly added project

- ▶ Enter a commit comment.
- ▶ To commit the added project to your repository, click **Finish**.

Your project is added and committed to your selected repository. After you have confirmed a confirmation dialog, the **Team Synchronizing** perspective opens up and displays the shared project. For information on the **Team Synchronizing** perspective, see [Section 6.4.4.2.1, “The Team Synchronizing perspective”](#).

6.4.3.3. Checking out a shared project from your SVN repository into your EB tresos Studio workspace

In this section you will learn how to check out an SVN-managed project into your EB tresos Studio workspace. You can check out a project from an SVN repository into your EB tresos Studio workspace directly from the EB tresos Studio GUI.

To check out an SVN-managed project into your EB tresos Studio workspace, you need to open the **Checkout from SVN** dialog first. To open the **Checkout from SVN** dialog:

- ▶ Right-click the registered repository in the **SVN Repositories** view, and select **Checkout....** The **Checkout from SVN** dialog opens up.

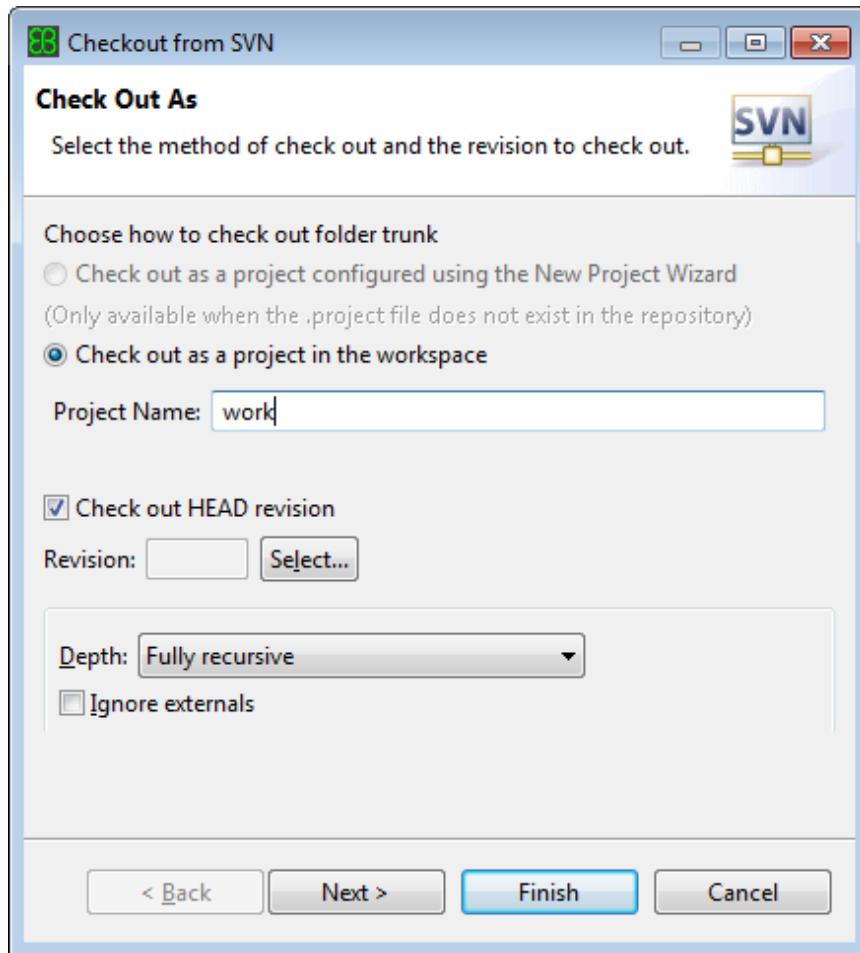


Figure 6.63. Checking out the SVN repository

- ▶ In the **Check Out As** page, you can specify how to check out the chosen SVN module:
 - ▶ If the selected SVN folder already contains the project infrastructure file `.project`, select **Check out as project in the workspace**.
 - ▶ If the chosen module does not yet contain the project infrastructure file `.project`, select **Check out as project configured using the New Project Wizard** and click **Next**. The **New Project Wizard** opens up.

Follow the **New Project Wizard**. For instructions on creating a new project with the **New Project Wizard**, see [Section 6.3.1.2, “Step 2: Creating a new project”](#).
 - ▶ If you do not want to check out the HEAD revision, clear the **Check out HEAD revision** option and enter the specific revision you want to check out.
 - ▶ If you do not want to use the default workspace location, click **Next** to proceed to another wizard page, where you can specify where to checkout the files to.
- ▶ Click **Finish**.



The selected files are being checked out to the indicated location. As a result you see the checked out project in your **Project Explorer** view, displayed with an overlay icon:

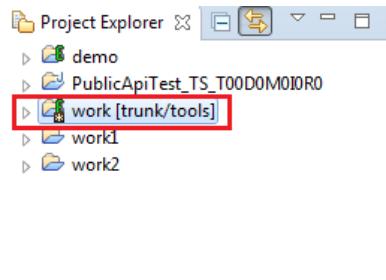


Figure 6.64. Checked out project in your **Project Explorer** view

For information on the meaning of the overlay icons, refer to "[Project Explorer overlay icons for SVN repository support](#)".

6.4.3.4. Updating, branching, and committing EB tresos Studio projects into your SVN repository

After you have registered and checked out your repository into EB tresos Studio, you may want to edit the files and commit your work to the repository.

Your files and folders are displayed with different overlay icons in the **Project Explorer** view. For information on the meaning of these overlay icons, see "[Project Explorer overlay icons for SVN repository support](#)".

To commit, update, branch your work to a repository or to choose another typical task for working with SVN repositories:

- ▶ Right-click the project in the **Project Explorer** view. A context menu opens up.
- ▶ In the context menu, select **Team**.

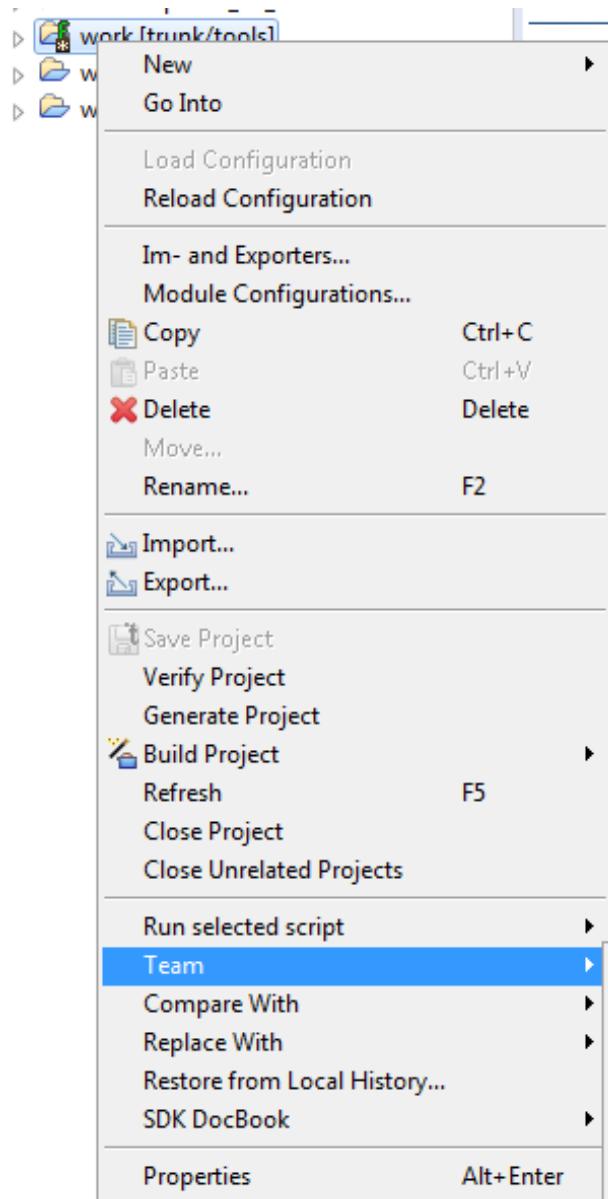


Figure 6.65. Opening the SVN context menu

- The SVN context menu opens up.

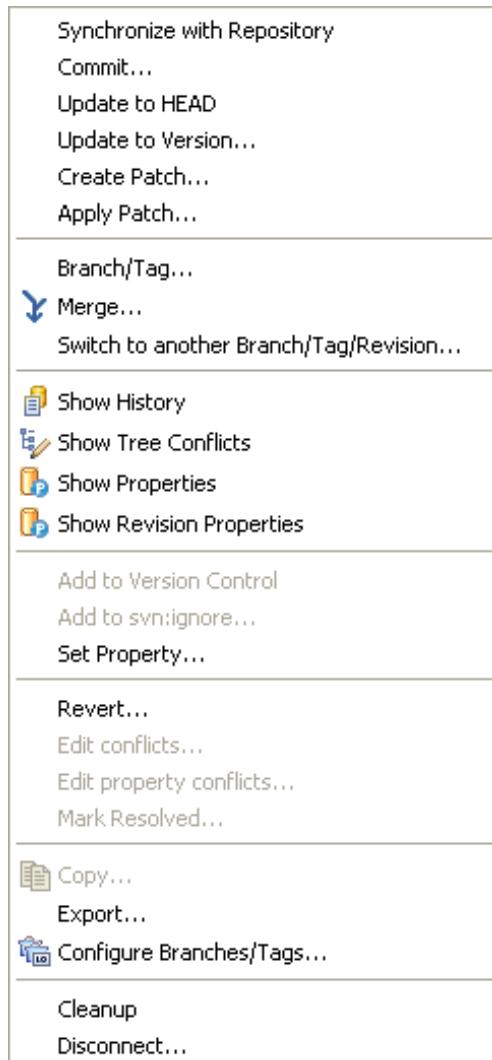


Figure 6.66. The SVN context menu

In the SVN context menu you can select the tasks typical for working with SVN repositories, such as committing, updating, branching, tagging the repository etc.

WARNING**Reload projects after updating**

It is strongly recommended to only update files from SVN, when the configuration project is not loaded yet. After you have updated files in loaded configuration projects, the project needs to be reloaded. For further information on reloading the project, see [Section 6.3.6.4, “Reloading a project”](#).

If you want to diff files and file versions, see [Section 6.5, “Differing files and file versions”](#).



6.4.4. Synchronizing your project with your repository

6.4.4.1. Overview

If you share your project with a team in a repository such as SVN or CVS, you might want to synchronize your work with the repository. Synchronizing means:

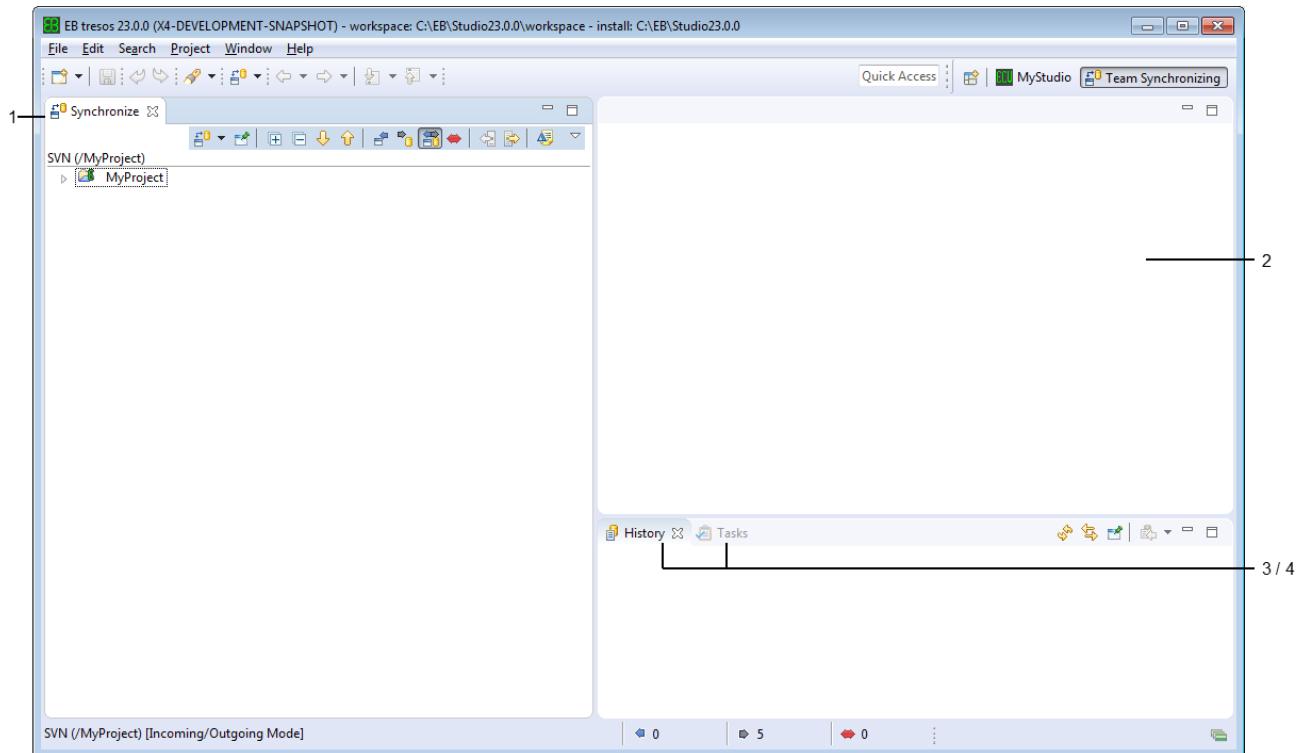
- ▶ viewing differences between your working copy and the repository,
- ▶ updating your project in EB tresos Studio to the latest repository version,
- ▶ committing your changes to the repository,
- ▶ committing and simultaneously updating your project to/with the repository.

6.4.4.2. Background information

To synchronize your project with repositories, EB tresos Studio provides a whole perspective in which differences between your local files and the repository are displayed. This makes synchronizing easier for you. Information on how to open the **Team Synchronizing** perspective is available in [Section 6.4.4.3, “Opening the Team Synchronizing perspective”](#). In this section you find basic information on the **Team Synchronizing** perspective and on its central view, the **Synchronize** view.

6.4.4.2.1. The Team Synchronizing perspective

The **Team Synchronizing** perspective consists of the following views:

Figure 6.67. The **Team Synchronizing** perspective

Number	Element	Description
1	Synchronize view	The Synchronize view displays differences between your working copy and the repository of a project. For a detailed description of the Synchronize view, see Section 6.4.4.2.2, "The Synchronize view" .
2	editor area	In the editor area, editors such as the Compare editor for text files or for XDM files open up. With these editors you can compare different versions or files with each other.
3	History view	In the History view, you can open the check-in history of a file, folder or project that is connected with a repository.
4	Tasks view	The Tasks view helps you to organize your work. You can add short task descriptions together with additional attributes such as a priority.

6.4.4.2.2. The Synchronize view

The **Synchronize** view is the central GUI element with which you may synchronize your files with a repository. It displays differences between selected workspace resources and the repository. The **Synchronize** view is



available on the left side of the **Team Synchronizing** perspective. For instructions on opening the **Team Synchronizing** perspective, see [Section 6.4.4.3, “Opening the Team Synchronizing perspective”](#).

The **Synchronize** view consists of the following two components:

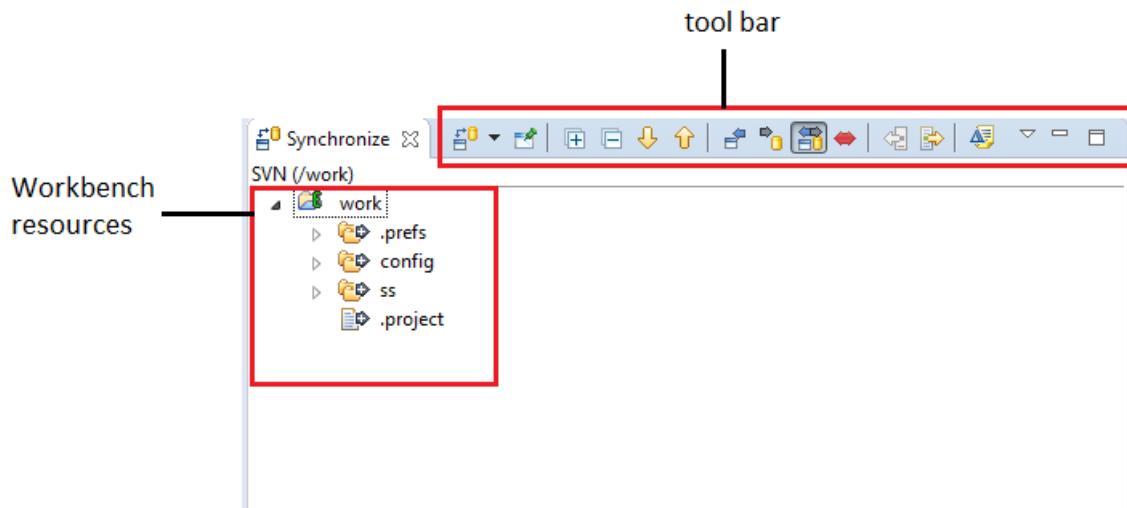


Figure 6.68. The **Synchronize** view

In the **Workbench resources** section of the **Synchronize** view, selected resources from your workbench are displayed. These workbench resources are displayed with overlay icons that indicate differences between the repository from which you have checked out these resources and your working copy. You find a list of these overlay icons with a description of their meanings at [Section 6.4.4.5, “Synchronizing your working copy with changes in the repository”](#).

The tool bar contains the following buttons:



Figure 6.69. The **Synchronize** view tool bar

Button	Name of the button	Description
	Synchronize	With this button you open a context menu in which you may choose which former set of resources you want to synchronize. From this menu, you can also open a wizard for creating a new set of resources to synchronize.
	Pin current synchronization	With this button you add a project to the context menu of the Synchronize button. A project that is pinned like this is automatically synchronized when the working copy changes.
	Expand all	With this button you expand all tree items displayed in the Synchronize view.



Button	Name of the button	Description
	Next difference	With this button you jump to the next difference between the repository and your working copy in the resources, that are currently displayed in the Synchronize view.
	Previous difference	With this button you jump to the previous difference between the repository and your working copy in the resources, that are currently displayed in the Synchronize view.
	Collapse all	With this button you collapse all tree items displayed in the Synchronize view.
	Incoming mode	With this button you filter your synchronization for incoming changes from the repository. Click this button if you want to update your local version to the head revision of the repository.
	Outgoing mode	With this button you filter your synchronization for outgoing changes from your local version to the repository. Click this button if you want to commit your local version to the repository.
	Incoming/outgoing mode	With this button you filter your synchronization for outgoing and incoming changes from your local version to the repository. Click this button if you want to commit your local version to the repository and simultaneously update your local version to the head revision of the repository.
	Conflicts mode	With this button you filter your synchronization for conflicts between your local copy and the repository. Click this button if you want to find out which files you will have to merge before committing them to the repository.
	Update all incoming changes	With this button you update all files visible in the Synchronize view with non-conflicting incoming changes with the newer versions in your repository.
	Commit all outgoing changes	With this button you commit all non-conflicting local changes visible in the view to your repository.
	Show change sets (SVN repositories only)	With this button you can enable the support for change sets. When enabled, you can organize the content of the Synchronize view into change sets for clarity.

6.4.4.3. Opening the Team Synchronizing perspective

To synchronize workbench resources with a repository, open the **Synchronize** perspective:

- ▶ Either open the **Team Synchronizing** perspective via the **Project Explorer** view:
 - ▶ Select the resources you want to synchronize with the repository in the **Project Explorer** view.



- ▶ Right-click the selection. A context menu opens up.
- ▶ Select **Team**. The **Team Collaboration** context menu opens up.
- ▶ In the **Team Collaboration** context menu, select **Synchronize with Repository**.

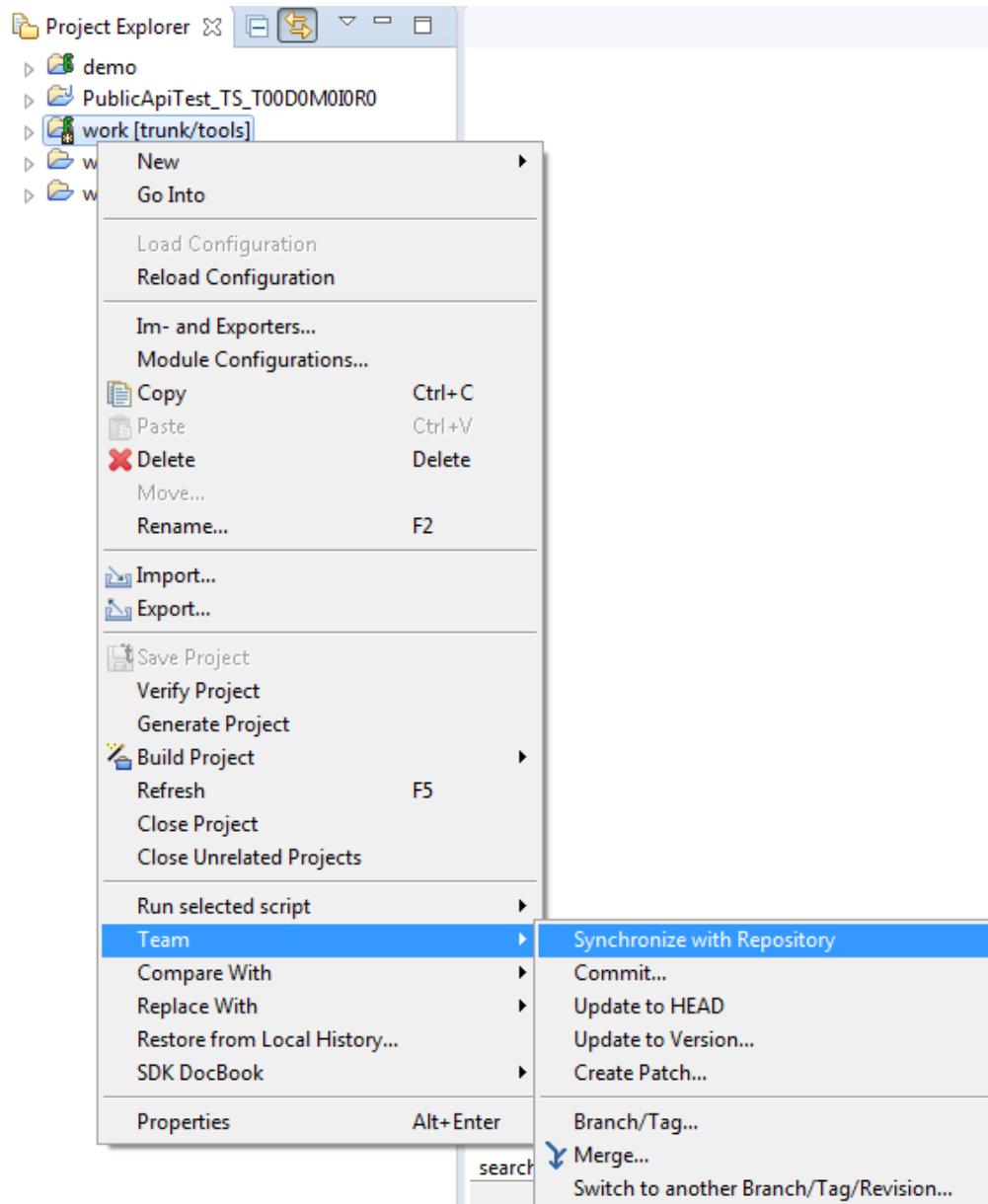


Figure 6.70. Opening the **Team Synchronizing** perspective

- ▶ Or select the **Team Synchronizing** perspective from the perspective bar:
 - ▶ In the perspective bar, click the **Open Perspective** button . The Open Perspective Dialog opens.

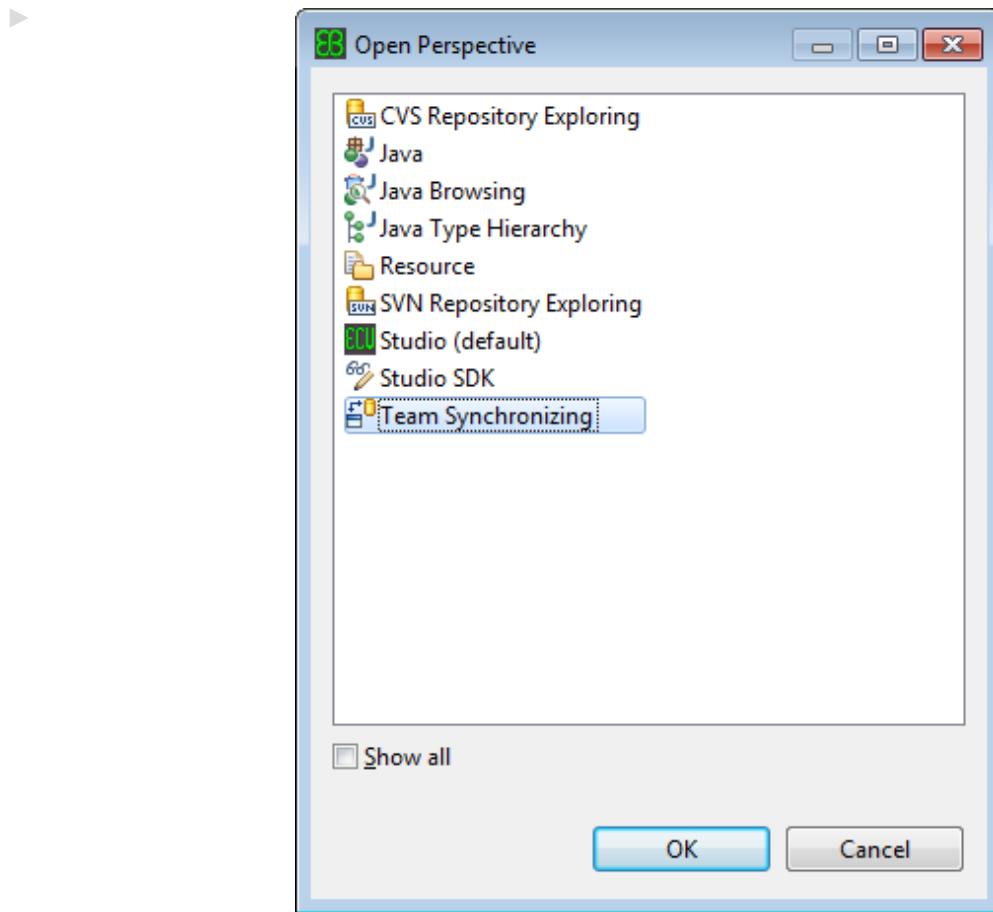


Figure 6.71. Opening the **Team Synchronizing** perspective via the perspective bar

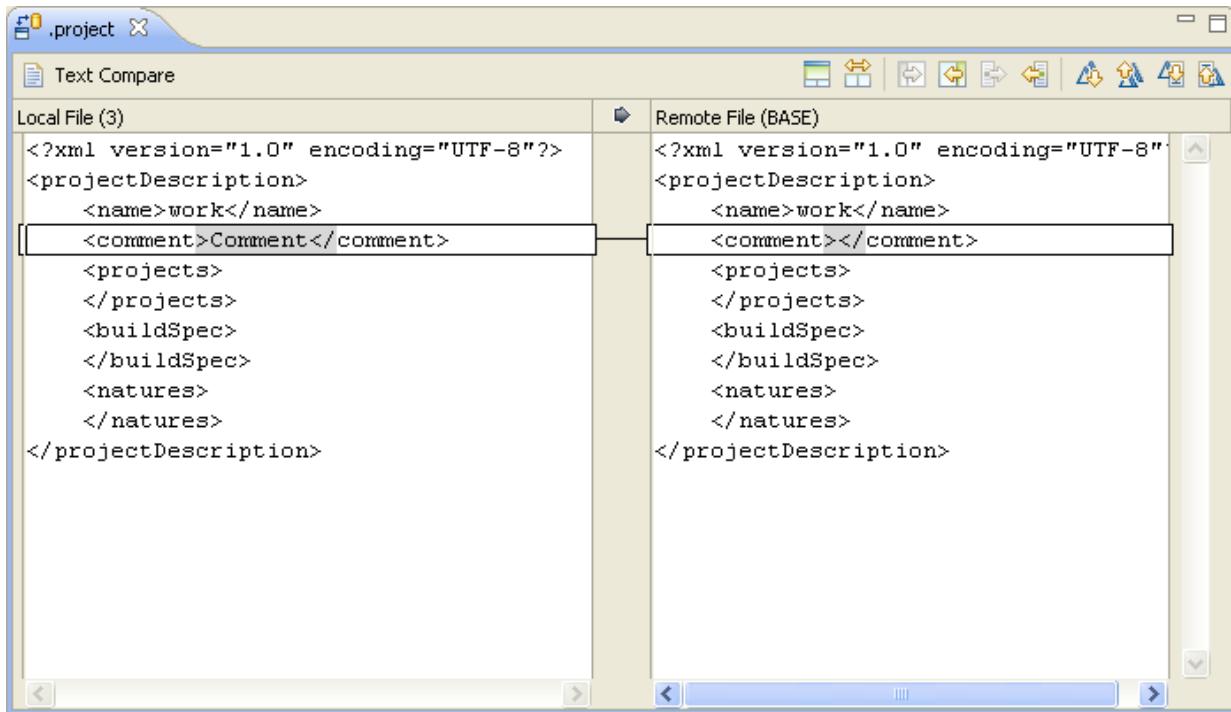
- ▶ Select **Team Synchronizing** and click **OK**.

The **Team Synchronizing** perspective opens up. For a detailed description of the **Team Synchronizing** perspective, see [Section 6.4.4.3, “Opening the Team Synchronizing perspective”](#).

6.4.4.4. Viewing differences between your working copy and the base file in the repository

In the **Team Synchronizing** perspective you can view the differences between your working copy and the base file. The base file is the file you have checked out to edit as working copy. To find out how your working copy differs from the base file:

- ▶ In the **Synchronize** view, expand your project.
- ▶ Double-click the file you want to check for differences. A compare editor opens up in the editor area.

Figure 6.72. Differencing with the **Text Compare** editor

- ▶ Which compare editor opens up depends on the file format you want to compare. For detailed information on the compare editors provided by EB tresos Studio, see [Section 6.5, “Differencing files and file versions”](#).

6.4.4.5. Synchronizing your working copy with changes in the repository

The **Synchronize** view indicates differences between the repository and your working copy. These differences are displayed with the following overlay icons:



Outgoing change which indicates that the file is locally modified.



Outgoing change which indicates that the file has been added locally.



Outgoing change which indicates that the file has been removed locally.



Incoming change which indicates that there is a newer version available in the repository.



Incoming change which indicates that the file is new in the repository.



Incoming change which indicates that the file has been removed from the repository.



Conflicting change which indicates that the file has local modifications that conflict with changes that were checked in to the repository by another team member while you were editing the file.



Conflicting change which indicates that the file has been added locally, and conflicts with an equally named file, that has been added and committed to the repository by another team member.

There are three different ways of synchronizing your project with a repository:

- ▶ Synchronizing in *incoming mode*, i.e. updating your working copy to the head revision of the repository.
- ▶ Synchronizing in *outgoing mode*, i.e. committing your local changes to the repository.
- ▶ Synchronizing in *incoming/outgoing mode*, i.e. committing and updating your project simultaneously to the repository.

To synchronize your project with the repository:

1. In the tool bar of the **Synchronize** view, select the mode in which you want to synchronize your project with the repository:
 - ▶ To synchronize incoming changes (i.e. to update your project), click the **Incoming Mode** button . In the **Synchronize** view, new file versions are displayed that you have not yet updated from the repository.
 - ▶ To synchronize outgoing changes (i.e. to commit your project to the repository), click the **Outgoing Mode** button . In the **Synchronize** view, local changes are displayed that you have not yet committed to the repository.
 - ▶ To synchronize incoming and outgoing changes simultaneously (i.e. to commit your project to the repository and simultaneously update your project), click the **Incoming/Outgoing Mode** button . In the **Synchronize** view, local changes as well as new file versions in the repository are displayed.
2. In the **Synchronize** view, right-click the file, folder or project you want to synchronize. A context menu opens up.

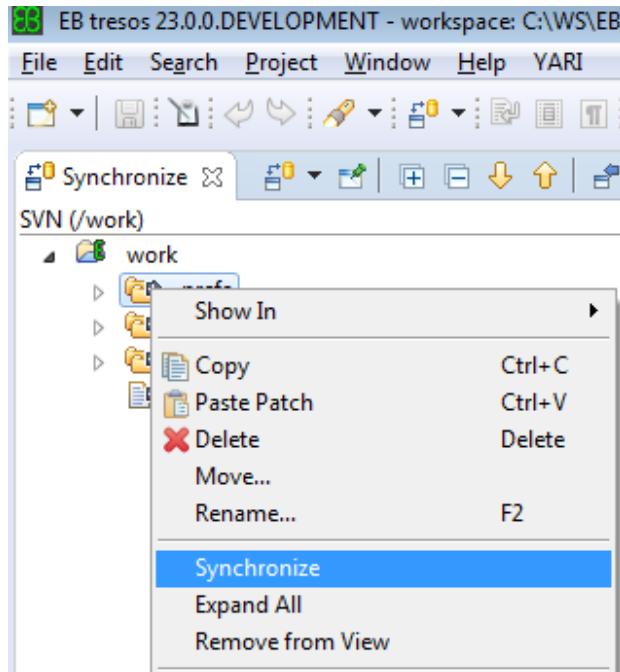


Figure 6.73. Synchronizing incoming changes

3. Select **Synchronize**.

Your project is being synchronized with the repository.

6.5. Diffing files and file versions

6.5.1. Overview

EB tresos Studio provides the possibility to compare files or file versions with each other. This is helpful if you work in a team, but also if you want to compare local files with each other. EB tresos Studio provides two different compare editors (differs) for comparing files, depending on which kinds of files you want to compare:

- ▶ The Eclipse standard **Compare** editor for text files. For detailed information on this compare editor, see the [Eclipse Workbench User Guide](#), which is also included in the EB tresos Studio onscreen help.
- ▶ The **Compare** editor for XDM files, described in [Section 6.5.2, “Diffing XDM files”](#).

6.5.2. Diffing XDM files



6.5.2.1. Overview

In this chapter you will learn how to compare XDM files using the **Compare** editor for XDM files. While comparing, you can also merge differences into one of the compared XDM files.

6.5.2.2. Background information

In EB tresos Studio many parameters are stored in XDM files within the project, e.g. the parameters of module configurations, or project-specific settings such as the AUTOSAR release version or the target and derivative settings.

Especially (but not limited to) when working in teams on module configurations, the need arises to view and probably merge differences between XDM files, either from different configuration projects or when comparing one file version to another using some version control system, e.g. CVS or SVN.

To compare XDM files, EB tresos Studio provides the **Compare** editor for XDM files.

6.5.2.3. Opening the compare editor

To compare two XDM files with each other:

- ▶ Select both XDM files in the **Project Explorer** view.
- ▶ Right-click one of the two files. A context menu opens up.
- ▶ Select **Compare With**. Another context menu opens up.
- ▶ In this context menu, select **Each other**.

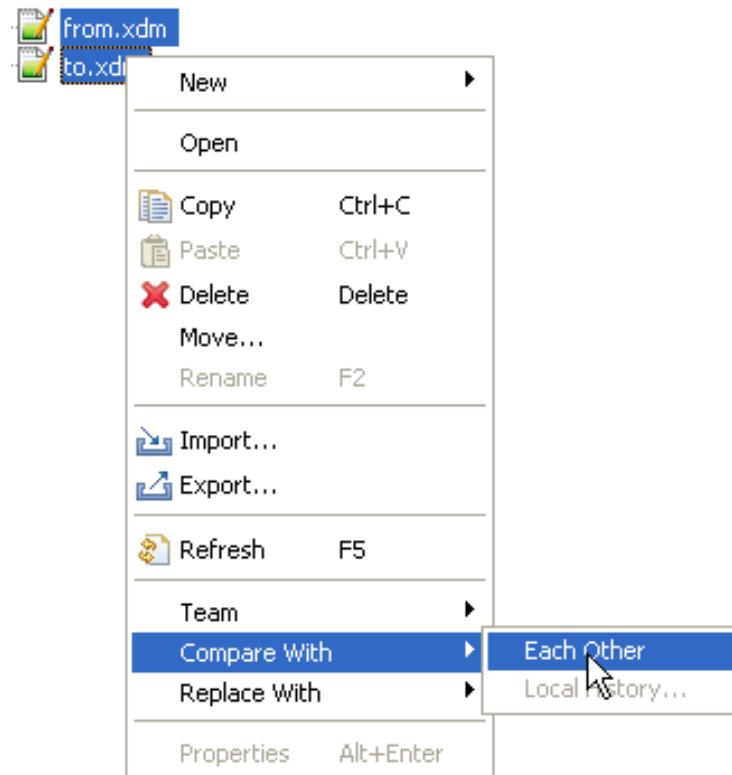


Figure 6.74. Directly comparing two XDM files

NOTE**Further ways to open the Compare editor for XDM files**

Apart from opening the compare editor from the **Project Explorer** view, there are further ways of opening this compare editor:

- ▶ Comparing different revisions from within the **History** view. For further information on the **History** view, see the [Eclipse Workbench User Guide](#). The Eclipse Workbench User Guide is also included in the EB tresos Studio onscreen help, which opens up when you press **F1**.
 - ▶ Using the team **Synchronize** view in the **Team Synchronizing** perspective. For further information on the **Team Synchronizing** perspective, see [Section 6.4.4.2.1, "The Team Synchronizing perspective"](#).
-
- ▶ A progress indicator dialog opens up.
 - ▶ If both files are editable, another dialog opens up in which you may select into which file you want to merge the differences.

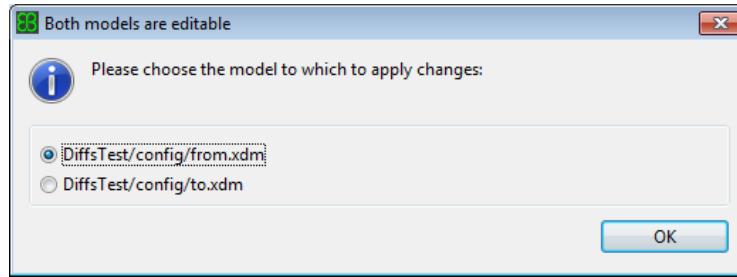


Figure 6.75. Selecting the XDM file to merge into

NOTE**Read-only mode**

If none of the files are editable, the compare editor opens up in a read-only mode. You cannot merge differences then.

-
- ▶ The **Compare** editor for XDM files opens up in the editor area.

6.5.2.3.1. The Compare editor for XDM files

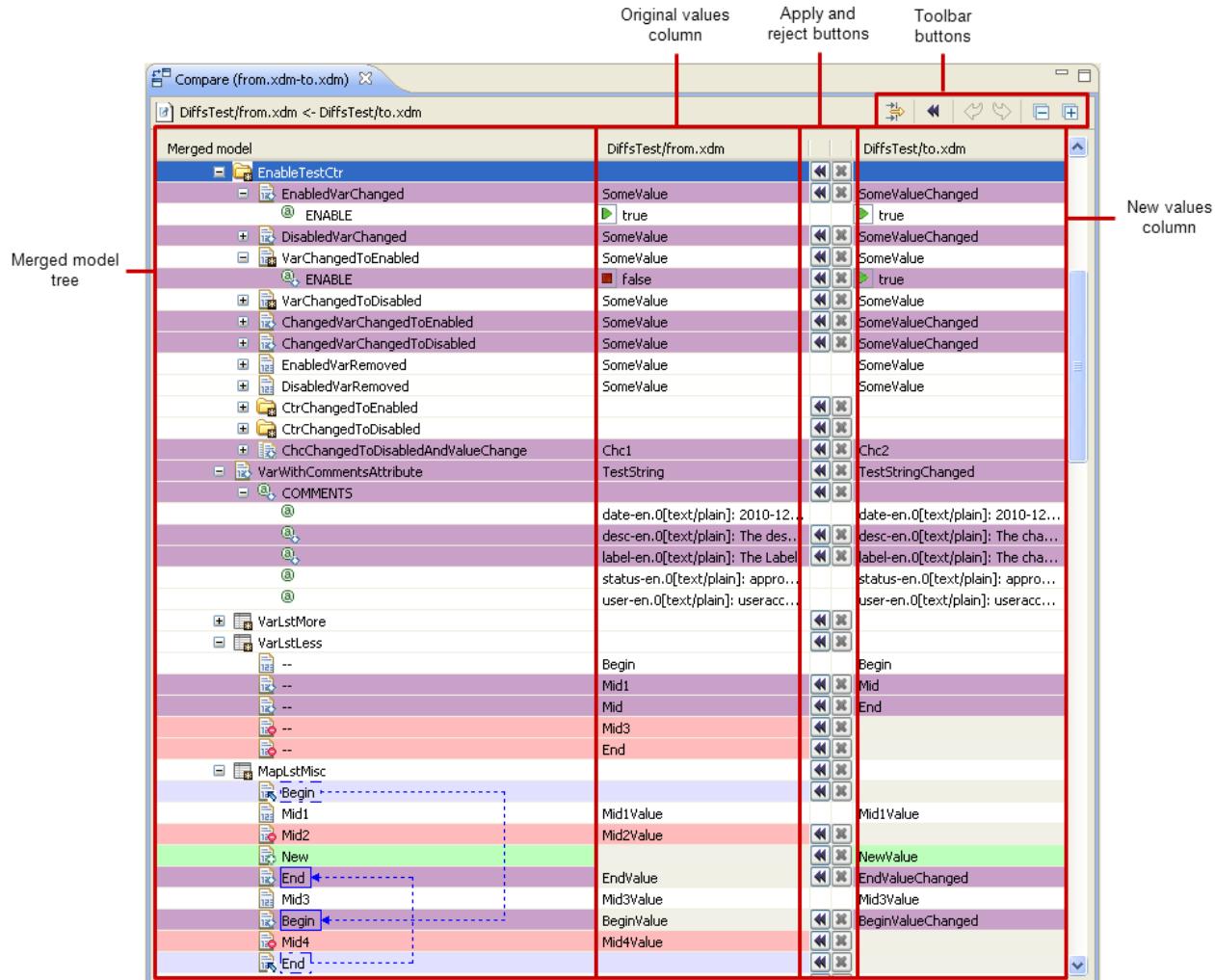


Figure 6.76. **Compare** editor for XDM files

Name	Explanation
Merged model tree	The left column called Merged Model shows the structure of the original XDM file. The original XDM file is taken as a reference schema. Differences are only made visible if they fit into the schema of the original file. When a named parameter has been moved into another location within a list, the move is displayed by two entries with a dotted-line arrow in between.
Original values column	The column next to the merged model tree shows the values of the original model, to which the new model is compared to. Optional XDM nodes show their activation status with additional icons in front of the value:



Name	Explanation
	<ul style="list-style-type: none"> ▶ meaning enabled ▶ meaning disabled <p>Additionally, the value of disabled nodes are grayed out.</p>
Apply and reject buttons	<p>The next two columns of the table contain the buttons for recursively applying or rejecting differences. In this context applying means to take over the changes from right to left. Rejecting means that the changes are not shown as a difference in this session any more. Mind that those two columns are only displayed if at least one of the two compared files is editable.</p>
New values column	<p>The column on the right shows the values of the new XDM file, which is being compared to the original XDM file.</p> <p>Optional XDM nodes are displayed in the same manner as in the original values column.</p> <p>You can select the font used in this column in the preferences settings as described in Section 6.2.2.12, “Setting colors and fonts”.</p>
Tool bar buttons	<p>The tool bar of the Compare editor for XDM files provides further action buttons for filtering, applying all remaining differences, undo and redo support, and collapsing or expanding the tree.</p>

Different background colors in the table represent the different types of differences. You can adjust those colors in the preferences settings as described in [Section 6.2.2.12, “Setting colors and fonts”](#).

The tool bar of the workbench contains two items for navigating to the next (and previous () difference. Those are bound to the keys **Ctrl + .** and **Ctrl + ,** by default.

A context menu is available for the items of the table, providing possible actions, like applying or rejecting all changes recursively or specific changes of the selected item, e.g.:



Figure 6.77. Context menu

When hovering over a changed item in the table, a tooltip appears, e.g.:

		4thStringValue	4thStringValue
Choice4	String		
EnableTestCtr	EnabledVarChange	Changed value: SomeValue --> SomeValueChanged	
	ENABLE	true	true
DisabledVarChanged		SomeValue	SomeValueChanged
VarChangedToEnabled	ENABLE	SomeValue	SomeValue
		false	true

Figure 6.78. Change tooltip

The status bar of the workbench displays two counters, one for the total number of differences, and another one for the number of unprocessed differences (i.e. applied or rejected).



Figure 6.79. Counters in the workbench status bar

6.5.2.3.2. Tree table icons and overlay icons

The icons used in the **Merged Model** tree column indicate the type of parameter. They are the same as the type icons used in the Generic Configuration Editor for parameters (see [Section 5.4.1.1, “Sections and parameters”](#)).

Differences in attributes are displayed with the icon in front of the changed tree item's child elements.

Overlay icons that are displayed over the type icon indicate the type of difference. The following list explains the meaning of all overlay icons, in ascending precedence, as only one overlay is displayed even if more changes exist for the item:



- ▶ Indicates a difference within the child elements of a tree item.
- ▶ Indicates a moved parameter value (the moved *to* position).
- ▶ Indicates a moved parameter value (the moved *from* position).
- ▶ Indicates an added parameter or attribute.
- ▶ Indicates a removed parameter or attribute.
- ▶ Indicates a changed value of a parameter or attribute.

6.5.2.4. Understanding the diffing results

Differences that are not yet processed (i.e. applied or rejected) between the compared datamodels are visualized by using overlay icons, different background colors, different values, etc. The following table describes the types of differences in detail:

Diff type	Overlay icon	Default background color	Explanation
Changed node value			The Original values column and the New values column display the different values.
Removed node			The Original values column displays the value of the removed node.
Added node			The New values column displays the value of the added node.
Moved node (from position)			The tree item displays the original position from where the tree item has been moved. This tree item neither displays any value nor any children nor any other difference. The tree item is surrounded by a frame (dotted line).
Moved node (to position)			The tree item displays the new position, to where the node has been moved. If the node also has a different value, it takes precedence for the overlay icon and background color. The tree item is surrounded by a frame (solid line). A blue arrow points from the original position of the node.



6.5.2.5. Navigating from difference to difference

In the **Compare** editor you can navigate from one difference to the next. This is useful if you want to compare large files that have few differences.

- ▶ To select the next item in the tree with an unprocessed difference, click the **Next Difference** button  in the workbench tool bar.
- ▶ To select the previous item in the tree with an unprocessed difference, click the **Previous Difference** button  in the workbench tool bar.

6.5.2.5.1. Filtering

In the **Compare** editor for XDM files you can filter the diffing results for processed and unprocessed differences. A difference is *processed* if you have applied or rejected the difference. A difference is *unprocessed* if you have not applied or rejected this difference.

To filter all items that are already processed or do not have any differences:

- ▶ Click the **Only show unprocessed changes** toggle button once  in the tool bar of the compare editor.
- ▶ The button shows the following toggled state:  . All tree nodes that are already processed or do not have any differences become invisible.
- ▶ To disable the filtering, click the button once again.

6.5.2.5.2. Expanding and collapsing the tree

You can expand or collapse arbitrary parts of the **Merged model** tree:

- ▶ To expand the whole tree, click the **Expand all** button  in the tool bar of the compare editor.
- ▶ To collapse the whole tree, click the **Collapse all** button  in the tool bar of the compare editor.
- ▶ To expand one or more subtrees:
 - ▶ Select the tree items which are the parent elements of the subtrees you want to expand.
 - ▶ Right-click one of the selected tree items and select **Expand** from the context menu.
 - ▶ The selected tree items are expanded deeply, i.e. all child elements beneath the selected tree items become visible and expanded themselves.
- ▶ To collapse one or more subtrees:



- ▶ Select the tree items which are the parent elements of the subtrees you want to collapse.
- ▶ Right-click one of the selected tree items and select **Collapse** from the context menu.
- ▶ The selected tree items are collapsed deeply, i.e. all child elements beneath the selected tree items become invisible and collapsed themselves.
- ▶ To expand one level of a single tree item:
 - ▶ Click the **Expand** button  next to the collapsed tree item you want to expand.
 - ▶ The tree item is expanded, i.e. its child elements become visible, which themselves may currently be collapsed or expanded.
- ▶ To collapse one level of a single tree item:
 - ▶ Click the **Collapse** button  next to the expanded tree item you want to collapse.
 - ▶ The tree item is collapsed, i.e. its child elements become invisible. The child element's expansion state remains unchanged, though.

NOTE**Applying and rejecting actions do not consider the expansion state of an item**

The **Apply** and **Reject** actions described in [Section 6.5.2.6.1.2, “Applying subtrees”](#) and [Section 6.5.2.6.2.2, “Rejecting subtrees”](#) do not consider the expansion state of any selected items, but always work on the complete subtrees.

6.5.2.5.3. Keyboard shortcuts

EB tresos Studio provides the following keyboard shortcuts when working with the XDM differ:

Keyboard shortcut	Task
Ctrl + .	Select the next difference.
Ctrl + ,	Select the previous difference.

6.5.2.6. Merging differences

In the **Compare** editor for XDM files you can directly merge the differences into one of the two compared files. Into which one of these two files these differences are merged you have to define when opening the compare editor. For instructions on how to do that see [Section 6.5.2.3, “Opening the compare editor”](#).

To merge a difference you have to either apply or reject this difference. After a difference has been applied or rejected, it is considered to be *processed*. Thus, an *unprocessed* difference is a difference that hasn't been applied or rejected yet.

**NOTE****Save your file after merging to persist the change**

After you have applied any changes, the compare editor displays the dirty indicator (*) in the title tab *Comp. To make the changes persistent, you need to save the file.

WARNING**Reload projects after changing any XDM files**

It is strongly recommended to only merge XDM files, when the configuration project is not loaded yet. After you have merged and saved XDM files in loaded configuration projects, the project needs to be reloaded. For further information on reloading the project, see [Section 6.3.6.4, "Reloading a project".](#)

Otherwise all changes done by merging the XDM files will be lost, when you save the configuration project the next time.

6.5.2.6.1. Applying differences

You can merge differences from the right file to the left file by applying changes. You can either apply one difference at a time or you can apply all differences for one tree item with their child elements simultaneously.

6.5.2.6.1.1. Applying single differences

To apply one selected difference:

- ▶ Right-click the tree item with the difference you want to apply.

A context menu opens up:



Figure 6.80. Applying one single difference



- ▶ Depending on the type of difference you want to apply, select the appropriate entry marked by the icon (i.e. select either **Take over value**, **Remove parameter**, **Add parameter**, **Change parameter activation** or **Move parameter**).
- ▶ The change is applied.

6.5.2.6.1.2. Applying subtrees

You can apply all changes of a tree item and all of its child elements in the tree simultaneously.

There are two alternatives to apply all changes of selected tree items recursively:

- ▶ Alternative 1:
 - ▶ Click the **Apply** button  in the table for the tree items you want to apply.
- ▶ Alternative 2:
 - ▶ Select the tree items in the table for which you want to apply the changes.
 - ▶ Right-click one of these selected items and select **Apply** from the context menu.

The differences are taken over from the **Original values** column to the **New values** column and the processed tree items are collapsed.

6.5.2.6.1.3. Applying all remaining differences

You can also apply all unprocessed changes at once.

To apply all remaining differences:

- ▶ In the tool bar of the editor, click the button **Apply all unprocessed changes** .
- ▶ All remaining changes are applied to the left file.
- ▶ The counter for the unprocessed changes is set to zero.

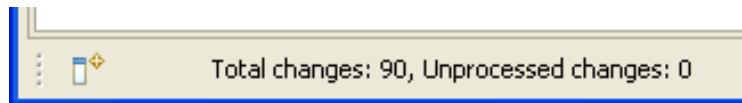


Figure 6.81. No more unprocessed changes

6.5.2.6.2. Rejecting changes

You can also reject changes. Rejected changes are not highlighted any longer in the compare editor and count as processed, both within the current compare session.



6.5.2.6.2.1. Rejecting single differences

To reject a selected difference:

- ▶ Right-click the tree item with the difference you want to reject.
- ▶ A context menu opens up:



Figure 6.82. Rejecting differences

- ▶ Depending on the type of difference you want to reject, select the appropriate entry marked by the icon (i.e. select either **Keep value**, **Keep parameter**, **Ignore parameter**, **Keep parameter activation** or **Do not move parameter**).
- ▶ The change is rejected.

6.5.2.6.2.2. Rejecting subtrees

You can reject all differences of a tree item and all of its child elements in the tree simultaneously.

There are two alternatives to reject all changes of selected tree items recursively:

- ▶ Alternative 1:
 - ▶ In the **Apply and reject buttons** column, click the button **Reject** button next to the tree items you want to reject.
- ▶ Alternative 2:
 - ▶ Select the tree items you want to reject.
 - ▶ Right-click one of the selected items and select **Reject** from the context menu.

The processed nodes are rejected and become collapsed.



6.5.2.6.3. Undoing/Redoing operations

You can successively undo or redo the latest apply or reject operations.

- ▶ To undo the latest apply or reject operation, click the **Undo** button  in the tool bar of the compare editor.
- ▶ To redo the latest undo operation, click the **Redo** button  in the tool bar of the compare editor.

NOTE**Only operations on unsaved files can be undone**

Operations can only be undone as long as you do not save the changed file.



6.5.2.7. Limitations

The **Compare** editor for XDM files has the following limitations you need to be aware of.

6.5.2.7.1. Ignored differences

Not all possible datamodel differences are being supported by the compare editor.

6.5.2.7.2. Schema support

The algorithm used for calculating the differences has no explicit schema support, but takes the structure of the original file (the left side) as a reference. Any nodes within the compared file, that do not fit into that structure, are silently ignored.

This may even go as far as not showing any differences at all, when comparing files that have a completely incompatible structure. E.g. when you compare a module config XDM file with the .prefs/preferences.xdm file of a configuration project, no differences are displayed.

6.6. Editing module configurations

This chapter deals with the management of module configurations of an ECU configuration project. In this chapter you learn how to add or remove module configurations, where to view detailed information about module configurations, how to adjust certain module configuration details, how to change the default settings for the pre- and recommended configurations, how to upgrade the module configurations.

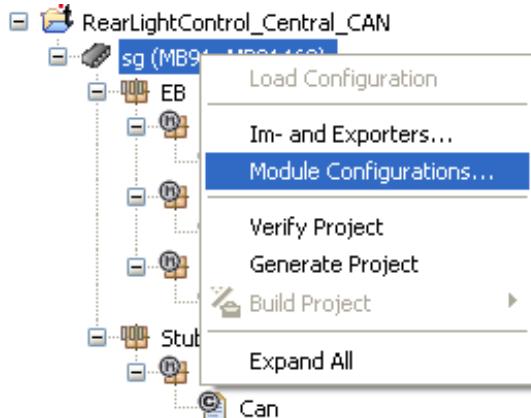
**TIP****When to edit module configurations**

You can edit the set of the module configurations of an ECU configuration project in different situations:

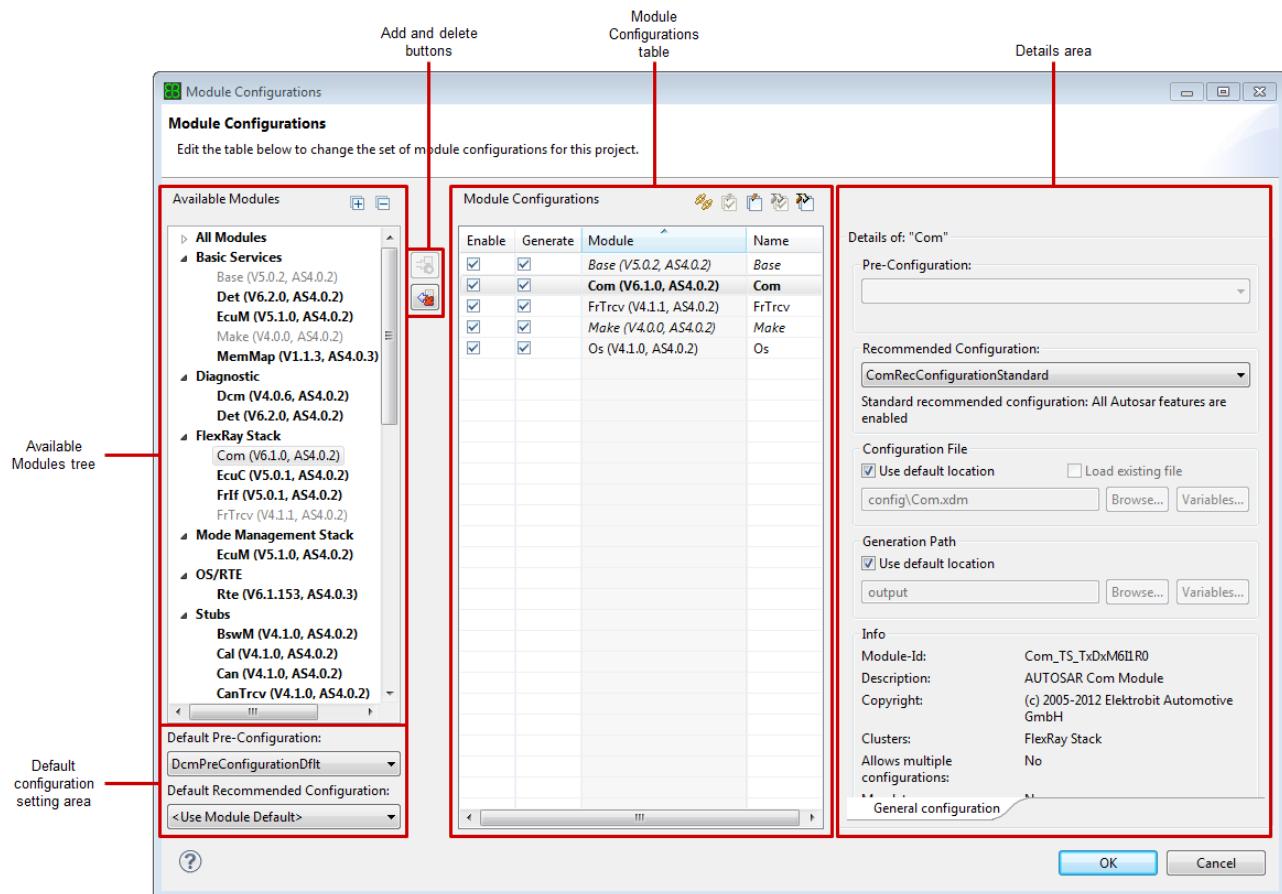
- ▶ During the creation of a new ECU configuration project using the **New Project Wizard**. For instructions see [Section 6.3.1.4, “Step 4: Defining module configurations”](#).
- ▶ For an existing project in the **Properties** dialog. For instructions see [Section 6.3.7.4, “Viewing information about the project's modules”](#).
- ▶ For an existing project in the **Module Configurations** dialog, as described below.

To edit the set of module configurations for a particular project, you need to open the **Module Configurations** dialog first:

- ▶ In the **Project Explorer**, right-click on the project or its ECU configuration node.
- ▶ Select **Module Configurations....**



The **Module Configurations** dialog opens up.

Figure 6.83. The **Module Configurations** dialog

The **Module Configurations** dialog consists of four areas:

- ▶ The **Available Modules** tree provides all modules which are available for the Target and Release Version of the project.
- ▶ In the **Default configuration settings** area, you may define the default pre- and recommended configuration for adding new module configurations.
- ▶ In the **Module Configurations** table, you see the module configurations that have been added to your project.
- ▶ The **Details** area provides detailed information either about a module selected within the **Available Modules** tree or a module configuration selected within the **Module Configurations** table. In the latter case it also provides some edit functionality for module configurations.

By clicking **Ok**, the **Module Configurations** dialog closes and any changes are applied to the project.

By clicking **Cancel** the **Module Configurations** dialog closes without applying any changes.

**WARNING****Changes are lost**

When canceling the dialog any changes you made are discarded without further notice.

6.6.1. Adding module configurations to your project

You may add single module configurations, several module configurations at once or clusters of module configurations to your project.

NOTE**You can recognize whether a module has been added by its font**

Modules that are still available and modules that already have been added to your project are displayed in different font styles in the **Available Modules** tree:

- ▶ Modules that have not been added to your project and thus are still available are displayed in bold font.
- ▶ Modules that already have been added to your project are displayed in standard font. The modules that allow multiple configurations are displayed in black. All others are grayed-out and cannot be added to your project anymore.

To add new module configurations to your project:

- ▶ In the **Available Modules** tree, select all modules/clusters of modules for which you want to add configurations for.
- ▶ Click the **Add** button

TIP**Add single modules with a double-click**

Module configurations for single modules can also be added by double-clicking the required module.

- ▶ If the required module configurations cannot be added to your project, the following dialog opens up:

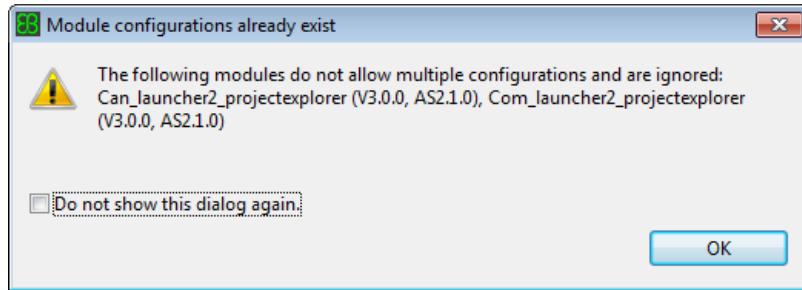


Figure 6.84. Some modules may be added only once to your project

The selected modules are added to your project as module configurations with default settings.

NOTE



Only one module configuration is added per selected module

Since modules can be part of more than one cluster, it is possible that a selection in the **Available Modules** tree refers several times to the same module. Note that only one configuration is added in this case, even if the module allows multiple configurations.

As long as the recently added module configurations have not been stored into your project yet, you may edit the (default) settings of the module configuration. Mind that depending on the module, there might be more tab pages next to the **General configuration** tab on the bottom of the **Details** area.

6.6.2. Removing module configurations from your project

You may remove single module configurations or several module configurations at once from your project.

NOTE



You can recognize the state of a module by its font

Module configurations that previously had been added to your project, mandatory module configurations and recently added module configurations are displayed in different fonts in the **Modules Configuration** table:

- ▶ Mandatory module configurations are displayed in italic font. It is not recommended to remove these module configurations from your project.
- ▶ Recently added module configurations are displayed in bold font. You may edit them.
- ▶ Module configurations that had been added to your project before are displayed in standard font.

To remove module configurations from your project:

- ▶ In the **Module Configurations** table, select the module configuration(s) you want to remove.

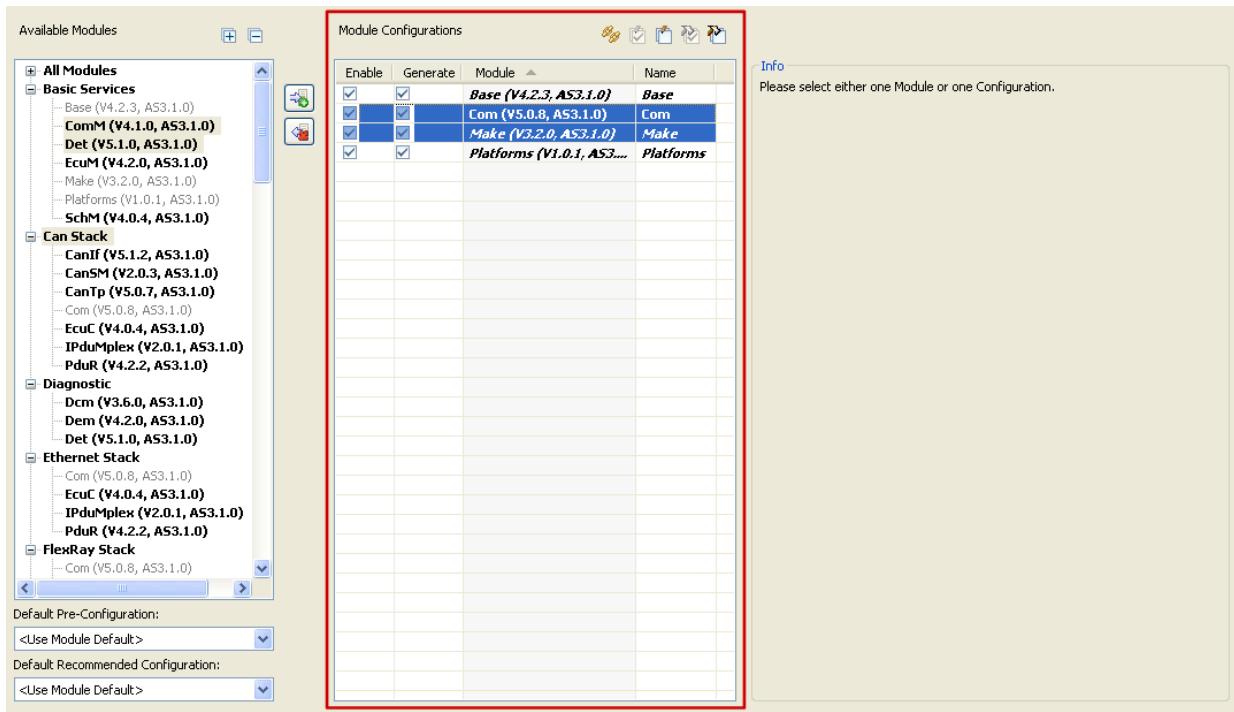


Figure 6.85. Removing module configurations

- ▶ Click the **Delete** button .
- ▶ If none of the selected module configurations refers to an already existing configuration file, the items selected are removed from the **Module Configurations** table without further notice. Otherwise the **Delete Module Configurations** dialog opens up

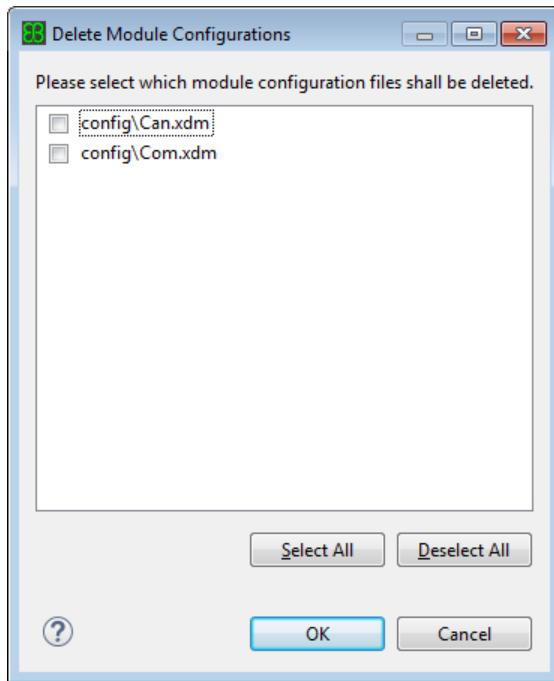


Figure 6.86. Confirming which module configuration files you want to remove

- ▶ Select the check box of the configuration files you want to delete.

NOTE**Data loss**

The selected files will be deleted from the file system when the changed module configurations are applied. After that, you cannot restore these files anymore.

- ▶ To confirm the deletion of the configuration files, click **OK**.

The items selected are removed from the **Module Configurations** table.

- ▶ To cancel the deletion of the **Module Configurations** and their referred configuration files, click **Cancel**.

6.6.3. Enabling and disabling module configurations

In the **Module Configurations** table you may (temporarily) enable or disable the module configurations you have added to your project. Disabled configurations are treated as if they do not exist, which means you cannot edit or verify them, and code generators cannot access them. To enable module configurations:

- ▶ To enable only one module configuration, select the **Enable** check box of the module configuration you want to enable in the **Module Configurations** table.



Module Configurations			
Enable	Generate	Module	Name
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms

Figure 6.87. Enabling one module configuration

- ▶ To enable several module configuration at once:
 - ▶ In the **Module Configurations** table, select the module configurations you want to enable.
 - ▶ In the tool bar of the **Module Configurations** table, click the **Enable module configuration** button .

Alternatively, you may enable module configurations in the **Project Explorer** view:

- ▶ To enable only one module configuration:
 - ▶ In the **Project Explorer** view, right-click on the module configuration you want to enable.

A context menu opens up.

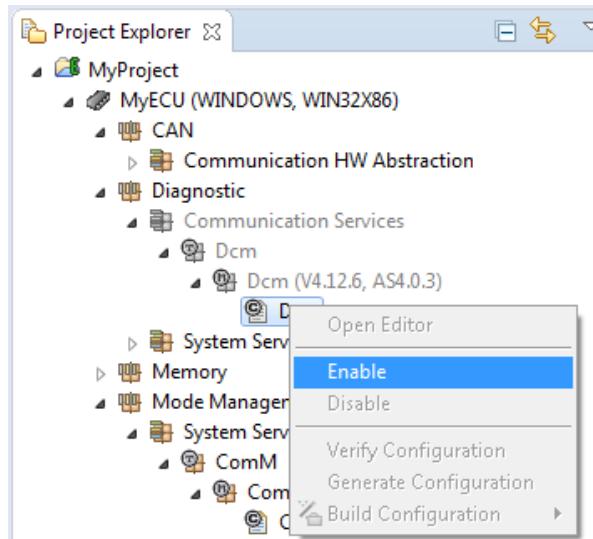


Figure 6.88. Enabling one module configuration

- ▶ In the context menu, select **Enable**.

The module configuration of the selected module is enabled and thus displayed in black standard font in the **Project Explorer** view.

- ▶ To enable all module configurations of one node:



- ▶ In the **Project Explorer** view, right-click on the category/layer/type/module node whose module configurations you want to enable.

A context menu opens up.

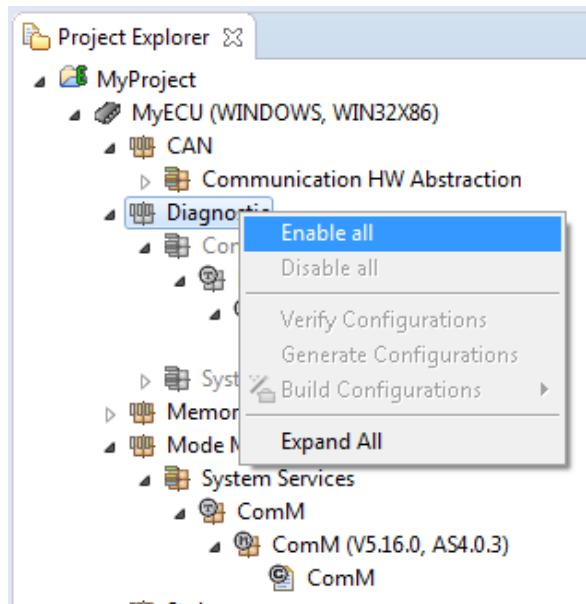


Figure 6.89. Enabling all module configurations of one node

- ▶ In the context menu, select **Enable all**.

NOTE



The Enable all command might not be available

The **Enable all** command is only available if the selected node contains more than one module configuration and if one of these module configurations is disabled. If the node contains only one module configuration, the **Enable** command is displayed instead.

The module configurations of the selected node are enabled and are displayed in black standard font in the **Project Explorer** view.

To disable module configurations:

- ▶ To disable only one module configuration in the **Module Configurations** dialog/page, clear the **Enable** check box of the module configuration you want to disable in the **Module Configurations** table.



Enable	Generate	Module	Name
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms

Figure 6.90. Disabling one module configuration

To disable several module configuration at once:

- ▶ In the **Module Configurations** table, select the module configurations you want to disable.
- ▶ In the tool bar of the **Module Configurations** page, click the **Disable module configuration** button .

Alternatively, you may disable module configurations in the **Project Explorer** view:

- ▶ To disable only one module configuration:
 - ▶ In the **Project Explorer** view, right-click on the module configuration you want to disable.

A context menu opens up.

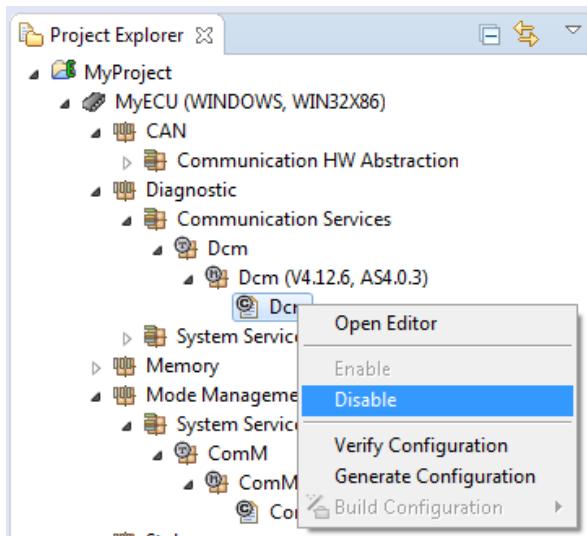


Figure 6.91. Disabling one module configuration

- ▶ In the context menu, select **Disable**.

The selected module configuration is now disabled and grayed out.

- ▶ To disable all module configurations of one node:
 - ▶ In the **Project Explorer** view, right-click on the category/layer/type/module node whose module configurations you want to disable.



A context menu opens up.

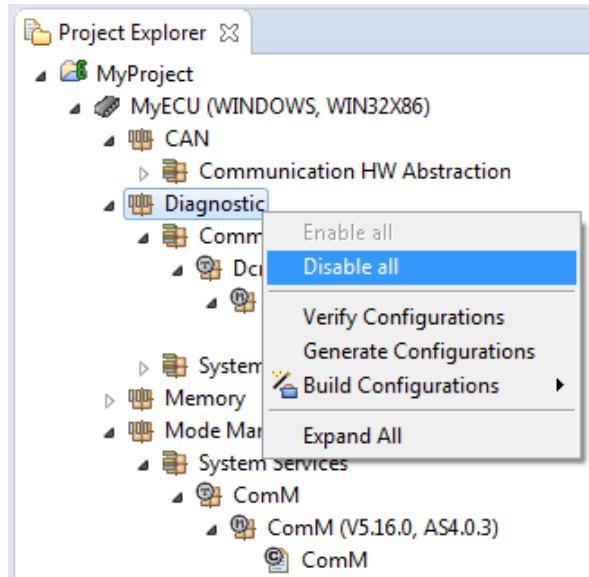


Figure 6.92. Disabling all module configurations of one node

- ▶ In the context menu, select **Disable all**.

NOTE**The Disable all command might not be available**

The **Disable all** command is only available if the selected node contains more than one module configuration and if one of these module configurations is enabled. If the node contains only one module configuration, the **Disable** command is displayed instead.

The selected module configurations are now disabled and grayed out.

6.6.4. Enabling and disabling code generation for module configurations

In the **Module Configurations** table you may enable or disable code generation for the module configurations of your project.

WARNING**Only partly generated code leads to incorrectly working ECUs**

If you only generate the code of a part of the module configurations you later want to implement on an ECU, depending on which modules you choose, the code might not work correctly. This might lead to incorrectly working ECUs.

To avoid incorrectly working ECUs, generate the code of all module configurations in your project before deploying them on an ECU.

To enable code generation for module configurations:

- ▶ To enable code generation for only one module configuration, select the **Generate** check box of the module configuration for which you want to enable code generation in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.93. Enabling code generation for one module configuration

- ▶ To enable code generation for several module configurations at once:
 - ▶ In the **Module Configurations** table, select the module configurations, for which you want to enable code generation.
 - ▶ In the tool bar of the **Module Configurations** page, click the **Enable code generation** button

To disable code generation for module configurations:

- ▶ To disable code generation for only one module, clear the **Generate** check box of the module configuration for which you want to disable code generation in the **Module Configurations** table.

Module Configurations				
Enable	Generate	Module	Name	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Base (V4.2.3, AS3.1.0)	Base	
<input type="checkbox"/>	<input type="checkbox"/>	CanSM (V2.0.2, AS3.1.0)	CanSM	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Com (V5.0.8, AS3.1.0)	Com	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Make (V3.2.0, AS3.1.0)	Make	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Platforms (V1.0.1, AS3....	Platforms	

Figure 6.94. Disabling code generation for one module configuration

- ▶ To disable code generation for several module configurations at once:



- ▶ In the **Module Configurations** table, select the module configurations, for which you want to disable code generation.
- ▶ In the tool bar of the **Module Configurations** page, click the **Disable code generation** button

6.6.5. Naming Module Configurations

When you add new module configurations to your project, the name of the new module configuration is set automatically to a name that is unique within the project using either the default configuration name provided by the module or the module name itself.

Instead of using the default name you may want to use another name for the module configuration.

NOTE
Only in recently added module configurations the name can be changed


You can only change the name of a module configuration if you added this module configuration during the same session of the dialog.

To edit the name of a module configuration:

- ▶ Add the required module configuration to your project.
- ▶ In the **Module Configurations** table, click into the **Name** cell of the module configuration whose name you want to change.
- ▶ Type the new name.



Figure 6.95. Editing the name of a new module configuration

NOTE
Name constraints


Mind that the module configuration name must meet the following constraints:

- ▶ The name must be unique within the project.
- ▶ The name must start with a letter.
- ▶ The name must contain only letters and digits plus the underscore character ('_').
- ▶ The name must not be empty.
- ▶ The name must not exceed 255 characters.

-
- ▶ Press **ENTER**.

The name of the module configuration is changed.



6.6.6. Choosing Pre-Configurations

Module suppliers optionally can equip modules with one or more preconfigurations that contain values for configuration parameters. If a module configuration of your project provides one or more preconfiguration(s), you must assign this module configuration to such a preconfiguration when adding it to your project. The parameters covered by the chosen preconfiguration are then fixed in the module configuration and cannot be edited by the user.

In the **Default configuration setting** area, you can set the **Default Pre-Configuration** which is used as a default value for all new module configurations. To support a more consistent selection, this preconfiguration setting is persisted. You can also edit the **Pre-Configuration** of an individual module configuration.

6.6.6.1. Setting the Default Pre-Configuration

In the **Default configuration setting** area of the **Module Configurations** page, you may edit the **Default Pre-Configuration** for new module configurations.

NOTE**The value set is a union set of all preconfigurations provided by all modules**

The preconfiguration names to choose from derive from the union set of all names provided by all modules that match the project's target architecture and AUTOSAR release version. If the chosen name is not valid for a specific module, the module's default preconfiguration is taken instead when you add it to your project.

To change the default preconfiguration, select the required preconfiguration in the **Default Pre-Configuration** drop-down list box.

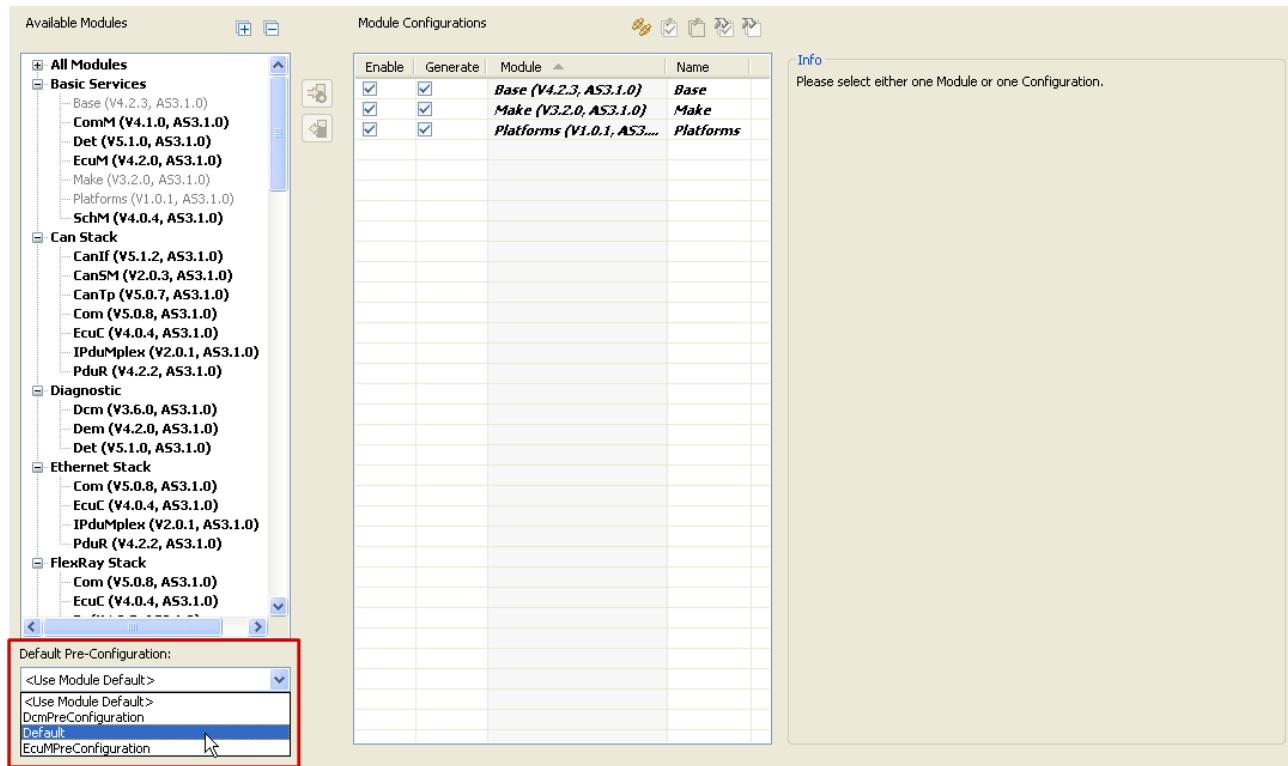


Figure 6.96. Selecting the default preconfiguration for new module configurations

- ▶ To use the respective default preconfiguration for new module configurations, select <Use Module Default>.

NOTE**Non-compilable code**

If you select <Use Module Default>, every module will be added to your project with a different preconfiguration. This might lead to incompatibilities of the module configurations and thus to non-compilable code.

- ▶ The selected default preconfiguration is saved, whenever you actually add new module configurations to your project.

6.6.6.2. Changing the preconfiguration of one specific module configuration

To change the preconfiguration of one specific module configuration:

- ▶ Add the required module configuration to your project.

**NOTE****Only in recently added module configurations the preconfiguration can be changed**

You can only change the preconfiguration of a specific module configuration if you added this module configuration during the same session of the dialog.

- ▶ In the **Module Configurations** table, mark the module configuration whose preconfiguration you want to change.

In the **Details** area of the **Module Configurations** page, detailed information about the specific module is displayed.

The screenshot shows the 'Module Configurations' dialog. On the left, there is a tree view of 'Available Modules' under 'All Modules'. The 'Com' module is selected in the main table, which has columns 'Enable', 'Generate', 'Module', and 'Name'. The 'Name' column shows entries like 'Base (V5.0.2, AS4.0.2)', 'Com (V6.1.0, AS4.0.2)', etc. To the right of the table is the 'Details' area, which is highlighted with a red border. This area contains several sections: 'Pre-Configuration:' dropdown, 'Recommended Configuration:' dropdown, 'Configuration File' section with checkboxes for 'Use default location' and 'Load existing file', 'Generation Path' section with a 'Use default location' checkbox, and an 'Info' section with details such as Module-Id: Com_TS_TxDxM6I1R0, Description: AUTOSAR Com Module, Copyright: (c) 2005-2012 Elektrobit Automotive GmbH, Clusters: FlexRay Stack, Allows multiple configurations: No, and Mandatory: General configuration.

Figure 6.97. The **Details** area in the **Module Configurations** page

- ▶ In the **Pre-Configuration** drop-down list box in the **Details** area, select the required preconfiguration for the selected module configuration.

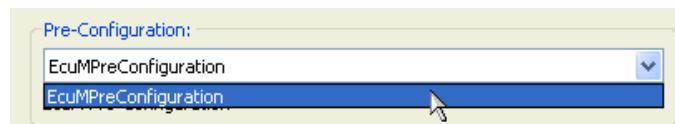


Figure 6.98. Changing the preconfiguration of one specific module

**NOTE****Grayed-out preconfigurations are not available**

If the **Pre-Configuration** drop-down list is grayed-out, the selected module does not provide a preconfiguration of its own.

6.6.7. Applying recommended configurations

Module suppliers optionally can equip modules with one or more recommended configurations that contain values for configuration parameters. If a module configuration of your project provides one or more recommended configurations, you can choose one when you add a configuration for this module to your project. The parameter values covered by the chosen recommended configuration can be considered as a guideline. They may be edited by the user later.

In the **Default configuration setting area**, you can set the **Default Recommended Configuration** which is used as a default value for all new module configurations. To support a more consistent selection, this setting is persisted. You can also edit the **Recommended Configuration** of an individual module configuration.

6.6.7.1. Setting the Default Recommended Configuration

In the **Default configuration setting** area of the **Module Configurations** page, you may edit the **Default Recommended Configuration** for new module configurations.

NOTE**The value set is a union set of all recommended configurations provided by all modules**

The names of the recommended configurations to choose from derive from the union set of all names provided by all modules that match the project's target architecture and AUTOSAR release version. If the chosen name is not valid for a specific module, the module's default configuration is taken instead when you add it to your project. If the module does not provide a default configuration, no recommended configuration is applied.

To change the default recommended configuration, select the required recommended configuration in the **Default Recommended Configuration** drop-down list box.

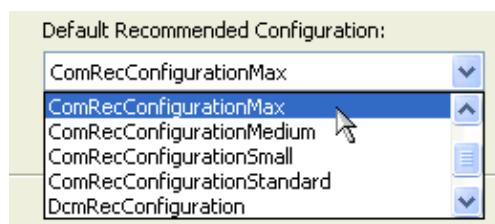


Figure 6.99. Selecting the default recommended configuration for new module configurations



- ▶ If you don't want to apply any of the recommended configurations, select <None>.
- ▶ To use the respective default recommended configuration for new module configurations, select <Use Module Default>.

NOTE**Non-compilable code**

If you select <Use Module Default>, every module will be added to your project with a different recommended configuration. This might lead to incompatibilities of the module configurations and thus to non-compilable code.

- ▶ The selected default recommended configuration is saved, when you actually add new module configurations to your project.

6.6.7.2. Changing the recommended configuration of one specific module configuration

To change the recommended configuration of one specific module configuration:

- ▶ Add the required module configuration to your project.

NOTE**Only in recently added module configurations the recommended configuration can be changed**

You can only change the recommended configuration of a specific module configuration if you have added this module configuration during the same session of the dialog.

- ▶ Mark the module configuration whose recommended configuration you want to change.

In the **Details** area of the **Module Configurations** page, detailed information about the specific module is displayed.



Figure 6.100. The Details area in the Module Configurations page

- In the **Recommended Configuration** drop-down list box, select the required recommended configuration for the selected module configuration.

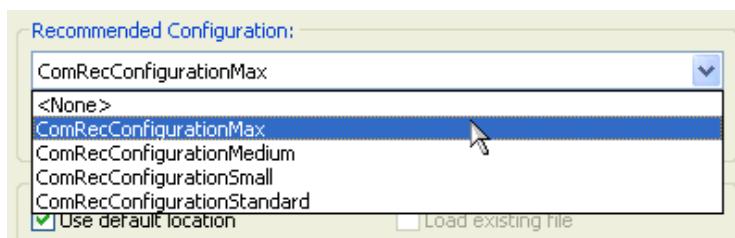


Figure 6.101. Changing the recommended configuration of one specific module

**NOTE****Grayed-out recommended configurations are not available**

If the **Recommended Configuration** drop-down list is grayed-out, the selected module does not provide a recommended configuration of its own.

6.6.8. Editing the configuration file settings

The configuration file stores all parameters for a specific module configuration. For detailed information on how to edit those parameters, e.g. by using the Generic Configuration Editor, see [Section 6.8, “Editing parameters of a module configuration”](#).

When you add new module configurations to your project, per default the configuration file name is set to config/<moduleconfigurationname>.xdm, relative to your project folder. This is also referred to as the default location.

Instead of using the default location you can use another location or name for the configuration file. To use another location or file name:

- ▶ Add the required module configuration to your project.

NOTE**Only in recently added module configurations the configuration file can be changed**

You can only change the configuration file of a module configuration if you have added this module configuration during the same session of the dialog.

- ▶ In the **Module Configurations** table, mark the module configuration whose configuration file you want to change.
- ▶ In the **Configuration File** panel of the **Details** area, clear the check box **Use default location**.

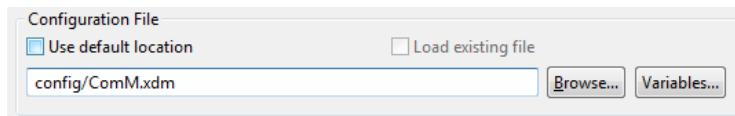


Figure 6.102. Editing the configuration file settings

- ▶ Enter or choose the new file name. Either use absolute path names or path names relative to your project. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#)

**NOTE****The file extension .xdm is mandatory**

You can only use file names ending with the extension `.xdm`. When you choose an existing file, you must make sure that it matches the targeted module configuration schema. Otherwise the new module configuration will fail to load when applied.

6.6.8.1. Using existing configuration files

Instead of configuring the module configuration from scratch you may also use an already existing configuration file.

NOTE**You cannot set preconfigurations and recommended configurations when you load an existing configuration file**

If you use an existing configuration file, you cannot apply any preconfiguration or recommended configuration.

To load an already existing configuration file into your project:

- ▶ In the **Configuration File** panel, browse to the file you want to load.

NOTE**The selected configuration file will be modified**

The selected configuration file will be modified while you edit the parameters of the module configuration. If you want to preserve the file as it is, copy the file to your project before loading it to your project.

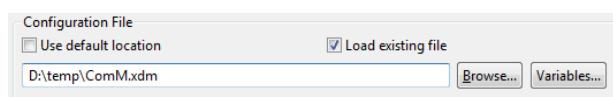


Figure 6.103. Using an existing configuration file

Whenever you point the **Configuration File** panel to an existing file in the file system, the **Load existing file** check box is marked.

**NOTE****Data loss**

If you clear the **Load existing file** check box, the existing file will be overwritten and cannot be restored again.

6.6.9. Editing the code generation output path settings

In **Generation Path**, specify the directory into which EB tresos Studio writes the generated code files for this module configuration.

When you add new module configurations to your project, the **Generation Path** is set to the **Default Generation Path** of the project. This is also referred to as the default location. For more information about the **Default Generation Path** of the project, see [Section 6.3.7.2, “Editing the Code Generator settings”](#).

Instead of using the default location you can use another location for the generated code of this module configuration. To use another location:

- ▶ In the **Module Configurations** table, mark the module configuration whose generation path you want to change.
- ▶ In the **Generation Path** panel of the **Details** area, clear the check box **Use default location**.



Figure 6.104. Editing the generation path settings

- ▶ Enter or choose the new path name. Either use absolute path names or path names relative to your project. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#).

6.6.10. Upgrading module configurations

To upgrade module configurations from older software versions of the modules to newer ones:

- ▶ Click the **Upgrade** button in the tool bar of the **Module Configurations** table.
- The **Upgrade** dialog opens up.

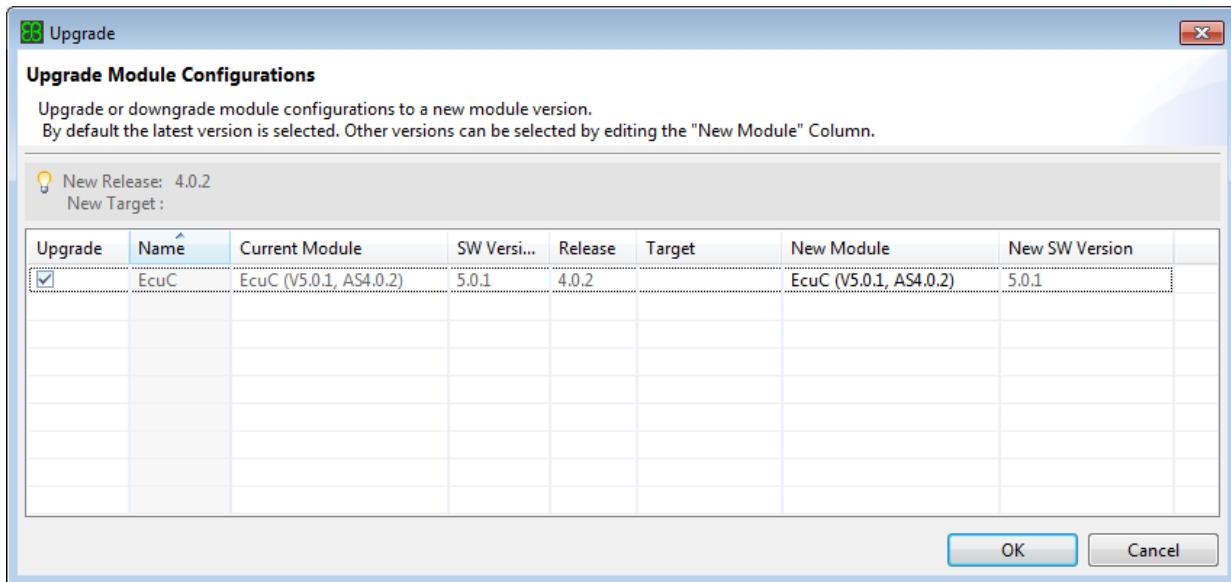


Figure 6.105. Upgrading module configurations in the **Upgrade** dialog

For detailed information on the **Upgrade** dialog, see [Section 6.7, “Upgrading projects and configurations”](#).

6.7. Upgrading projects and configurations

This chapter covers several aspects of working with projects and modules of different versions.

When a project is created, you have to decide on an AUTOSAR release version for the project. It may happen at a later point in your development cycle that you want to migrate your project to a newer AUTOSAR version. [Section 6.7.1, “Converting a project to a new AUTOSAR version”](#) describes how EB tresos Studio can help you with this.

Regardless of the AUTOSAR release version number, the modules themselves contain a version number, called the *software version*. It is possible that you have installed modules of the same AUTOSAR version, but with different software versions. If you have started a project with modules of an older software version, you may want to upgrade these modules to newer versions. For this case the **Module Configurations** dialog provides an upgrade function that is accessible via a button. The usage of this feature is explained in [Section 6.7.2, “Upgrading module configurations to new module versions”](#).

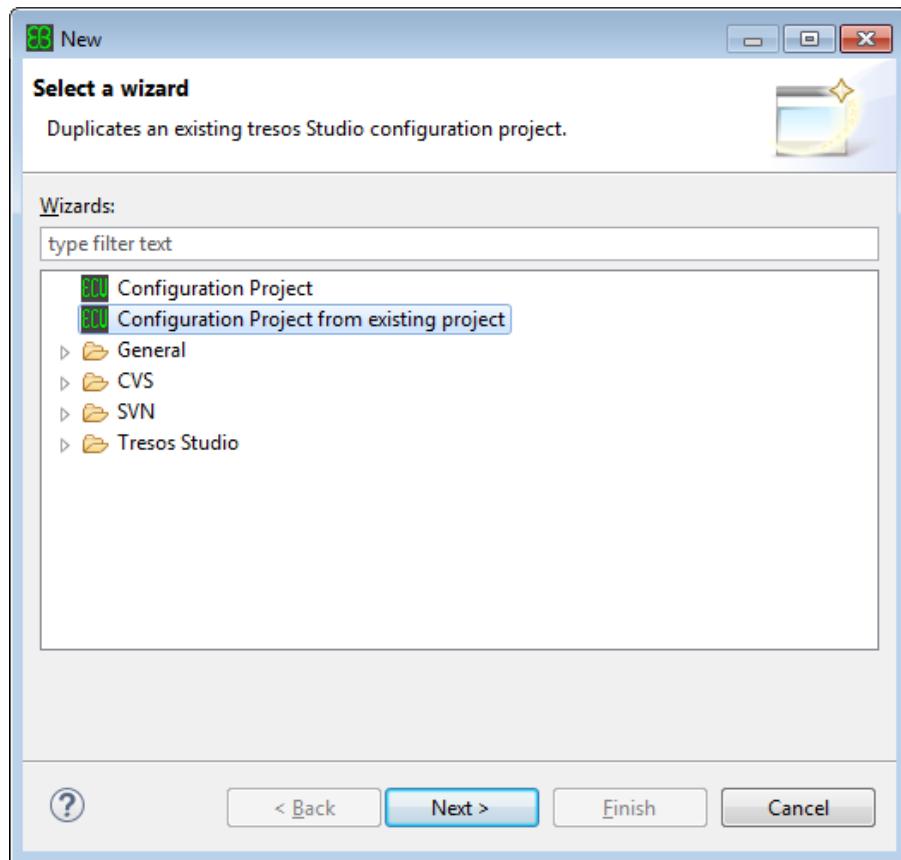
To upgrade projects and configurations you need to install plug-ins. To find out how to install different plug-ins, see the EB tresos® installation guide. As PDF the EB tresos® installation guide is located at \$TRESOS_BASE/doc/1.0_Installation with \$TRESOS_BASE being the location on your computer where you installed EB tresos Studio. To access the installation in the Eclipse help, open EB tresos Studio and press **F1**, then navigate to the folder **Installation**.



6.7.1. Converting a project to a new AUTOSAR version

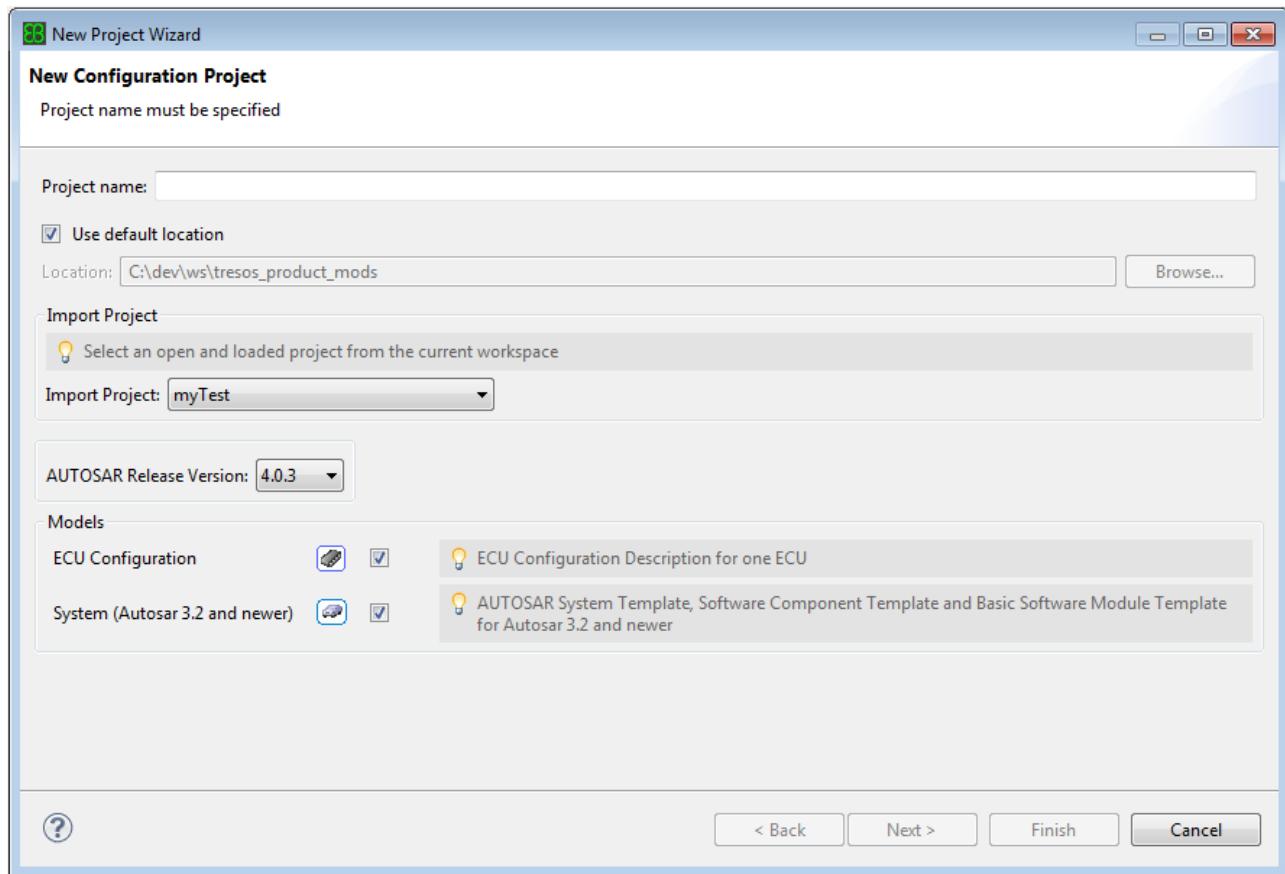
To convert a project (e.g. AUTOSAR 2.0 configuration) to a newer version (e.g. AUTOSAR 2.1 configuration):

- ▶ Follow the step 1 of [Section 6.3.1.1, “Step 1: Starting the Project Wizard”](#).
- ▶ In step 2, select **Configuration Project from existing Project**.



- ▶ Click **Next**.

The **New Configuration Project** window opens up.

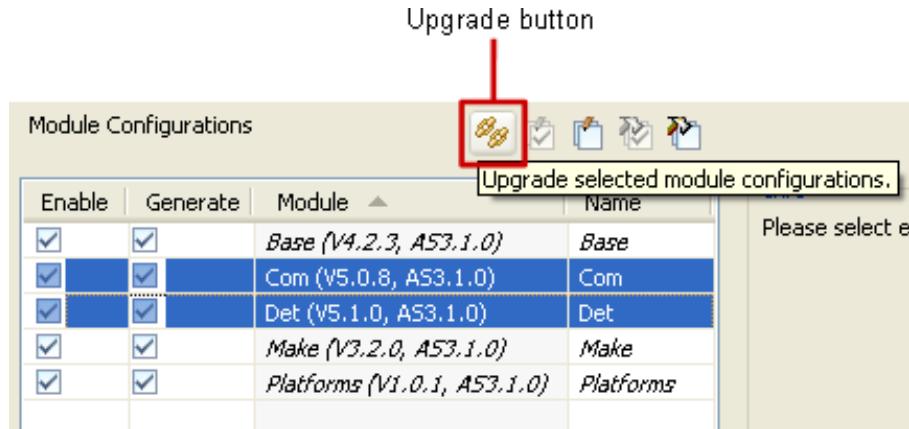


- ▶ Type in a name for the project in the **Project name** text box.
- ▶ Provide the source project to copy from in the drop-down listbox **Import Project**.
- ▶ Proceed with the wizard analog to [Section 6.3.1.2, “Step 2: Creating a new project”](#).

The drop-down listbox **Import Project** allows you to select only projects of your current workspace, which are open and loaded. For opening and loading closed projects, see chapter [Section 6.3.6, “Closing and opening projects”](#). If the project you would like to convert is not yet part of your workspace, see chapter [Section 6.10.1.1, “Importing a project”](#) to import the desired project first. After you have imported the project into your workspace, the **Import Project** drop-down listbox will automatically list the imported project as available.

6.7.2. Upgrading module configurations to new module versions

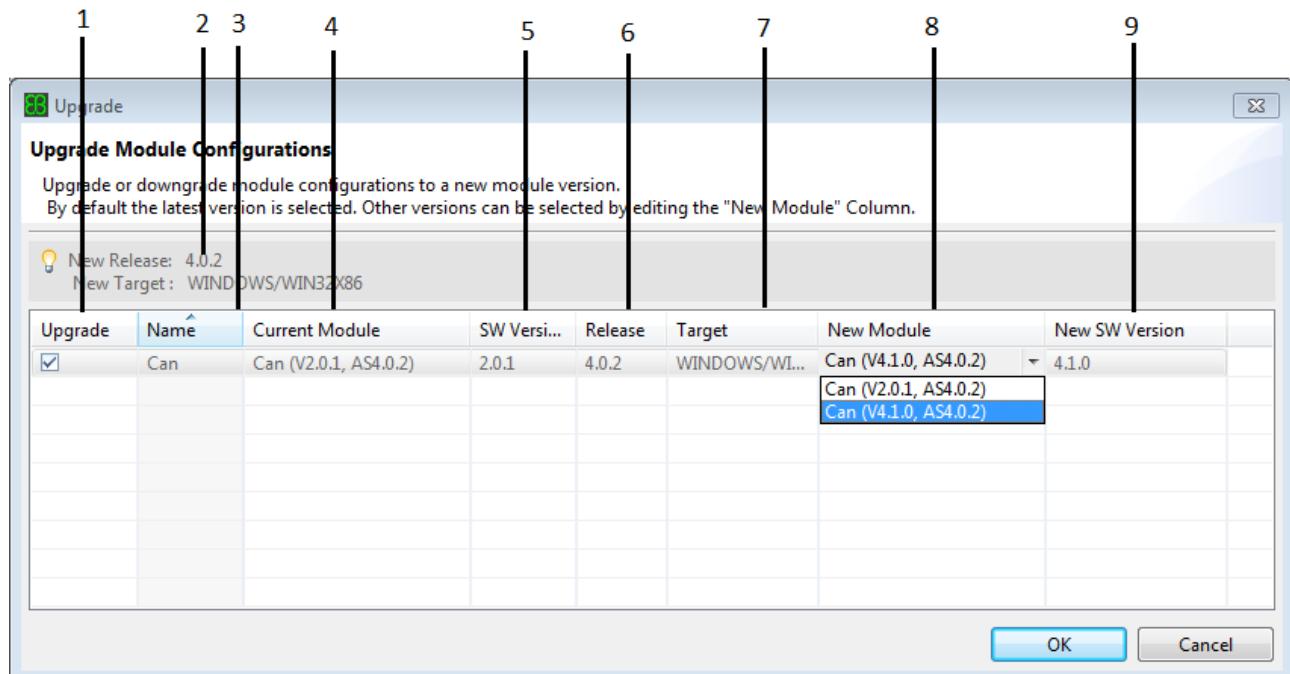
When new versions of modules are delivered and the configuration is changed, the new version is installed to EB tresos Studio as a new module with a new software version number. To switch a configuration between different versions of a module the **Module Configurations** dialog provides an upgrade function that is accessible via the **Upgrade** button.



The upgrade function is available from:

- ▶ The **New Project** wizard for new configuration projects (see: [Section 6.3.1, “Creating a new project”](#)).
- ▶ The **New Configuration Project from existing project** wizard for derived configuration projects (see: [Section 6.7.1, “Converting a project to a new AUTOSAR version”](#)).
- ▶ The **Module Configuration** dialog accessible via the context menu of the **Project Explorer** (see: [Section 6.6, “Editing module configurations”](#)).
- ▶ The **Properties for project name** dialog accessible via the context menu of the **Project Explorer** (see: [Section 6.3.7, “Viewing and changing project properties”](#)).

If you click the **Upgrade** button, the **Upgrade** dialog is shown. When module configurations are selected in the **Module Configurations** dialog, only these are selected for upgrade if no configurations are selected all module configurations are presented for upgrade.





Number	Description
1	Column Upgrade : The dialog allows to select the module configurations that are actually upgraded via the check boxes in the column labeled Upgrade .
2	The info box shows release version and the target of the project which matches all presented new modules.
3	Column Name : This column shows the name of the module configuration.
4	Column Current Module : Shows the name of the module that is currently used for the configuration.
5	Column SW Version : Shows the software version of the module that is currently used for the configuration.
6	Column Release : Shows the release version of the module that is currently used for the configuration.
7	Column Target : Shows the target and derivate of the module that is currently used for the configuration.
8	Column New Module : Allows to select the new module to use for the configuration. By default the latest available is selected. Clicking into a cell of this column presents a drop down box that allows to switch the module.
9	Column New SW Version : Shows the software version of the selected module.

Pressing the **Ok** button closes the dialog and updates the Module Configuration dialog. Closing the Module Configuration dialog actually switches the module configuration of the project to their new module definitions adapting configuration parameters if necessary.

NOTE**Upgrade module configuration using the command line**

You can trigger the upgrade for module configurations of a project using the EB tresos Studio command line interface, too. For more information, see [Section 6.12.2.12, “Upgrading module configurations for a project”](#).

6.8. Editing parameters of a module configuration

6.8.1. Overview

In this chapter you learn how to edit parameters of a module configuration,

- ▶ either manually with the ECU Configuration Editor in [Section 6.8.3, “Opening the ECU Configuration Editor”](#),



- ▶ or automatically with unattended wizards [Section 6.8.10, “Autoconfiguring routine jobs”](#).

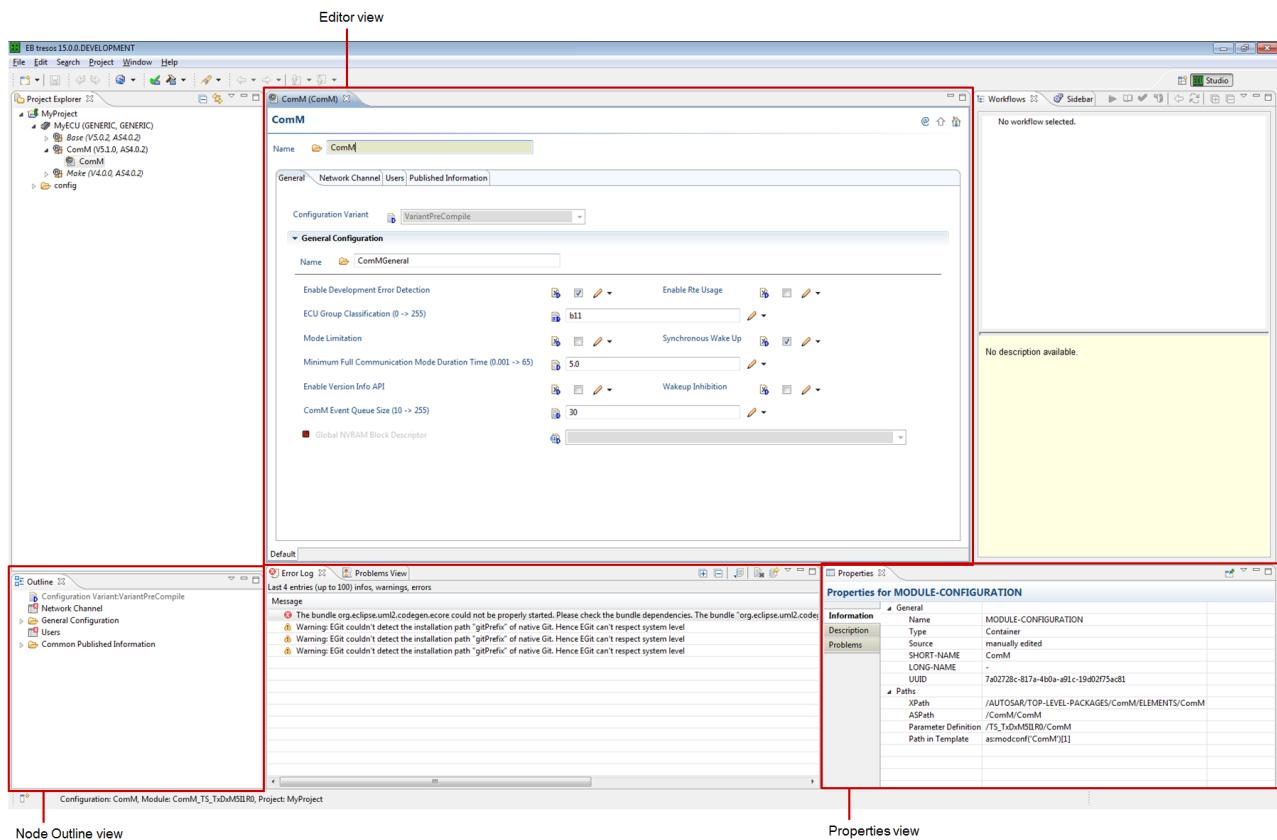
This chapter also shows you how to search for specific configuration parameters in [Section 6.8.11, “Searching for configuration parameters”](#).

6.8.2. Background information

EB tresos Studio provides a standard implementation of a Generic Configuration Editor that takes the parameter definition of a module as input and creates an editor. You can change the configuration data in the editor. Depending on the content of the editor and the currently selected editor element, several other views change their content and selection.

An editor may consist of:

- ▶ the **Editor view** in the center of the EB tresos Studio window, which enables editing the configuration data of a project
- ▶ the **Node Outline** view on the left bottom of the workbench, which shows a tree view of the editable configuration parameters that you may edit in the **Editor view**
- ▶ the **Properties** view on the right bottom of the workbench. It consists of three tabs that show information about the currently selected parameter in the **Editor view**





Detailed information on the Generic Configuration Editor is available in [Section 5.4.1, “The Generic Configuration Editor”](#).

For a detailed description of the **Node Outline** view and **Properties** view, see [Section 5.3.3, “Node Outline view”](#) and [Section 5.3.9, “Properties view”](#).

6.8.3. Opening the ECU Configuration Editor

To configure a module, you need to open the ECU Configuration Editor:

- ▶ Expand the tree underneath the project in the **Project Explorer**.
The **Project Explorer** tree opens up to display several layers. It depends on your **Project Explorer** display settings which layers are displayed.
- ▶ Select the module configuration you want to edit. Either
 - ▶ double-click on the module configuration
 - ▶ or right-click on the module configuration and select **Open Editor**.

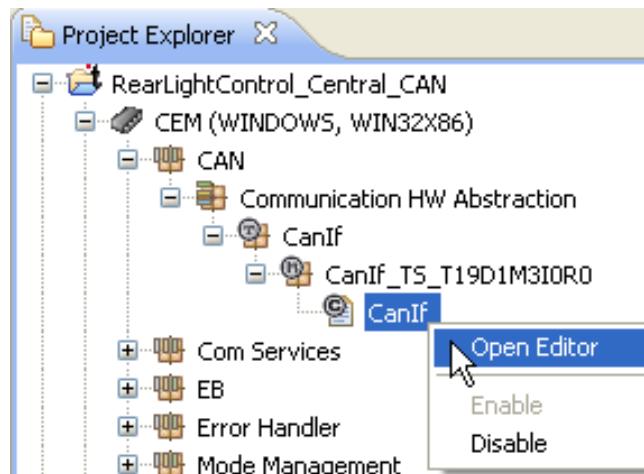


Figure 6.106. Opening the ECU Configuration Editor

The corresponding **Editor** view opens up to the right of the Project Explorer.

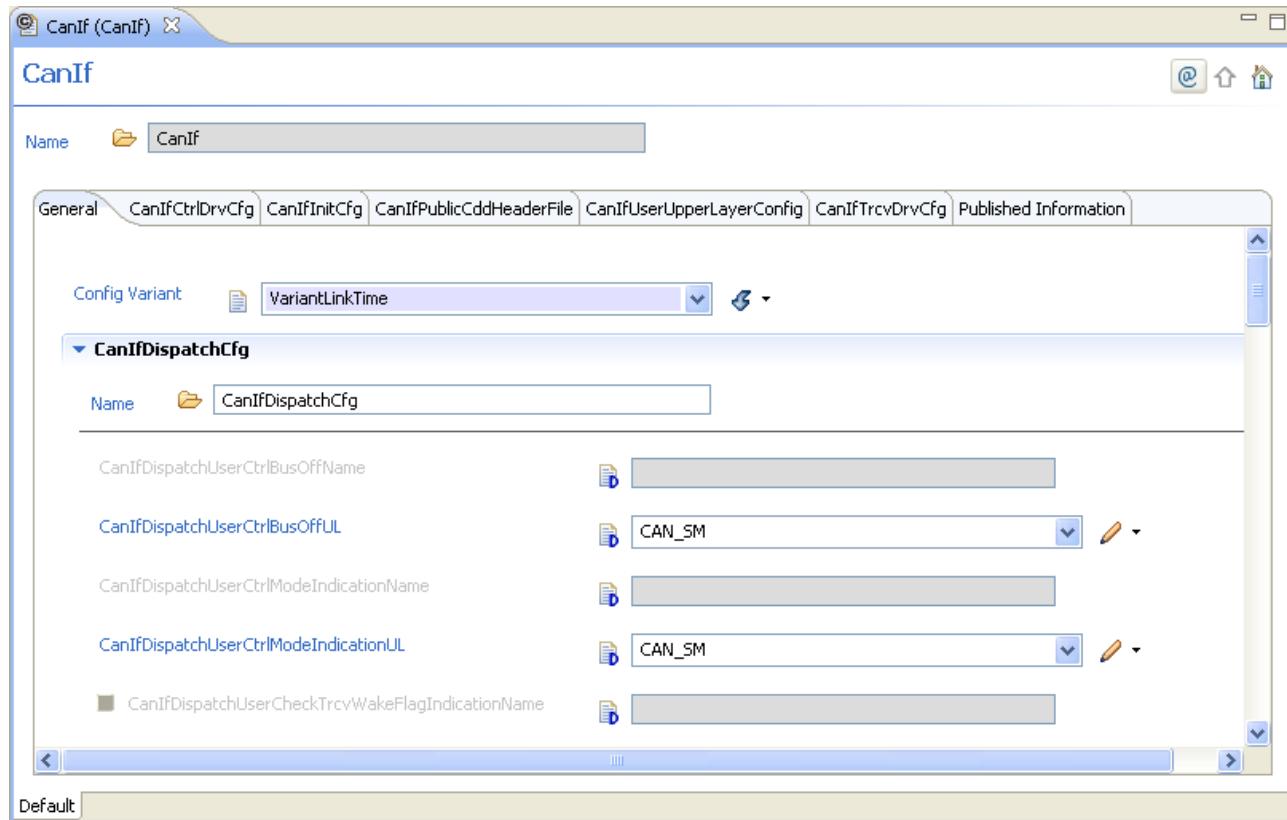


Figure 6.107. The Editor view

An **Editor** view of a module configuration consists of several pages, which are displayed as tabs in the **Editor** view. Please refer to [Section 5.3.2, “The Editor view”](#) to get to know the basic set-up of this view and its basic functionalities.

6.8.4. Setting the configuration variant of the module

The first parameter on the **General** tab is the configuration variant, displayed as **Config Variant** drop-down list box in the **Editor** view. The configuration variant defines which configuration class is used for each parameter of the module and influences the editable state of the parameters. For background information on the configuration variant, refer to [Section 3.5.2.1, “Configuration time of projects”](#).

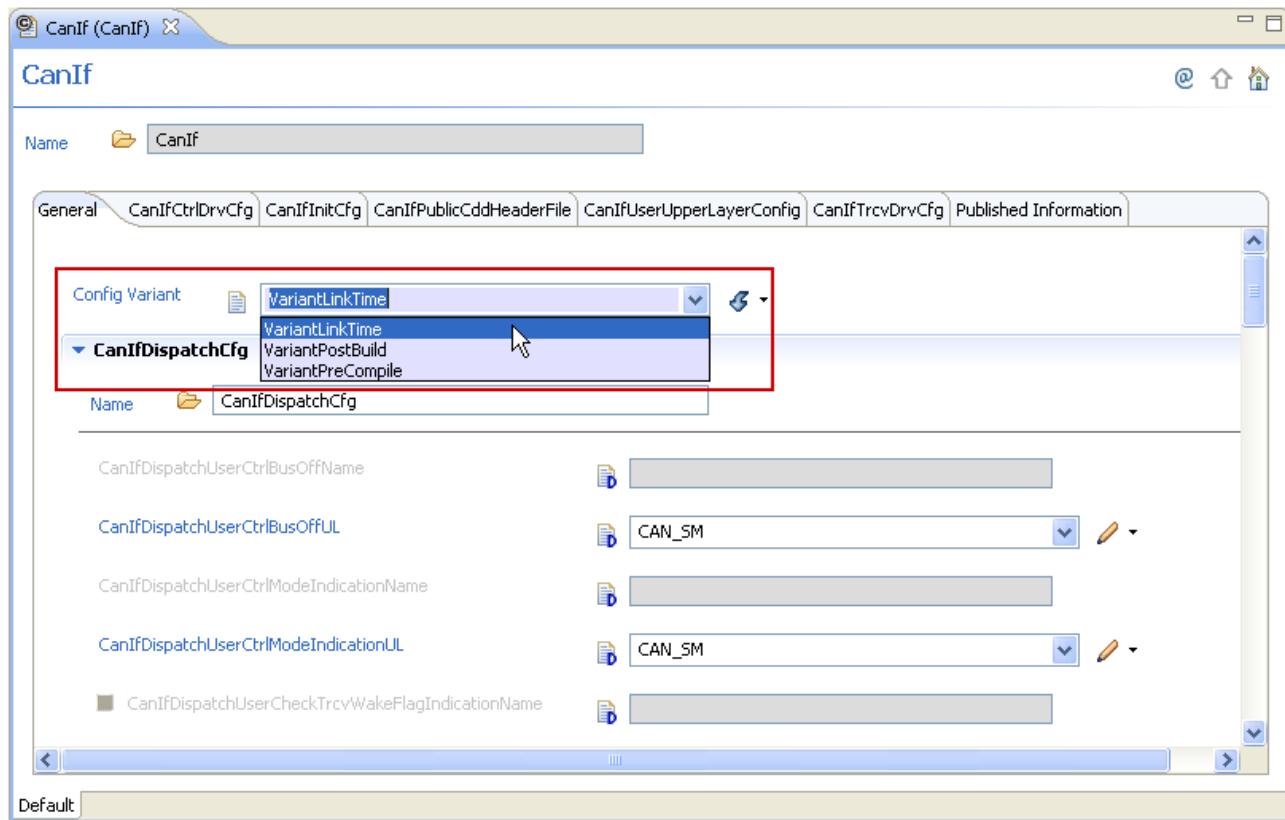


Figure 6.108. The configuration variant parameter

- ▶ To specify that the module is configured before the source code is compiled, select **VariantPreCompile**.
- ▶ To specify that the object code of the module receives parts of its configuration from another object code file, select **VariantLinkTime**.
- ▶ To specify that the module gets the parameters of its configuration, select **VariantPostBuild**. The module either downloads a separate file to the ECU memory (**POST-BUILD-LOADABLE**) or defines multiple configuration sets and chooses one during boot-time (**POST-BUILD-SELECTABLE**).

6.8.5. Editing variant conditions

For all variant-affected parameters it is possible to edit the variant conditions. A variant-affected parameter is marked with a on the type icon of the parameter.

To edit variant conditions, take the following steps:

1. In the Generic Configuration Editor or in the **Node Outline**, click or right-click the type icon of the required parameter.

A context menu opens up.

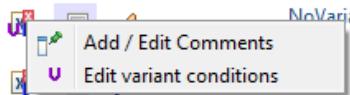


Figure 6.109. Editing variant conditions

TIP**Finding context menus in the Generic Configuration Editor**

To find out whether a context menu is available, move your mouse pointer over the type icon. If the pointer turns into a hand, a context menu is available. To open it, click or right-click the icon.

2. In this context menu, select **Edit variant conditions**.

The **PostBuildVariantConditions for <parameter name>** dialog opens up.

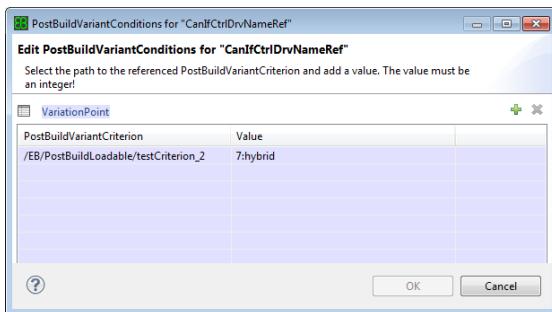


Figure 6.110. Editing PostBuildVariantConditions

The table shows the following information:

- ▶ PostBuildVariantCriterion

The cell displays the path to the selected PostBuildVariantCriterion.

The drop-down list box only contains paths to valid criterions. If the parameter is loadable then all paths to loadable criterions are shown and if the parameter is selectable, then all paths to selectable criterions are available. If a parameter supports both, all paths are available. For further information on how to find out if a parameter is Loadable or Selectable, see [Section 6.13.1.2, “PostBuild variation points”](#).

You can look up the mapping from PredefinedVariants to PostBuildVariantCriterions and to which type (loadable or selectable) they belong in the **Edit PostBuildVariants** wizard. For further information, see [Section 7.7, “Using the Edit PostBuildVariants wizards”](#).

- ▶ Value

The cell displays the criterion value. If there exists a textual representation for the integer value, then both are shown in the way <integer value>:<textual representation>



The drop-down list box provides all possible values for the selected criterion. If you already know the value you want to select, then you can also type in the value, either the integer value or the textual representation. If you type in an integer value which is not valid, the value is also accepted. If you type in a textual representation which is not valid, then this value is refused.

One row in the table represents one PostBuildVariantCondition which consists of the reference to the PostBuildVariantCriterion and the integer value.

NOTE**Generic Configuration Editor always shows the currently selected variant**

The Generic Configuration Editor always shows only parameter which match the currently selected variant. So be aware that after you change the variant conditions, the parameters are removed or exchanged.

If the variant condition of a mandatory parameter was changed in a way that the parameter is not part of the current variant anymore, the parameter in the editor is exchanged by a parameter that matches the conditions. If such a parameter does not already exist, a new instance of this parameter is created and displayed in the GUI. A new parameter always gets the variant conditions for the currently selected variant.

In the **Node Outline** view it is possible to display all instances of a parameter even if the variant conditions do not match the currently selected variant. For more information on how to display parameters for all variants, see [Section 5.3.3, “Node Outline view”](#).

If the variant condition of a list entry was changed in a way that the entry does not match the current variant, the list entry gets disabled in the editor and the entry is removed from its list.

6.8.6. Adapting references when you rename containers

When you rename a container in EB tresos Studio, it is usually necessary to adapt all references which refer to or into this container.

You can decide to let EB tresos Studio automatically adapt those references:

1. In the Generic Configuration Editor, rename a container.

A dialog opens up.

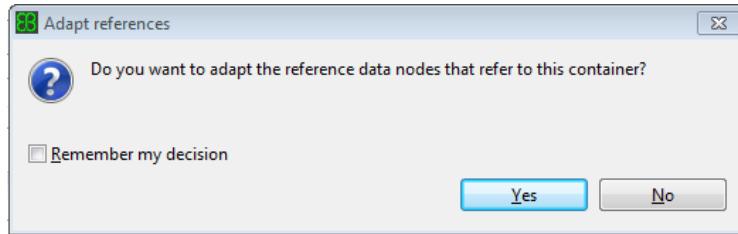


Figure 6.111. Adapt references

2. In this dialog, select **Yes**.

If there are any references that refer to the renamed container, they are adapted accordingly.

If you do not want to adapt the references, select **No**.

3. If you want EB tresos Studio to remember your decision and do not want the dialog to appear again, select the check box in the dialog.

If you want to modify your decision or want the dialog to appear again, you can modify your decision in the **Preferences EB tresos Studio** window. The chapter [Section 6.2.2.1, “Setting general preferences”](#) covers how to modify your decision.

TIP**Undoing/redesigning rename operation**

All applied changes to container as well as the references are applied to the undo stack. Thus, you may undo or redo the changes.

6.8.7. Adding parameter comments

In EB tresos Studio, you may add comments to a parameter. In this comment you may add a summary, a description and set the current status of this comment. It is possible to add several comments in different languages to a parameter.

TIP**Undoing/Redesigning changes of parameter comments**

All applied changes to parameter comments as well as remove comment actions are applied to the undo stack. Thus, you may undo or redo the changes.

6.8.7.1. Adding and editing a parameter comment

To add or edit a parameter comment:



1. In the Generic Configuration Editor, click or right-click the type icon of the required parameter.

A context menu opens up.



Figure 6.112. Adding or editing a parameter comment

TIP



Finding context menus

To find out whether a context menu is available, move your mouse pointer over the type icon. If the pointer turns into a hand, a context menu is available. To open it, click or right-click on the icon.

2. In this context menu, select **Add/Edit Comments**.

The **Comments for <parameter name>** dialog opens up.

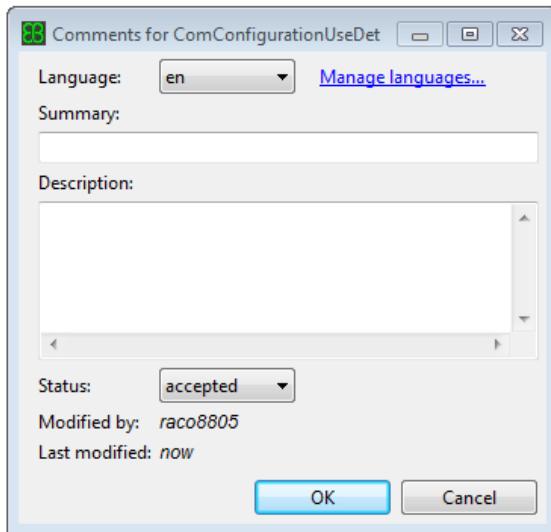


Figure 6.113. Adding a new comment to a parameter

- ▶ In the **Language** drop-down list box, select the language in which you want to write the comment.

To change the set of languages that appear in the **Language** drop-down list box, click the **Manage languages...** link next to the **Language** drop-down list box. For further information on changing the languages for the comments, see [Section 6.8.7.2, “Managing languages of parameter comments”](#).

- ▶ In the **Summary** text box, you may add a summary to your comment. The summary is then displayed in the Generic Configuration Editor when hovering with the mouse pointer over the parameter icon.
- ▶ In the **Description** text field, you may add your comment.



- ▶ In the **Status** drop-down list box, select a status for your comment.

The fields **Modified by** and **Last modified** are filled in automatically by EB tresos Studio

You may now choose between two alternative steps to continue:

- ▶ To apply the changes and close the dialog, click **OK**.
- ▶ To discard the last changes and close the dialog, click **Cancel**.

A confirmation dialog appears if your comment contains unapplied changes and you

- ▶ have changed the language in the **Language** drop-down list box, or
- ▶ have selected another configuration parameter in the Generic Configuration Editor or via the **Outline** view.

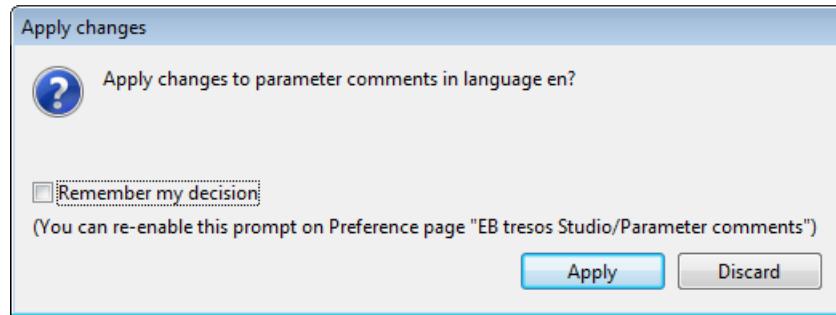


Figure 6.114. The Apply changes dialog

If you check the **Remember my decision** check box, this dialog does not appear again. Instead EB tresos Studio uses your choice for all unapplied changes.

NOTE



Turning confirmation dialogs on and off

In the EB tresos Studio preferences you may change whether this confirmation dialogs shall appear or not. For further information on the EB tresos Studio preferences, refer to [Section 6.2.2.11, “Setting preferences for parameter comments”](#).

6.8.7.2. Managing languages of parameter comments

To manage the languages that are available in the **Language** drop-down list box of the **Comments for <parameter name>** dialog, select the **Manage Languages ...** link in the **Comments for <parameter name>** dialog.

The **Manage Languages** dialog opens up:

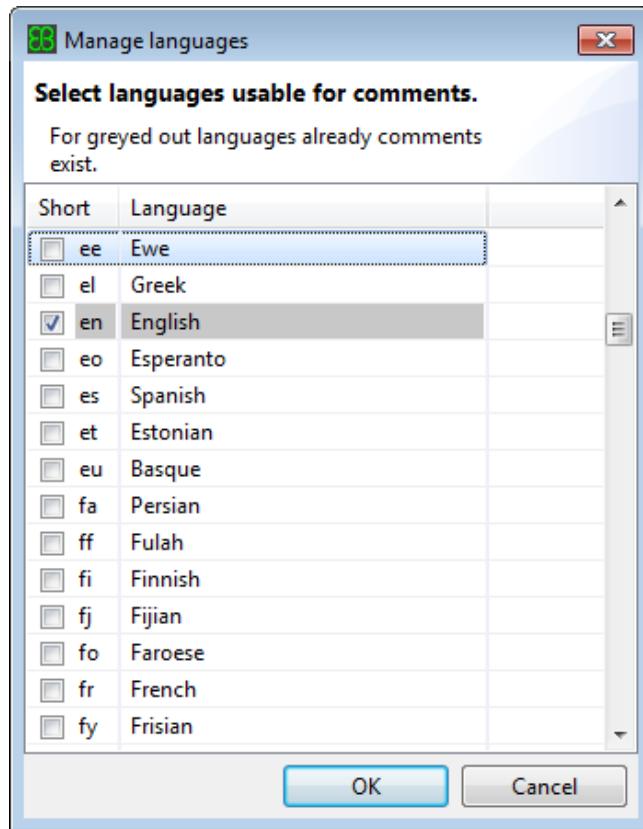


Figure 6.115. Managing languages for parameter comments

- ▶ To add a new language, select the required language. Click **OK** to apply your changes and close the dialog. The new language is now available in the **Language** drop-down list box of the **parameter comment dialog**.
- ▶ To remove a language select the language you want to remove. Click **OK** to apply changes and close the dialog. The removed language is also removed from the language box of the **Comments for <parameter name>** dialog.

NOTE**You cannot remove the default language**

The default language **en** as well as all languages for which comments already exist, are highlighted in gray background color and cannot be removed.

6.8.7.3. Removing parameter comments

To remove all comments for one parameter, click or right-click the type icon of the required parameter and select **Remove Comments** from the context menu that opens up.



Figure 6.116. Removing parameter comments

A confirmation dialog appears:

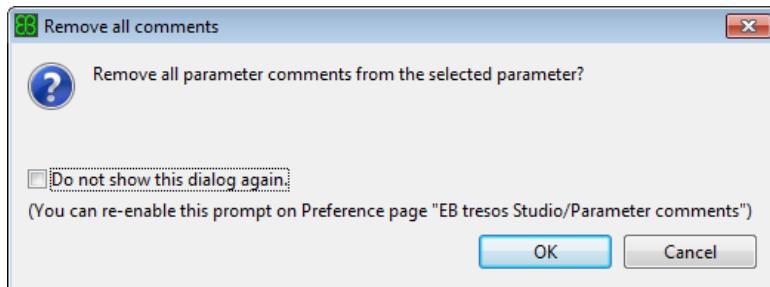


Figure 6.117. Confirming the removal of parameter comments

If you check the **Do not show this dialog again** check box and click **OK**, this confirmation dialog does not appear again until you reset the preference (see [Section 6.2.2.11, “Setting preferences for parameter comments”](#)).

6.8.7.4. Generating a parameter comment report

With the application example *DemoReportGenerator*, you can generate a report for the configuration of your project which also includes parameter comments. See [Section 6.8.8, “Generating a configuration report which includes comments”](#) for information on how to generate the report.

6.8.8. Generating a configuration report which includes comments

With the application example *DemoReportGenerator*, you can generate a report for the configuration of your project which includes any parameter comments.

6.8.8.1. Installing the application example *DemoReportGenerator*

1. Close EB tresos Studio to install the application example.
2. Copy the *DemoReportGenerator* from <tresos-install-dir>/demos/Studio to the plugins directory <tresos-install-dir>/plugins.
3. Restart EB tresos Studio.



6.8.8.2. Generating a report

1. Load the configuration of the project you want to create a report for into EB tresos Studio.
 2. Select the configuration project node in the **Project Explorer**.
 3. In the tool bar, click the **View** button next to the **Generate** button
- A context menu opens up.
4. In the context menu, select **Configuration Report (incl. Comments)**.

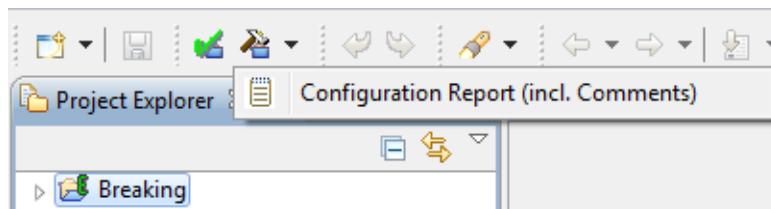


Figure 6.118. Generating a configuration report

The configuration report is generated. You find this report at <project-root>/output/reports/ecu_configuration_report.csv.

6.8.8.3. Understanding the configuration report

After you generated the configuration report, you find it at <project-root>/output/reports/ecu_configuration_report.csv.

Open the ecu_configuration_report.csv file in any spreadsheet program.

You see a table with several rows, one for each configuration parameter. The available columns are described below.

	A	B	C	D	E	F	G	H	I	J	K
1	Project: test										
2											
3	Module	Summary	Comment	Status	User	Date	Language	Value	Path		
1			You can add a multi-line description.	todo	masc4319	2011-01-12T11:56:19 CET	en				
25	PublicApiTest	This is a summary	Line description.						ExamplesCommon	Bool	
26	PublicApiTest								ExamplesCommon	Int	
27	PublicApiTest								ExamplesCommon	String	
28	PublicApiTest								ExamplesCommon	StringList	
29	PublicApiTest						Default		ExamplesResourceFile	SubDerivate	
30	PublicApiTest						4000000.0		ExamplesResourceFile	InputFrequency	
31	PublicApiTest						PRE_1.0		ExamplesResourceFile	OutputFrequencyPrescaler	
32	PublicApiTest								ExamplesResourceFile	FeatureCan	
33	PublicApiTest								ExamplesResourceFile	FeatureCan	CanParam1
34	PublicApiTest								ExamplesResourceFile	FeatureCan	CanParam2

Figure 6.119. A configuration report

1	Module	Displays the module name of the node currently selected.
2	Summary	Displays the summary of the comment of the current node. If no comment has been added to this parameter, this cell is empty.



3	Comment	Displays the comment description of the current node. If no comment has been added to this parameter, this cell is empty.
4	Status	Displays the status of the comment of the current node. If no comment has been added to this parameter, this cell is empty.
5	User	Displays the name of the user who has added the comment for the current node. If no comment has been added to this parameter, this cell is empty.
6	Date	Displays the date and time of the last edit of the comment. If no comment has been added to this parameter, this cell is empty.
7	Language	Displays in which language the comment is written. If no comment has been added to this parameter, this cell is empty.
8	Value	If the node is a parameter that contains a value, this value is displayed in this column.
9	Path	Displays the path of the root node in the model. To enhance filter and search support, the nodes of the path are displayed in the columns to the right of the Path column.

6.8.8.4. Customizing the DemoReportGenerator

The DemoReportGenerator uses the template-based Code Generator API. To adapt it to your needs, edit the template file `ecu_configuration_report.csv` in the subdirectory `generate/reports`.

For details about code templates, see the EB tresos Studio developer's guide, chapter Template-based Code Generator API.

6.8.9. Editing several table rows simultaneously

To change several rows of a table at the same time, EB tresos Studio provides the **Bulk Change** dialog. This **Bulk Change** dialog keeps you from having to change the same value in different rows manually by automating the value change over several rows.

- ▶ To bulk change values in several table rows, select all table rows in which you would like to change values. For details about tables and its entries, see [Section 5.4.1.4, “Tables”](#).
- ▶ Open the **Bulk Change** dialog.

To open the **Bulk Change** dialog, click the **Bulk change** button  in the table tool bar.

The **Bulk Change** dialog opens up.

- ▶ Alternatively, you may open the **Bulk Change** dialog from the context menu of a table.



To open the **Bulk Change** dialog via the context menu of the table, right-click on the table and select **Bulk Change...** from the context menu.

The **Bulk Change** dialog opens up.



Bulk Change

Edit the columns that should be changed

5 rows are selected for this bulk change operation.
Use check button to the left to select the columns that should be changed.

Name	Change Parameter Value	Change Enablement
CanControllerPhysicalChannel	<input checked="" type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> FCAN2	<input checked="" type="checkbox"/>
CanIfBusoffNotifFun	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/>	
CanIfWakeupNotifFun	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/>	
CanIfWakeupValidNotifFun	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/>	
CanControllerRef	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> @	<input checked="" type="checkbox"/>
CanControllerActivation	<input checked="" type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> X <input checked="" type="checkbox"/>	
CanControllerBaudRate	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> 100	
CanControllerId	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> 3	
CanControllerPropDelay	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> 3	
CanControllerTimeQuanta	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> 1.0	
CanControllerTseg1	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> 3	
CanControllerTseg2	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> 3	
CanRxProcessing	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> Interrupt	<input checked="" type="checkbox"/>
CanTxProcessing	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> Interrupt	<input checked="" type="checkbox"/>
CanWakeupProcessing	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> Interrupt	<input checked="" type="checkbox"/>
CanMaxUsedHTHs	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/>	
CanBusoffProcessing	<input checked="" type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> Polling	<input checked="" type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/>
CanBusoffProcessing_1	<input checked="" type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> Interrupt	<input checked="" type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/>
CanBusoffProcessing_2	<input type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/> Interrupt	<input checked="" type="checkbox"/> <input style="width: 20px; height: 20px; vertical-align: middle;" type="button" value="..."/>

OK **Cancel**

Figure 6.120. The Bulk Change dialog



The **Bulk Change** dialog provides an entry field for each parameter, including optional parameters, and parameters located inside optional containers, regardless if they are visible or not in the table. You can change the value of a parameter or enable it, or do both.

- ▶ To change which columns are displayed, use the tool bar of the table. For information on the tool bar of the table, see [Section 5.4.1.4, “Tables”](#).
- ▶ To edit the value of a field, click the check box listed under the section **Change Parameter Value** that follows the name of the parameter.
- ▶ If the parameter is optional, the **Change Enablement** section is displayed. To change the enablement, click the check box below the section **Change Enablement** and the **Enablement** button becomes active. This button has two states: the green state means enabled and the red state means disabled. The same representation is used in the Generic Editor.

The parameters are changed to the value *entered/selected* when you close the **Bulk Change** dialog.

- ▶ To execute the change, click on **Ok**.
- ▶ To cancel the operation, click on **Cancel**.
- ▶ To undo the operation, click on the undo button in the EB tresos Studio tool bar.

6.8.10. Autoconfiguring routine jobs

If you use the ECU Configuration Editor, you have to edit all values manually. This may be a tedious task. EB tresos Studio supports you performing those routine jobs by supplying *unattended* wizards. These wizards guide you through difficult configuration jobs and handle routine tasks that do not require user-interaction.

Each unattended wizard has its own graphical user interface which is suited for the wizard's task. You need to configure each unattended wizard before you can run it within your project.

The list of available wizards can be extended by wizards that are designed for the modules installed. By default, EB tresos Studio ships with some wizards that are described in the following sections. The module-specific unattended wizards are explained in the module's documentation.

NOTE

Unattended wizards can serve other purposes



Not only can you use unattended wizards to automatically edit module configuration, but they can be used for any repetitive task that you run during your project that uses the same configuration.

6.8.10.1. Configuring unattended wizards



Before you can use unattended wizards, you need to configure them. To configure an unattended wizard:

- ▶ In the **Project Explorer**, select the project on which you would like the wizard to operate.
- ▶ Alternative 1: In the menu bar, **Project** menu, select the **Unattended Wizards** item.

A submenu opens up.

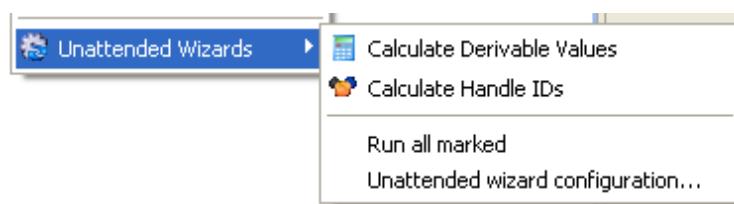


Figure 6.121. The **Unattended Wizards** submenu

- ▶ Alternative 2: In the tool bar, click the **View** button next to the **Run all unattended wizards marked with a '*' button**.

A menu opens up.

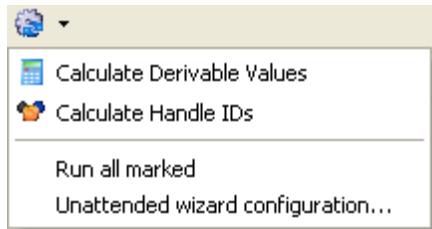


Figure 6.122. The **Unattended Wizards** menu

- ▶ Select **Unattended wizard configuration....**

The **Unattended Wizards** configuration dialog opens up.

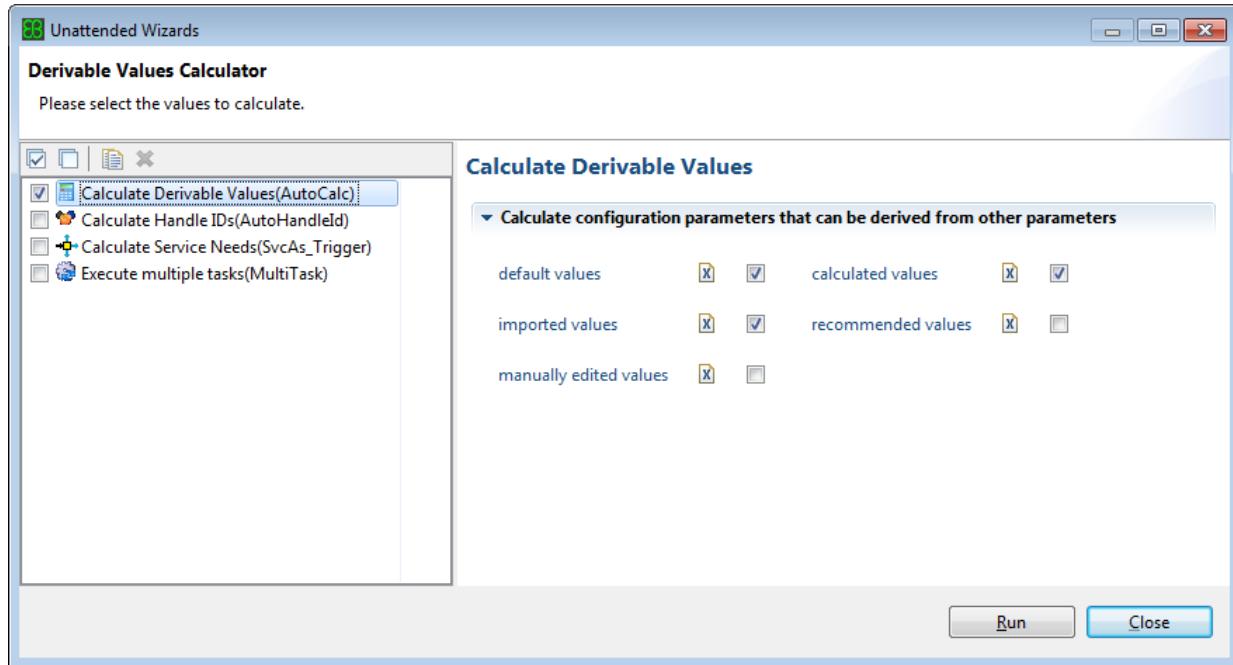


Figure 6.123. Configuring unattended wizards with the **Unattended Wizards** configuration dialog

- ▶ On the left side, the dialog lists all available unattended wizards. Select the wizard you want to configure from this list. The right part of the dialog window now displays the user interface of the wizard. The appearance of the wizard depends on the wizard you have selected. If the wizard is not configured correctly, warnings or error messages are displayed at the top of the window.
- ▶ To close the **Unattended Wizards** configuration dialog, click **Close**. All changes you made in the configuration will be saved when closing the **Unattended Wizards** configuration dialog.
- ▶ To run a wizard, see [Section 6.8.10.3, “Running unattended wizards”](#).

6.8.10.2. Managing multiple instances of unattended wizards

By default the **Unattended Wizards** configuration dialog shows all wizards registered via extension point exactly once in the list on the left side.

You can duplicate each available wizard configuration and with this create a new instance of this wizard. Therefore you can configure one task (e.g. Calculate Handle IDs) for several use cases.

To create a new instance of a special wizard, just select the existing wizard configuration and click the **Copy the selected unattended wizard configuration** button

Then a dialog opens where you can choose a name for the new configuration. A name proposal is already given. It consists of the trigger ID of the wizard registration followed by an underscore and a zero-base consecutive number.

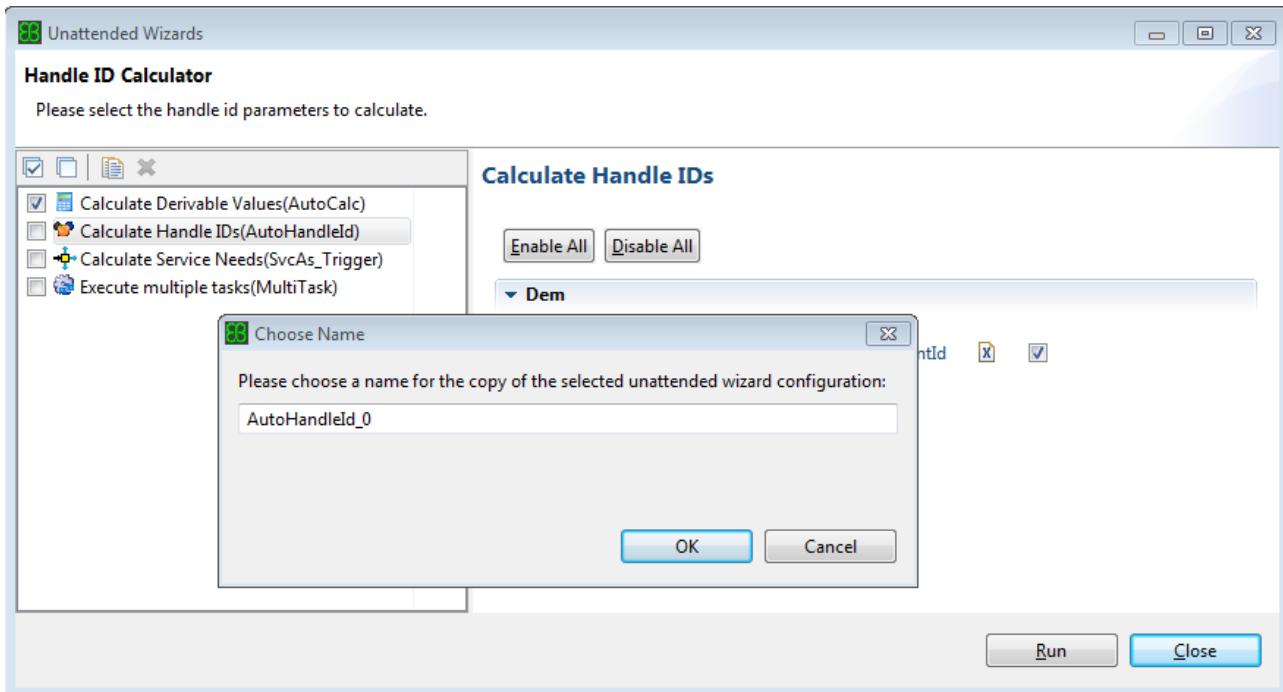


Figure 6.124. Choose a name for the new wizard configuration

After you clicked **OK**, the new wizard is selected in the list and the details area on the right side shows the configuration.

If you do not need this wizard configuration anymore, you can also remove it by clicking the **Delete the selected unattended wizard configuration** button .



NOTE**Removing wizard configurations**

You can only remove those wizard configurations you have created on your own. You cannot delete common wizard configurations that are registered via extension point.

NOTE**Wizard configuration IDs**

The list on the left side of the **Unattended Wizards** configuration dialog shows the label of the wizard and in parentheses the configuration ID. You can use this ID to run an unattended wizard from a workflow command and also from the command line.

NOTE**Reusing unattended wizard instances across projects**

Starting with release 24.0 EB tresos Studio saves the additionally created unattended wizard configurations in separate preference xdm files. These Files are located in the .prefs folder named <wizardConfigName>_pref_wizard.xdm.

You can share unattended wizard instances between projects by copying the individual preference file and the memento file <wizardConfigName>.mem that contains the configuration of the wizard to another project. This might be helpful when sharing workflows between projects.

6.8.10.3. Running unattended wizards

You may either run one single wizard or several wizards in a row. In both cases, the output is displayed in the **Results** view after the wizard has finished the work.

- ▶ If you want to know how to run one single unattended wizard, go to [Section 6.8.10.3.1, “Running a single unattended wizard”](#).
- ▶ If you want to know how to run several unattended wizards in a row, go to [Section 6.8.10.3.2, “Running multiple unattended wizards”](#).
- ▶ If you want to find information on the **Results** view that opens up after an unattended wizard has finished running, go to [Section 6.8.10.3.3, “Result feedback”](#).
- ▶ Some unattended wizards can be run from the command line. You find instructions for launching unattended wizards from the command line in [Section 6.12.2.10, “Executing unattended wizards on projects”](#).

To read additional information such as command line options or prerequisites, consult the documentation of the unattended wizard you want to run.

6.8.10.3.1. Running a single unattended wizard

There are two ways to run a single unattended wizard:



Alternative 1:

- ▶ In the **Project Explorer**, select the project on which you would like the wizard to operate.
- ▶ In the tool bar, click the **View** button next to the **Run all unattended wizards marked with a '*' button** .

A menu opens up.



Figure 6.125. **Unattended Wizards** menu

- ▶ In the menu, select the wizard you want to run.

The wizard runs automatically.

- ▶ When the wizard has finished, the **Results** view opens up. To read about the **Results** view, proceed to [Section 6.8.10.3.3, "Result feedback"](#).

Alternative 2 provides an opportunity for you to configure the unattended wizard before running it:

- ▶ In the **Project Explorer**, select the project on which you would like the wizard to operate.
 - ▶ In the tool bar, click the **View** button next to the **Run all unattended wizards marked with a '*' button** .
- A submenu opens up.
- ▶ In the submenu, select **Unattended wizard configuration....**

The **Unattended Wizards** configuration dialog opens up.

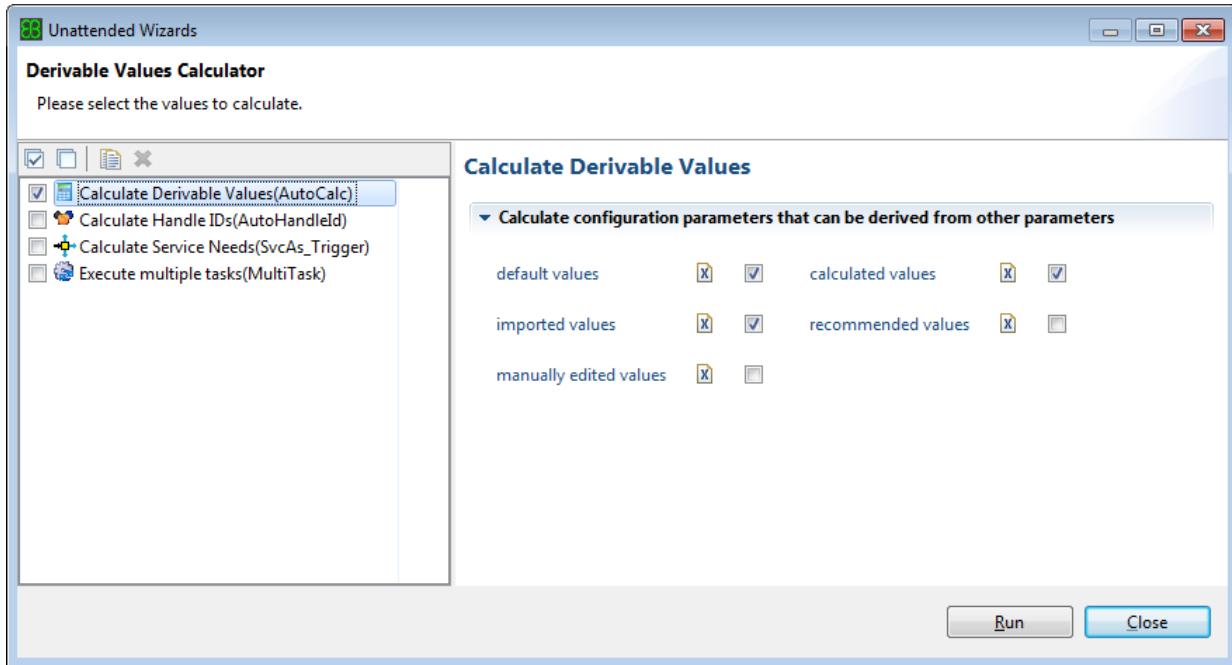


Figure 6.126. The **Unattended Wizards** configuration dialog

- ▶ In the dialog, highlight the wizard you want to run by selecting it from the list on the left side.
- ▶ The configuration options for the wizard are displayed on the right side of the dialog. You may change the configuration before running the unattended wizard.
- ▶ Click the **Run** button to run the wizard.

After the wizard has finished, the **Results** view opens up. To read more about the **Results** view, proceed to [Section 6.8.10.3.3, “Result feedback”](#).

6.8.10.3.2. Running multiple unattended wizards

You may define a set of unattended wizards which then can be run one after another with only one command. This is also called *batched execution*.

NOTE

Configure the unattended wizards before running them

You can only select the check box of unattended wizards you have configured before.



To define a set of wizards:

1. Open the **Unattended Wizards** configuration dialog.
 - ▶ In the **Project Explorer**, select the project on which you would like the wizard to operate.



- ▶ In the tool bar, click the **View** button next to the **Run all unattended wizards marked with a '*' button**



A submenu opens up.

- ▶ In the submenu, select **Unattended wizard configuration....**

The **Unattended Wizards** configuration dialog opens up.

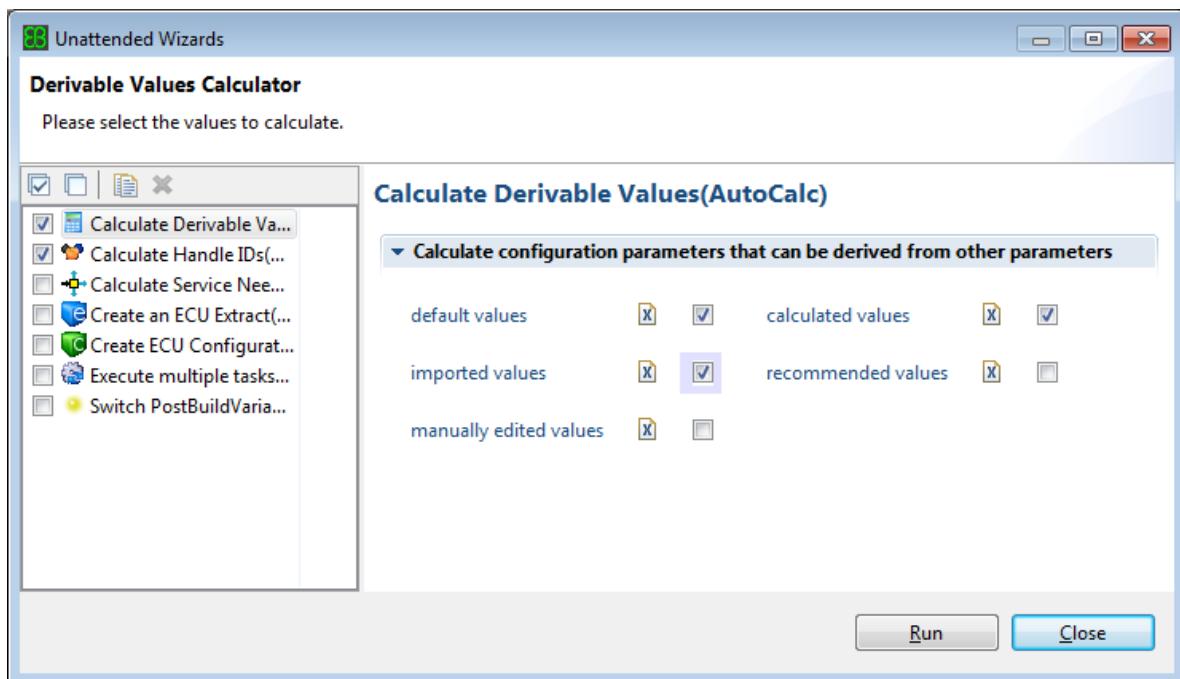


Figure 6.127. Defining the set of unattended wizards to run in a row

2. On the left side of the dialog you see a list of unattended wizards. For each wizard you want to include in the batched execution, select the corresponding check box.
3. Close the **Unattended Wizards** configuration dialog by clicking on the **Close** button. The state of the check boxes is saved automatically when you close the **Unattended Wizards** configuration dialog.

There are two ways to run the wizards you have just checked in the dialog.

Alternative 1:

- ▶ Make sure the configuration project on which you want the wizards to run is still selected in the **Project Explorer**.
 - ▶ In the tool bar, click the **View** button next to the **Run all unattended wizards marked with a '*' button**
- A menu opens up.
- ▶ In the menu, select **Run all marked**



All selected wizards are run.

Alternative 2:

- ▶ Make sure the configuration project on which you want the wizards to run is still selected in the **Project Explorer**.
- ▶ In the tool bar, click the **Run all unattended wizards marked with a '*' button**

All wizards that you have checked earlier are run.

After the wizards have finished, the **Results** view opens up. For information on the **Results** view, proceed to [Section 6.8.10.3.3, “Result feedback”](#).

6.8.10.3.3. Result feedback

After one or more unattended wizards have completed execution, the **Results** view opens up.

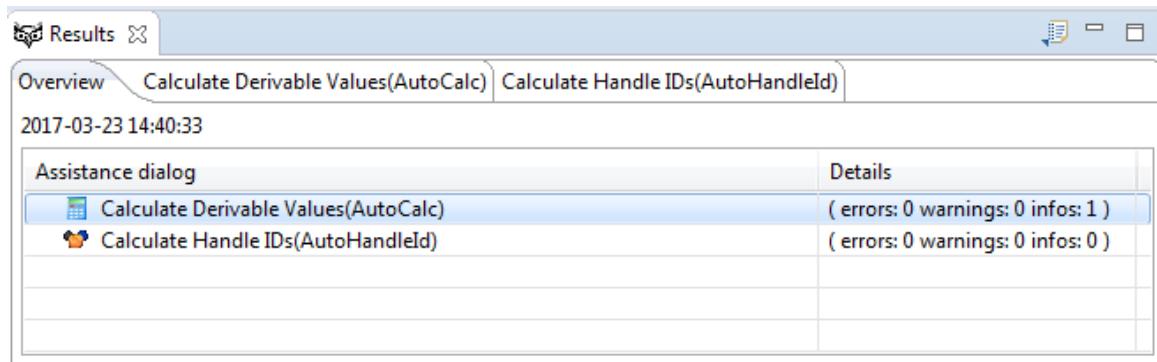


Figure 6.128. The **Results** view

The **Results** view displays the result of each wizard run. The view consists of several tabs. The exact amount of tabs and their content depends on the wizards that have been executed.

The leftmost tab always is the **Overview**. This tab displays a summary of the results of each wizard. In the image above, you see the results of two wizards, the **Calculate Derivable Values** and the **Calculate Handle IDs**. The **Calculate Derivable Values** has one *info* message as output. To view this message, click on the **Calculate Derivable Values** tab.

Every wizard that was executed has a result page in the view. To open the result page, click on the tab with the name of the wizard.

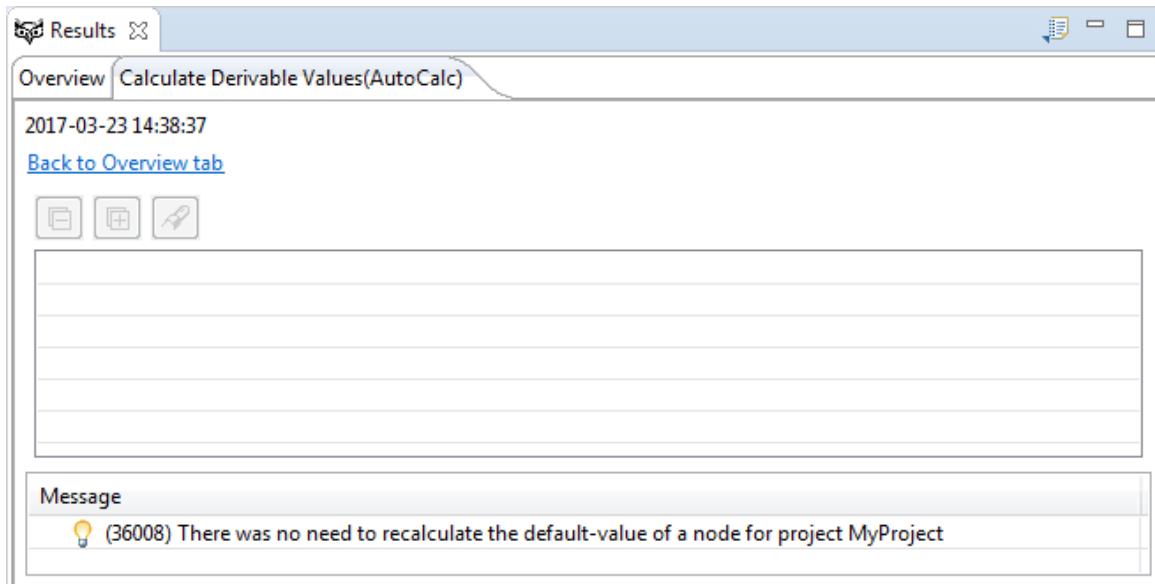


Figure 6.129. The **Results** view displays the results of an unattended wizard

The output of the unattended wizards is displayed as a list at the bottom of the window, which is labeled **Message**. This list contains all errors, warnings and info messages that occurred while the unattended wizards were running. Some unattended wizards display more detailed information as GUI elements in the middle of the **Results** view.

In the image above, the wizard does not display additional GUI elements, but you can see an info message at the bottom of the window.

To export the results of an unattended wizard:

- ▶ Click on the export button .
- A dialog opens up, in which you may enter a filename and directory to save the results.
- ▶ Browse to the directory of your choice and enter a name for the exported file.
- ▶ Click on **Save**

The results of the unattended wizard are being exported.

6.8.10.4. Configuring the Calculate Derivable Values wizard

The **Calculate Derivable Values** recalculates the value of nodes which must have a default value. If you choose this wizard, the **Error Log** view displays a list of all values that have been recalculated or changed.

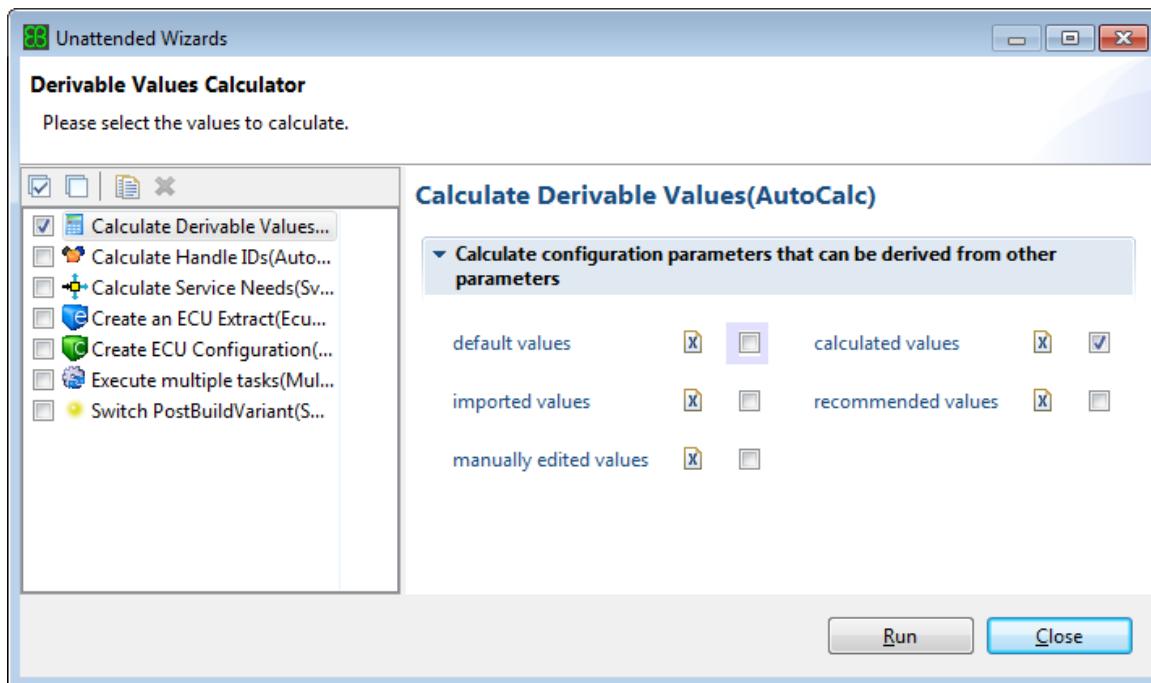


Figure 6.130. The Calculate Derivable Values wizard

In the **Calculate Derivable Values**, you may select which values you would like this wizard to recalculate:

Entry	Instructions
default values	To recalculate all values for which the module provider defined a default value calculation formula, select default values .
calculated values	To recalculate all values which have been calculated before, select calculated values . Calculated values are e.g. the values calculated by another unattended wizard, such as the Calculate Handle IDs .
imported values	To recalculate all values that have been overwritten by an import, select imported values .
recommended values	To recalculate all recommended values, select recommended values .
manually edited values	To recalculate all values you have been changed before, select manually edited values .

Table 6.1. Calculate values that contain

6.8.10.5. Configuring the Calculate Handle IDs wizard

The **Calculate Handle IDs** calculates the handle IDs for the modules of the communication stack.

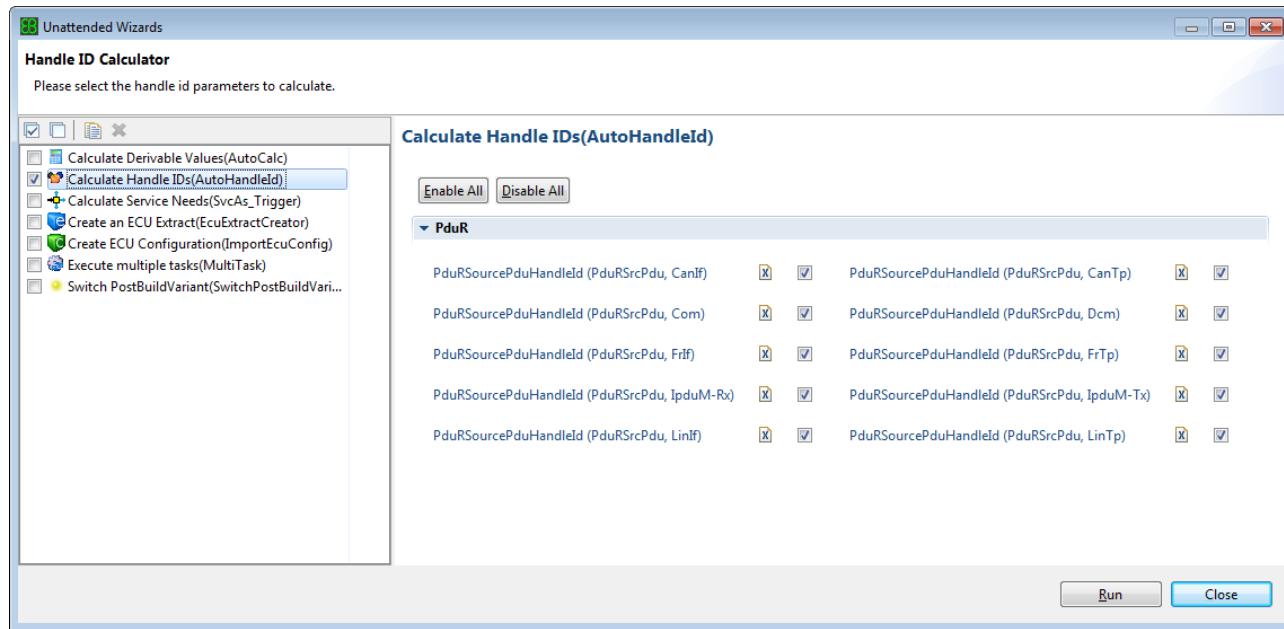


Figure 6.131. The Calculate Handle IDs wizard

The appearance of this wizard depends on the modules available in your project. There is a check box available for every handle ID parameter. If you check a check box, the handle IDs will be calculated for the corresponding parameter when the wizard is run.

6.8.10.6. Configuring the Execute multiple tasks wizard

The **Execute multiple tasks** wizard performs a set of actions in a specified order.

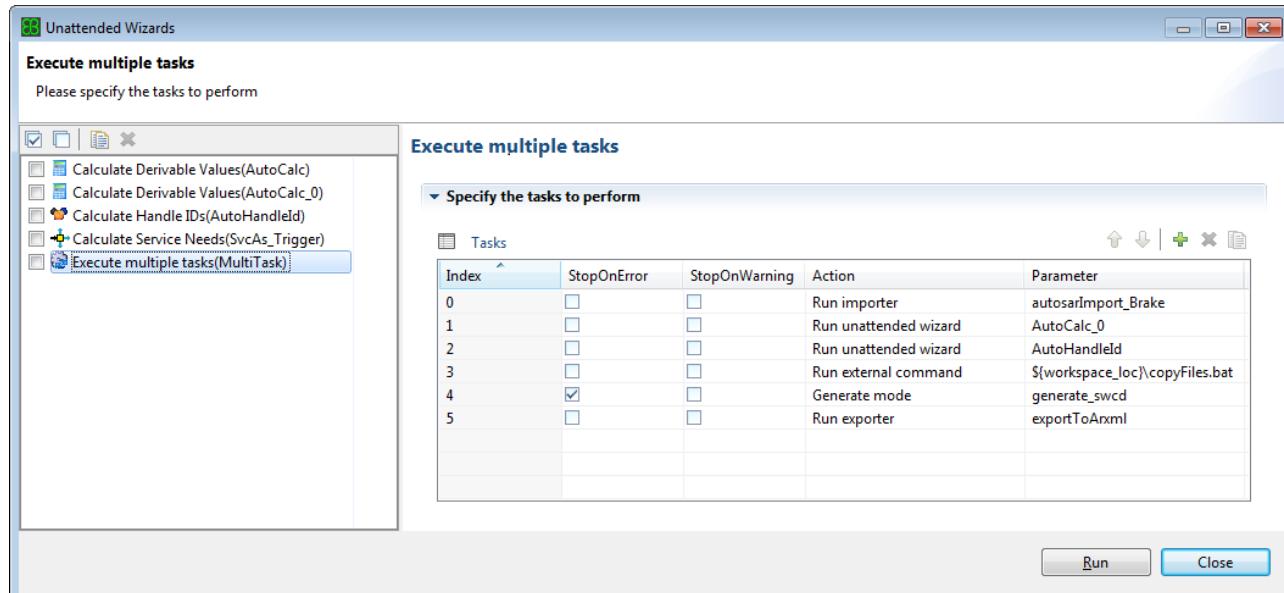


Figure 6.132. The Execute multiple tasks wizard



Available actions are the following:

- ▶ Run importer
- ▶ Run exporter
- ▶ Run unattended wizard
- ▶ Run code generation
- ▶ Run an external command

NOTE**Action parameters**

You can only select parameters for actions which are available for the current project, except for running an external command. For example, you can only select the importer name for an importer which is created in the current project.

For the action **Run external command** you can insert a command to, e.g. execute a batch file. If you press **CTRL+Space**, a content assist opens where you can choose variables, e.g. the workspace location `(${workspace_loc})` to prevent using absolute paths.

With the settings **StopOnError** and **StopOnWarning** it is possible to control execution of the wizard. For example if the importer returns with errors and warnings and further execution does not make sense in this case, just select **StopOnError** and **StopOnWarning** and execution will stop.

The result of the wizard execution is shown in the **Results** view.

6.8.10.7. Configuring the Switch PostBuildVariant wizard

With the Switch PostBuildVariant wizard it is possible to change the selected Selectable variant. This might be useful when using a MultiTask wizard to perform several steps for different Selectable variants. For this you may configure several instances of the unattended wizard **Switch PostBuildVariant**, one for each Selectable variant.

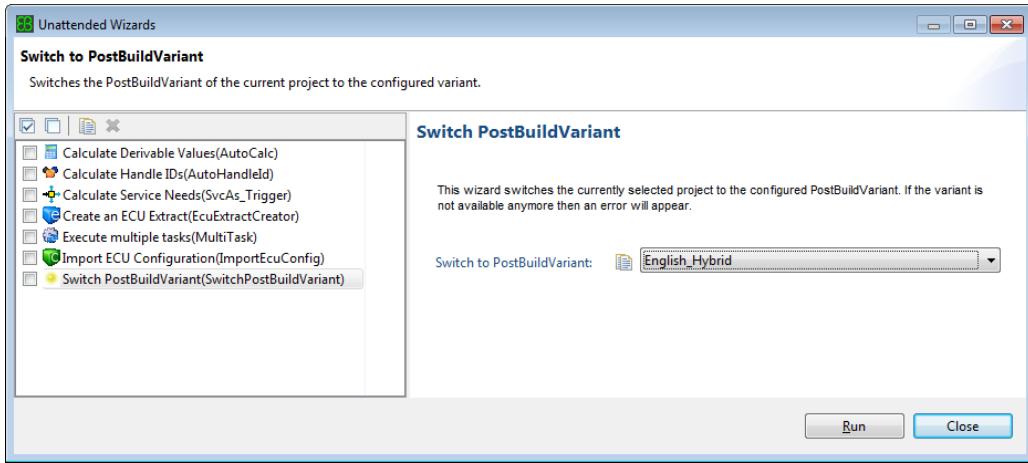


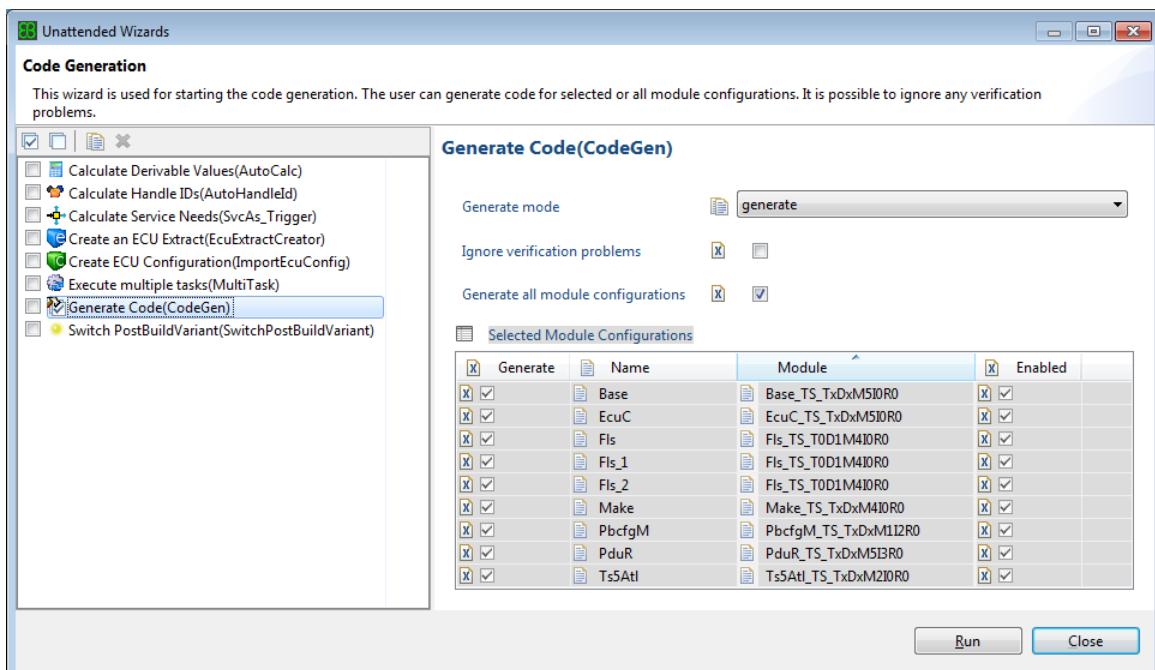
Figure 6.133. The unattended wizard to switch to a specified Selectable variant

NOTE**Unattended wizard only supports to switch Selectable variants**

The new unattended wizard **Switch PostBuildVariant** only supports to switch to a configured Selectable variant. Switching the selected Loadable variant is only possible via the ECUC module.

6.8.10.8. Configuring the Generate Code wizard

The **Generate Code** wizard provides an alternative way to start the code generation.

Figure 6.134. The **Generate Code** wizard



The wizard can be configured with the following options:

Option	Description	
Generate mode	Select the generator mode to run. The available modes depend on the modules which are configured in the project.	
	NOTE 	Applicable generation modes The modes generate and verify are applicable to any code generator. Other modes are only be applicable for the code generators, which support that mode.
Ignore verification problems	Check this option to deliberately start the code generation, even if verification problems exist in the ECU configuration data model.	
Generate all module configurations	If this option is checked, all module configurations will be considered, which are enabled for code generation in the project. See Section 6.6, "Editing module configurations" for how to enable or disable the code generation for module configurations. Uncheck this option to enable the Selected Module Configurations table.	
Selected Module Configurations	Uncheck the option Generate all module configurations and select the module configurations for which code shall be generated within the Generate column of the Selected Module Configurations table.	

Table 6.2. **Generate Code** wizard configuration

6.8.11. Searching for configuration parameters

EB tresos Studio provides the option of searching for configuration parameter names and their values within files or within the data model.

There are two alternative ways to start the search.

Alternative 1:

- ▶ Press **Ctrl+T**
- ▶ The **Search** dialog opens up and the **ECU Configuration Search** tab is selected.

Alternative 2:

- ▶ Select the **Search** menu from the main menu.



A submenu opens up.

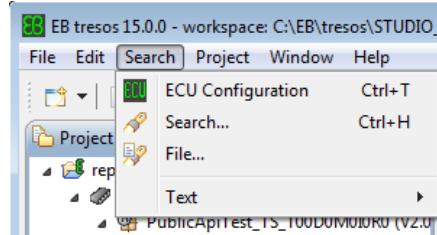


Figure 6.135. The Search submenu

- ▶ Select the **ECU Configuration** submenu.
- ▶ The **Search** dialog opens up and the **ECU Configuration Search** tab is selected.

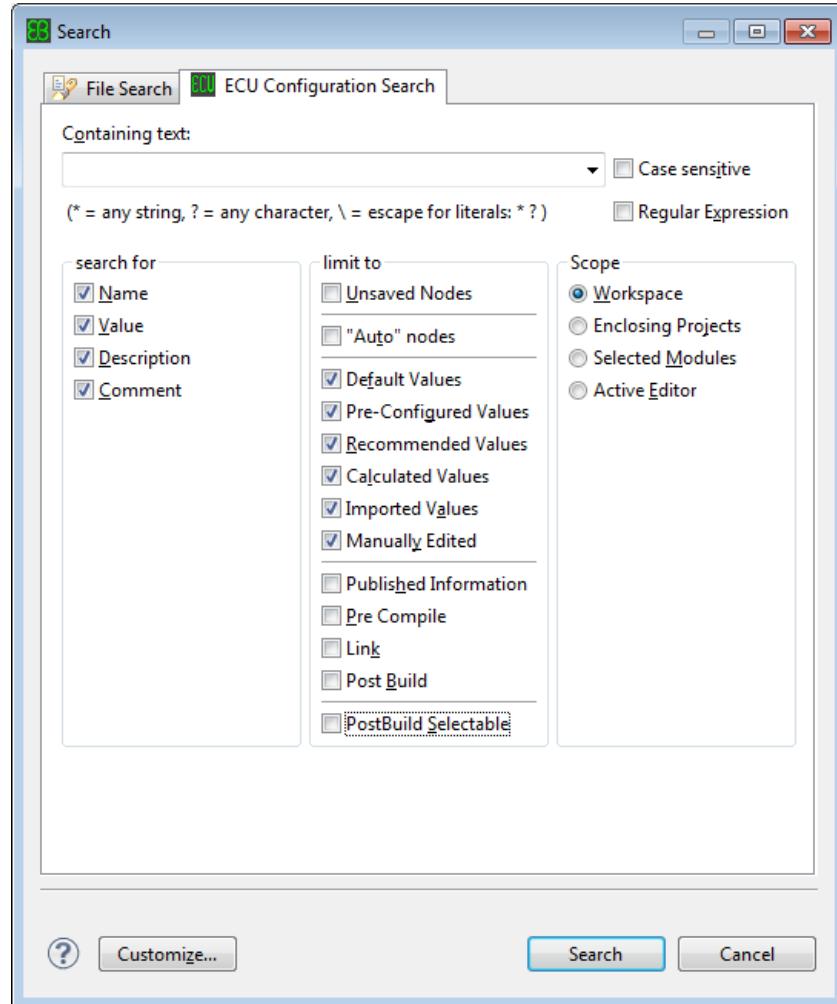


Figure 6.136. The **Search** dialog

**NOTE****Other tabs within the Search dialog are not used within EB tresos Studio**

The **Search** dialog also serves other purposes than to search within the data model of a project. Thus, it provides several other tabs. These tabs are contributed by the Eclipse framework and have no use in EB tresos Studio. Please consult the Eclipse documentation for more information.

6.8.11.1. Setting search options

To find configuration parameter names and their values within the data model, select the tab **ECU Configuration Search**. As you open the tab, the last search executed is automatically restored.

6.8.11.1.1. Defining general search options

In the top part of the **Search** dialog, you may define general search options:

Containing text:

 Case sensitive

(* = any string, ? = any character, \ = escape for literals: * ?)
 Regular Expression

Figure 6.137. General search options

Field/button name	Instructions
Containing text	<ul style="list-style-type: none"> ▶ To restore an earlier search, select one of your last searches from the Containing text drop-down combo box. All options for the respective search are restored. ▶ To search for a new text string, select Containing text and type in your text. You may use the wild cards ? for characters and * for strings. ? and * can be escaped by using the \, e.g. *.
Case sensitive	To search for case sensitive text strings, select the Case sensitive check box. This option is not available for regular expressions.
Regular Expression	To have the searched text interpreted as regular expression, select the Regular Expression check box.



Field/button name	Instructions
	<p>NOTE</p>  <p>The Regular Expression button starts a content assistance function If you check the Regular Expression check box, a content assistance function will start.</p>

6.8.11.1.2. Defining which kind of text is searched

In the bottom left part of the **Search** dialog, you may define the **search for** group of search options:

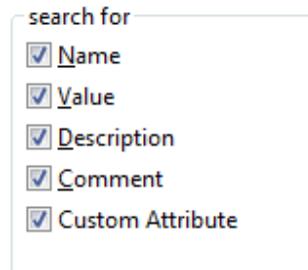
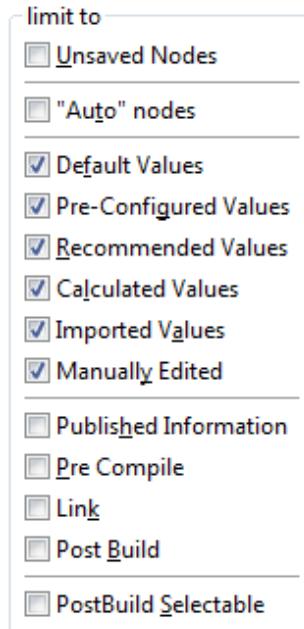


Figure 6.138. The **search for** group of search options

Field/button name	Instructions: Select the check box
Name	To search for the name of nodes.
Value	To search for the value of nodes.
Description	To search for node descriptions, i.e. the text you see in the Element Description view on the bottom right corner of the main window.
Comment	To search for values of parameter comments, i.e. in summary or description. It is also possible to search for parameter comments in a specific status or changed by a certain user.
Custom Attribute	To search for custom attributes, e.g. for specific Label values.

6.8.11.1.3. Limiting the kind of parameters that are searched

In the bottom middle part of the **Search** dialog, you may define the **limit to** group of search options:

Figure 6.139. The **limit to** group of search options

Field/button name	Instructions: Select the check box
Unsaved Nodes	To restrict your search to unsaved nodes, i.e. those which have not yet been saved after an import.
"Auto" Nodes	To restrict your search to nodes where the generator calculates the value during code-generation.
Default Values	To display all elements that are set to their default value.
Pre-Configured Values	To display all elements that are set to preconfigured values.
Recommended Values	To display all elements that are set to recommended values.
Calculated Values	To display all elements that are calculated.
Imported Values	To display all elements that are imported.
Manually Edited	To display all elements that are manually edited.
Published Information	To display all elements that have config class <i>Published Information</i> .
Pre Compile	To display all elements that have config class <i>Pre Compile</i> .
Link	To display all elements that have config class <i>Link</i> .
Post Build	To display all elements that have config class <i>Post Build</i> .
PostBuild Selectable	To display all elements that are subject to post-build variant handling.



6.8.11.1.4. Defining the scope of the search

In the bottom right part of the **Search** dialog, you may define the **Scope** group of search options:

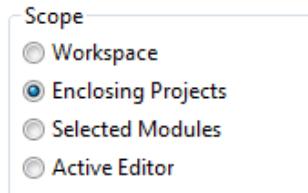


Figure 6.140. The **Scope** group of search options

Field/button name	Instructions: Select the option button
Workspace	To search in all open projects.
Enclosing Projects	To search in all projects whose tree contains a selection within the Project Explorer view. <div style="display: flex; align-items: center;"> NOTE Projects are not included automatically The project in the currently opened editor is not automatically included. </div>
Selected Modules	To search for all modules whose parent- or sub-tree contains a selection within the Project Explorer .
Active Editor	To search within the module of the currently active editor.

Proceed to [Section 6.8.11.2, “Displaying search results”](#) to read information about starting the search.

6.8.11.2. Displaying search results

To display the search results, open the **ECU Configuration Search** in the **Search** dialog and select the search options of your choice.

- ▶ To start the search, select the **Search** button in the **Search** dialog.

The **Search** view opens up on the bottom of the GUI next to the **Error Log** and the **Problems** view. The **Search** view contains all the results of the search, i.e. the configuration nodes that contain the searched text.

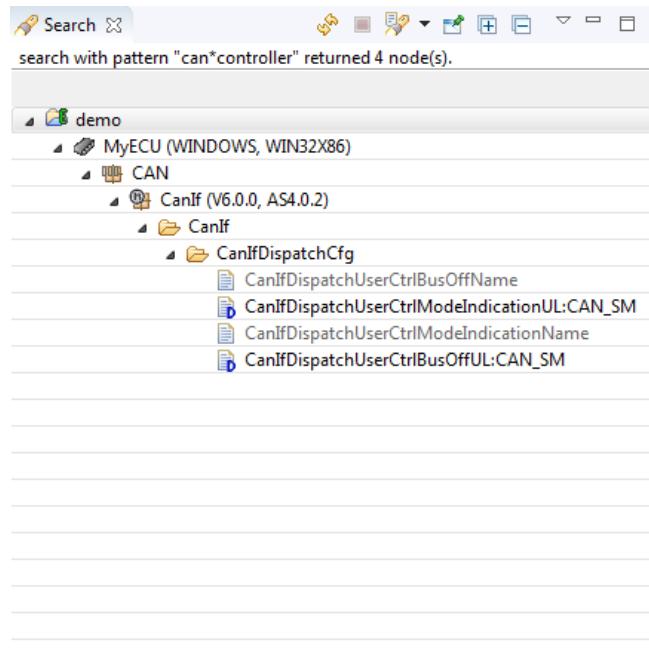


Figure 6.141. The Search view

The view displays a node tree for each project for which a configuration node was found. The upper part of the tree is the same as shown within the **Project Explorer** view. The lower part of the tree is the same as shown within the **Outline** view. Using the + and - icons at the top of the view expands or collapses the trees.

To open the **Editor** view of a node:

- ▶ Right-click on the node.
A context menu opens up.
- ▶ Select **Show in Editor**.

The configuration node is displayed in the **Editor** view.

To open the **Editor** view of a leaf node:

- ▶ Double-click on the node.
A context menu opens up.
- ▶ Select **Show in Editor**.

The configuration node is selected and displayed in the **Editor** view.

To read about how to configure the parameters with the ECU Configuration Editor, go to [Section 5.4.1, “The Generic Configuration Editor”](#).



6.9. Generating code for projects

After you have finished configuring a project, you can generate the source code of it. The resulting code depends on the values you have entered in the editor. A module only generates if the creator of the module registered a code generator for it.

NOTE
Your configuration project must not contain errors


You can only generate code if the configuration project contains no errors.

6.9.1. Verifying a project

Some problems with the configured values of a project are only discovered when you generate the code of your project. Thus, before you actually generate the code, you may choose to verify the code first. During verification the same processing steps are executed as during code generation. But the difference is that no files are written to the disk.

The result of the verification is written to the **Error Log** view.

The verification is limited to parameters of the selected configuration time and to parameters you have configured before generating the project. This means:

- ▶ If your configuration time is `LINK` only, all `LINK` and `PRE-COMPIL` parameters are verified.
- ▶ If you have selected `POST-BUILD` in your configuration time settings, all parameters are verified.

For further information on configuration time and configuration classes, see [Section 3.5.2.1, “Configuration time of projects”](#).

NOTE
All variants will be verified


If you have configured multiple post-build selectable variants in your project, all variants will be verified, not just the currently active variant.

TIP
Do not block info messages during verification of code


When verifying code, make sure that info messages are not blocked in the **Error Log**. Otherwise you might miss feedback.

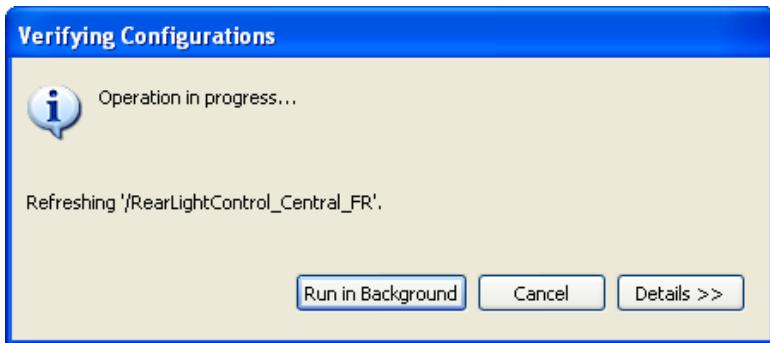
There are three alternatives to verify a project:



Alternative 1: To start the generators to verify a project:

- ▶ Select an ECU configuration node within the project from the **Project Explorer**.
- ▶ Right-click on the ECU configuration node.
- ▶ Select **Verify Project**.

A dialog window opens up and starts the verification process. The window closes automatically when verifying the project is finished.



Alternative 2: To verify a project:

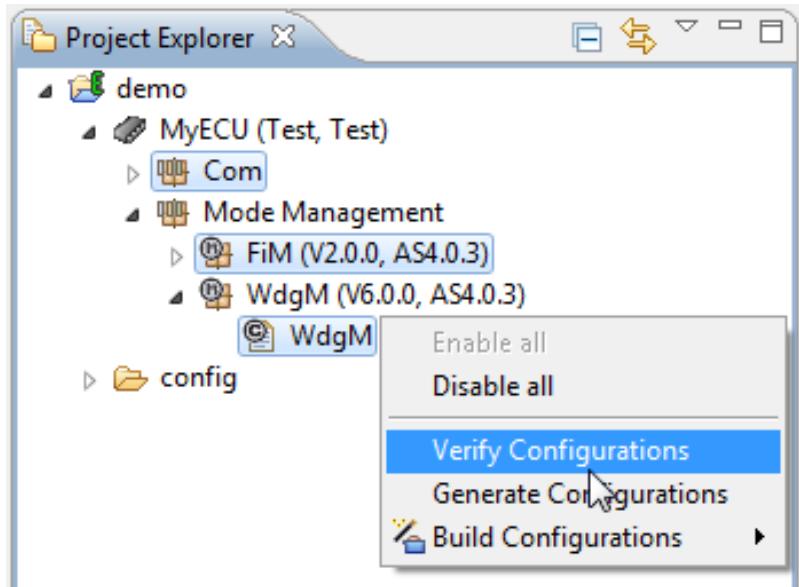
- ▶ Select an ECU configuration node within the project from the **Project Explorer**.
- ▶ Click the **Verify selected project** button  from the tool bar.

A dialog window opens up and the tool starts verifying the project. The window closes automatically when verifying the project is finished.

Alternative 3: To verify only selected module configurations of a project:

- ▶ Select one or several category/layer/type/module nodes that contain enabled module configurations within the **Project Explorer**.
- ▶ Right-click on one of the selected nodes.
- ▶ Select **Verify Configurations**.

A dialog window opens up and starts the verification process. The window closes automatically when the verification is finished.



6.9.2. Generating a project

If you generate code of your project, code is generated for all modules for which you have enabled code generation in the **Module Configurations** dialog (see [Section 6.6, “Editing module configurations”](#)).

The result of the generation is written to the **Error Log** view.



NOTE**Your configuration project must not contain errors**

You can only generate code if the configuration project contains no errors.

NOTE**The code generator depends on the configuration time**

The code generator checks whether the configuration time of your project fits to its own configuration time. This means that the configuration time of the project must be earlier or equal to the configuration time of the generator.

If a code generator does not define its own configuration time, disable the configuration time support in your project. Otherwise the code generator cannot start.

NOTE**All variants will be generated**

If you have configured multiple post-build selectable variants in your project, the code for all variants will be generated, not just for the currently active variant.

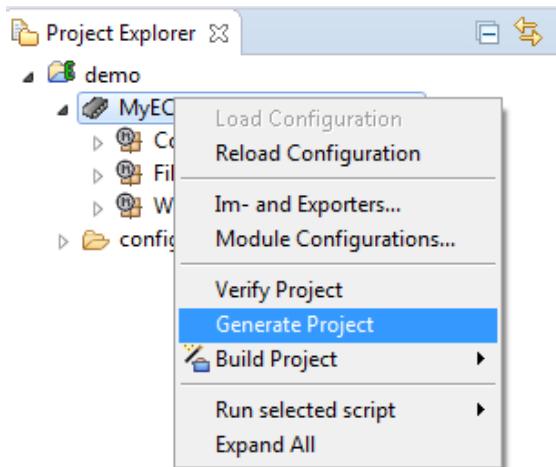
TIP**Do not block info messages during generation of code**

When you generate code, make sure that info messages are not blocked in the **Error Log**. Otherwise you might miss feedback. Refer to [Section 5.3.5.2, “Filtering Error Log messages by their severity level”](#) for information on messages in the **Error Log**.

There are three alternatives to generate code:

Alternative 1: To generate code:

- ▶ Select an ECU configuration node within the project.
- ▶ Right-click on the ECU configuration node.
- ▶ Select **Generate Project**.



A dialog window opens up and starts generating code.

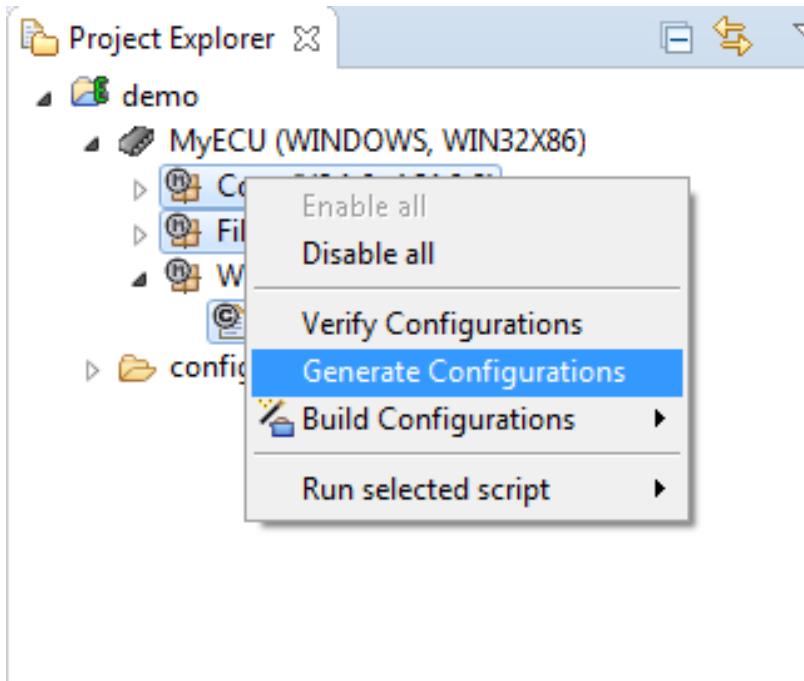
Alternative 2: Alternatively, to generate code:

- ▶ Click the **Generate** button from the tool bar.

A dialog window opens up and starts generating code.

Alternative 3: Alternatively, to generate code only for selected module configurations of a project:

- ▶ Select one or several category/layer/type/module nodes that contain enabled module configurations within the **Project Explorer**.
- ▶ Right-click on one of the selected nodes.
- ▶ Select **Generate Configurations**.



A dialog window opens up and starts generating code.

The generated code is written to the output folder set in the project preferences.

6.9.3. Building a project

To build a project means running the registered code generator engines in a specific mode which is different than the mode in which you generate code or verify the project. The available modes depend on the project you have currently selected and on the modules available in this project. The exact names of the generator modes and their outcome is controlled by the modules themselves. See the documentation of the modules present in your project.

The result of building the project is written to the **Error Log** view.

**NOTE****The Problems View view must not show errors**

You can only build your project if the **Problems View** view shows no errors.

NOTE**All variants will be built**

If you have configured multiple post-build selectable variants in your project, the code for all variants will be generated, not just for the currently active variant.

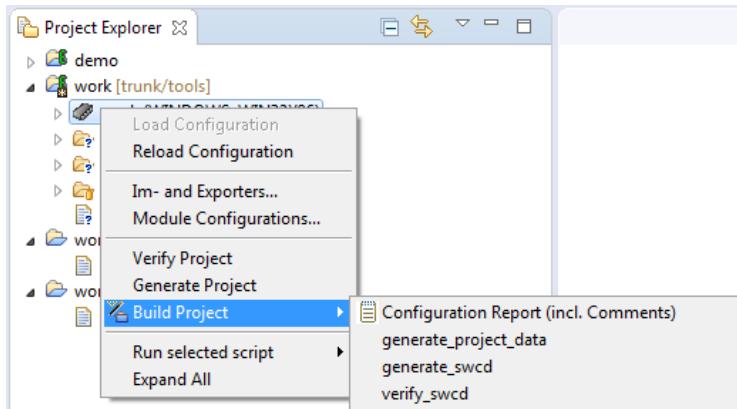
TIP**Do not block info messages when you build a project**

When you build a project, make sure that info messages are not blocked in the **Error Log**. Otherwise you might miss feedback. Refer to [Section 5.3.5.2, “Filtering Error Log messages by their severity level”](#) for information on messages in the **Error Log**.

There are three alternatives to build a project:

Alternative 1: To build code:

- ▶ Select an ECU configuration node within the project.
- ▶ Right-click on the ECU configuration node.
- ▶ Select **Build Project**.



A dialog window opens up and starts building code.

Alternative 2: Alternatively, to build code:

- ▶ Click the arrow next to the **Generate** button in the tool bar.

A context menu opens up. The menu items of this context menu depend on your project and on the modules available in this project.

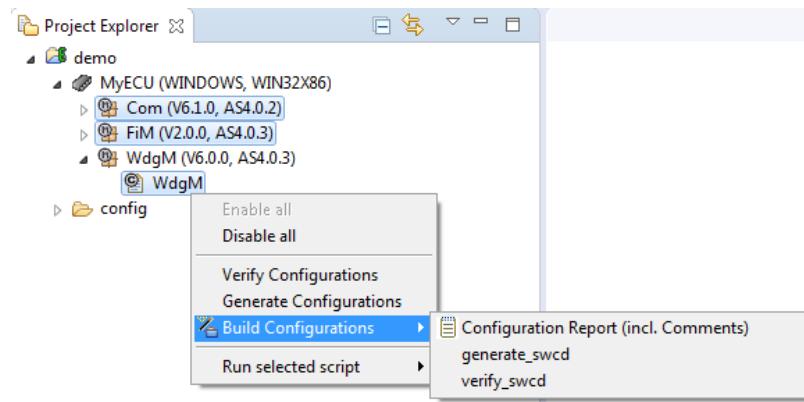


- ▶ Select the mode you wish to run.

A dialog window opens up and starts building code.

Alternative 3: Alternatively, to build code only for selected module configurations of a project:

- ▶ Select one or several category/layer/type/module nodes that contain enabled module configurations within the **Project Explorer**.
- ▶ Right-click on one of the selected nodes.
- ▶ Select **Build Configurations**.



A dialog window opens up and starts building code.

The generated code is written to the output folder set in the project preferences.

6.10. Importing and exporting

This section presents the different ways of importing and exporting projects and configuration data.

6.10.1. Exchanging a complete EB tresos Studio project with other installations

With EB tresos Studio you may import and export projects, e.g. to exchange a project with another EB tresos Studio installation, or to backup a workspace on external memory space. This feature is not the same as im-



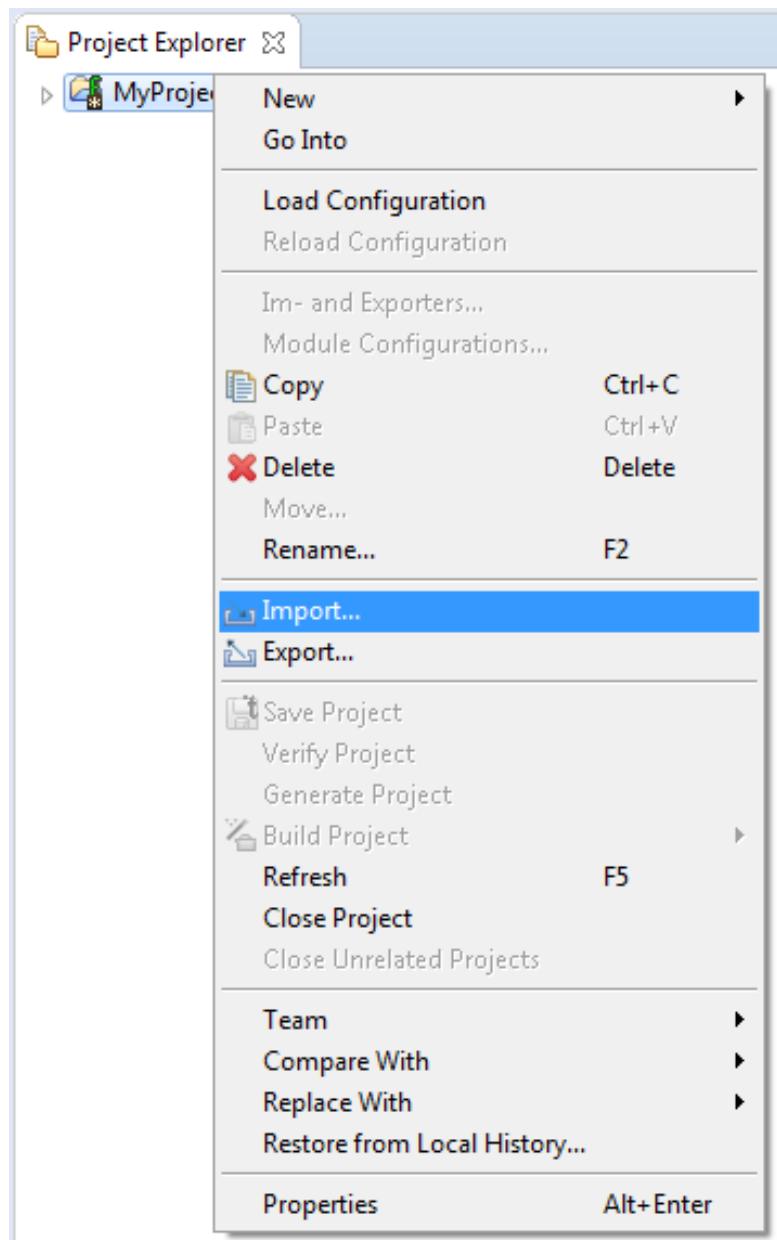
porting and exporting a configuration! To import or export a project configuration, proceed to [Section 6.10.2, "Importing and exporting configuration data"](#). The project import and export is done through the Eclipse framework of EB tresos Studio

6.10.1.1. Importing a project

To import a project from other locations to the currently active workspace:

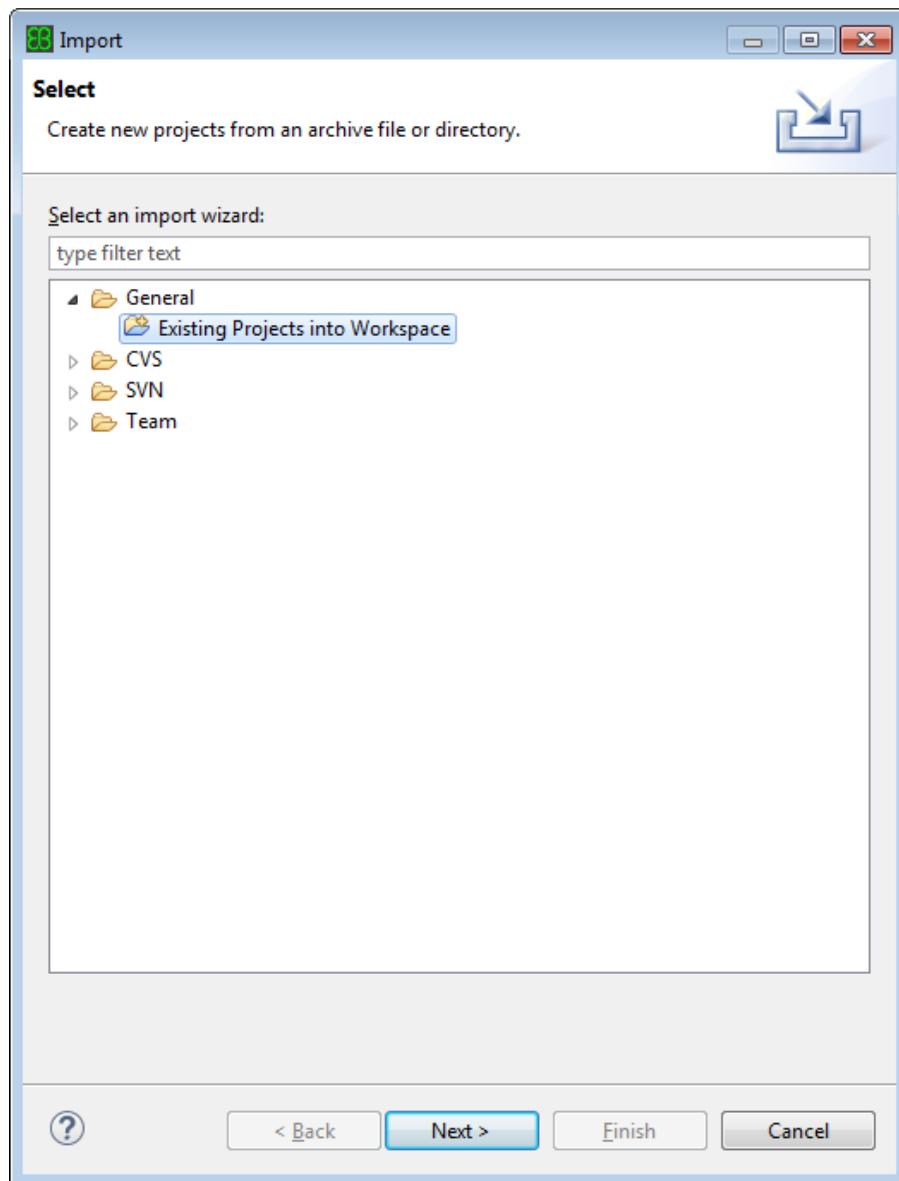
- ▶ Select the project node in the Project Explorer.
- ▶ Right-click on the node.

A context menu opens up.



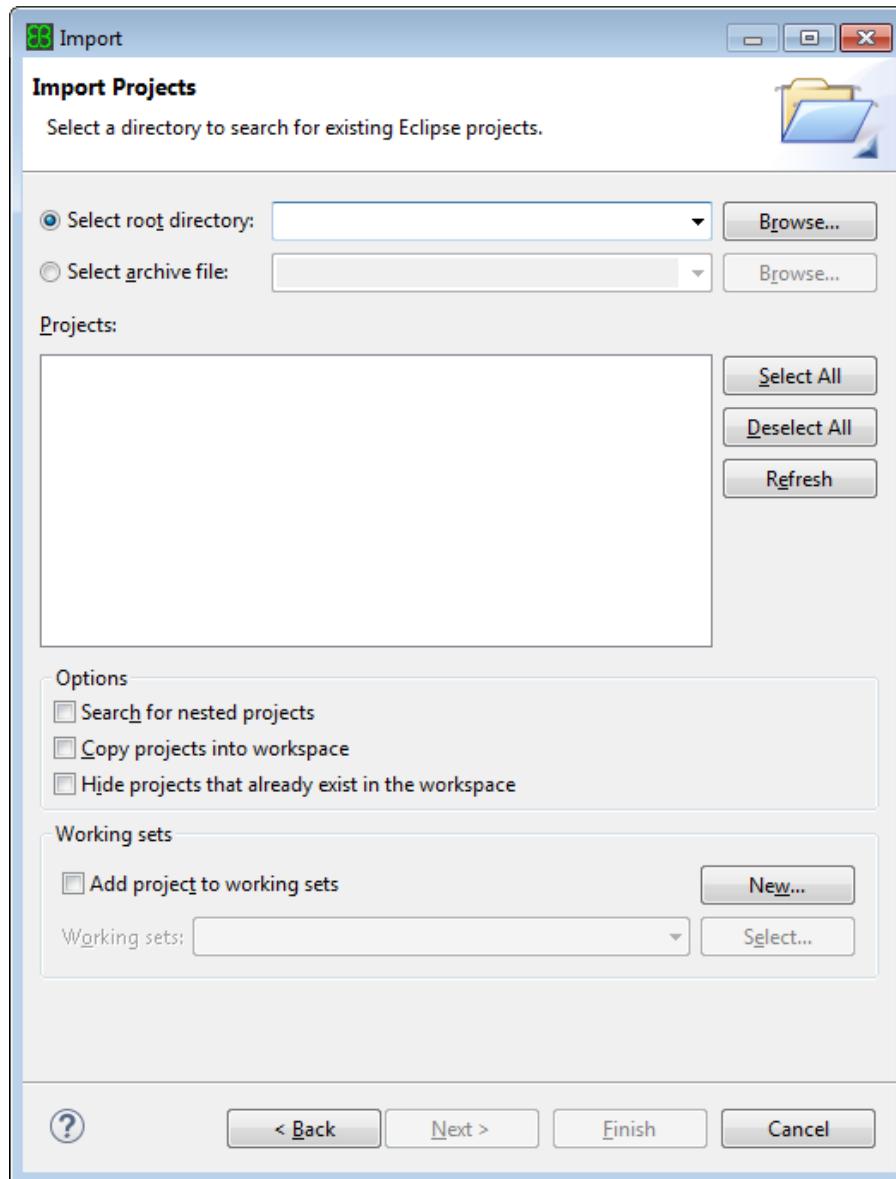
- Select **Import**.

The **Import** window opens up.



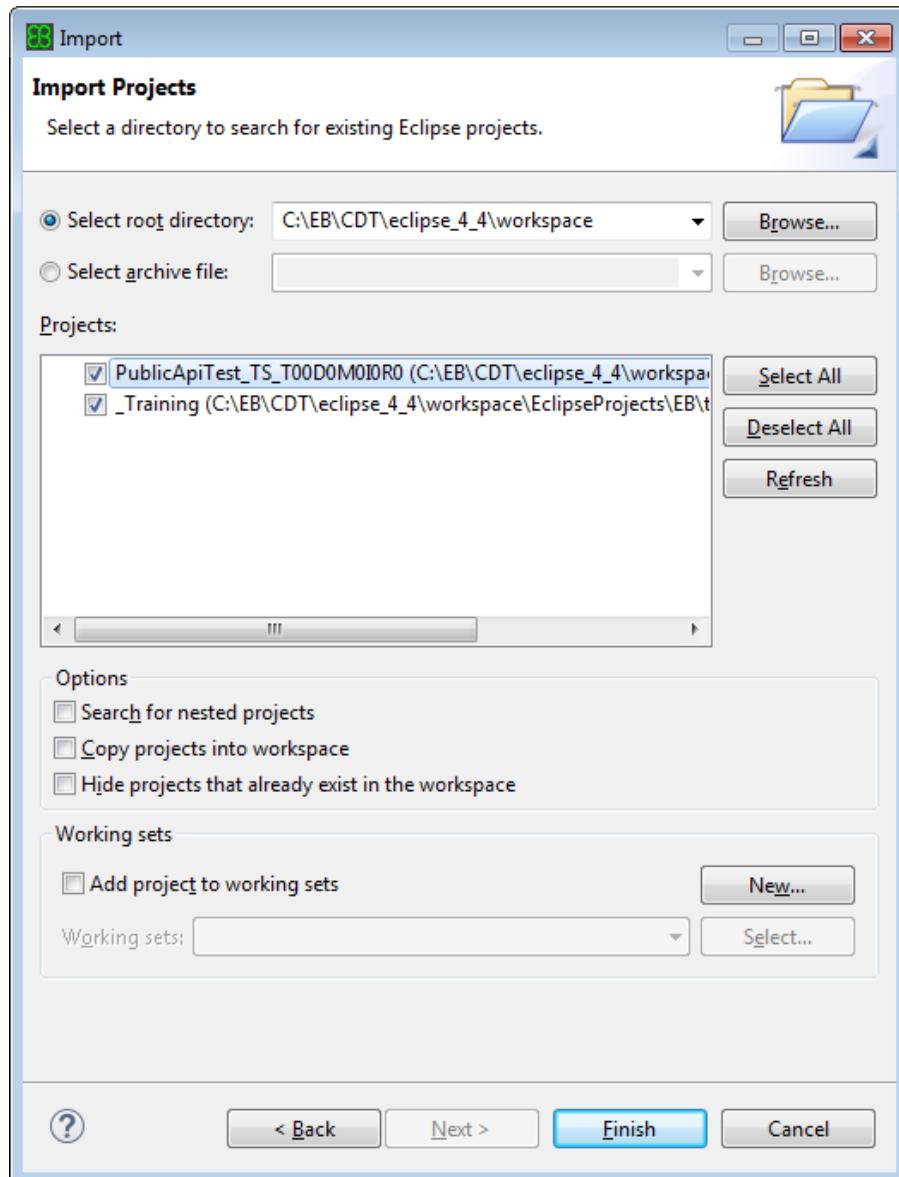
- ▶ Select **Existing Projects into Workspace**.
- ▶ Double-click **Existing Projects into Workspace**.

The **Import** window opens up.



- ▶ If you want to import a project that is located in the file system, click the option button **Select root directory** and enter the path to the project's directory in the text box. If you click the **Browse** button, you can choose an existing directory.
- ▶ If you want to import a project that is contained in an archive, click the option button **Select archive file** and enter the path and name of the archive file in which the project is contained into the text box. If you click the **Browse** button, you can choose an existing archive file.

All projects that can be imported from your source directory or archive are now listed in the **Projects** listbox. Note that a project is not listed if a project with the same name already exists in the currently active workspace.



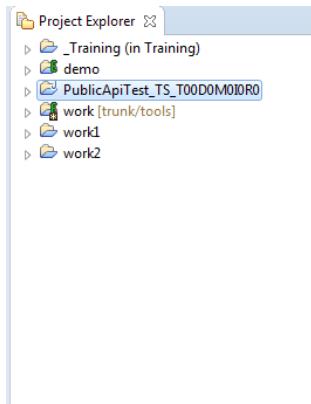
- ▶ Uncheck the projects you do not want to import from the list.
- ▶ To copy the project content into the currently active workspace, check the **Copy projects into workspace** check box. This option is only available if the import source is a directory and not an archive.

WARNING

If you do not check this box, EB tresos Studio generates a linked project. This means that you will later work on the files directly inside the folder from which you imported the project and not on a local copy inside your workspace.

- ▶ After you have chosen all projects that you want to import, click the **Finish** button to start the actual import.

The projects you imported are now listed as projects in the **Project Explorer**.

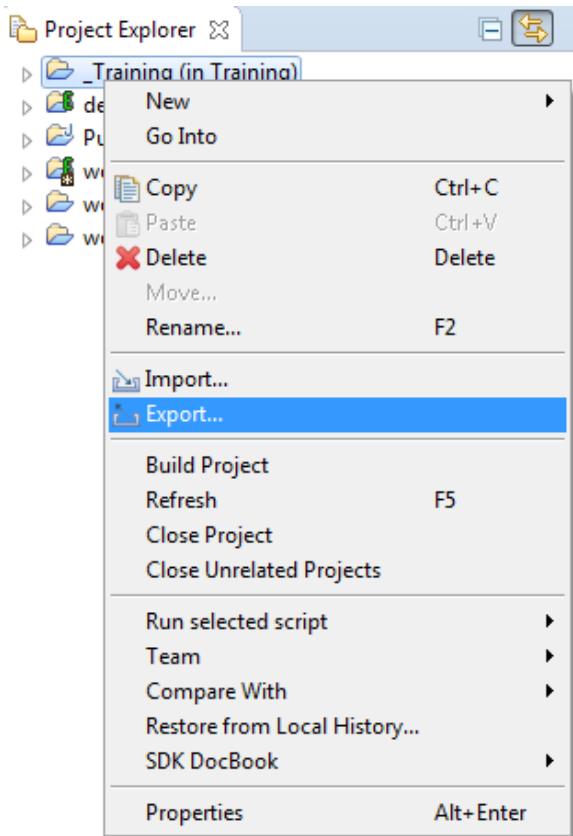


6.10.1.2. Exporting a project

If you want to share your project with other EB tresos Studio installations, or if you want to make a backup copy of a project, you can export the project. To export projects from the currently active workspace to other locations:

- ▶ Select the project node in the Project Explorer.
- ▶ Right-click on the node.

A context menu opens up.



- ▶ Select **Export**.

The **Export** window opens up.

Proceed with either of the following steps:

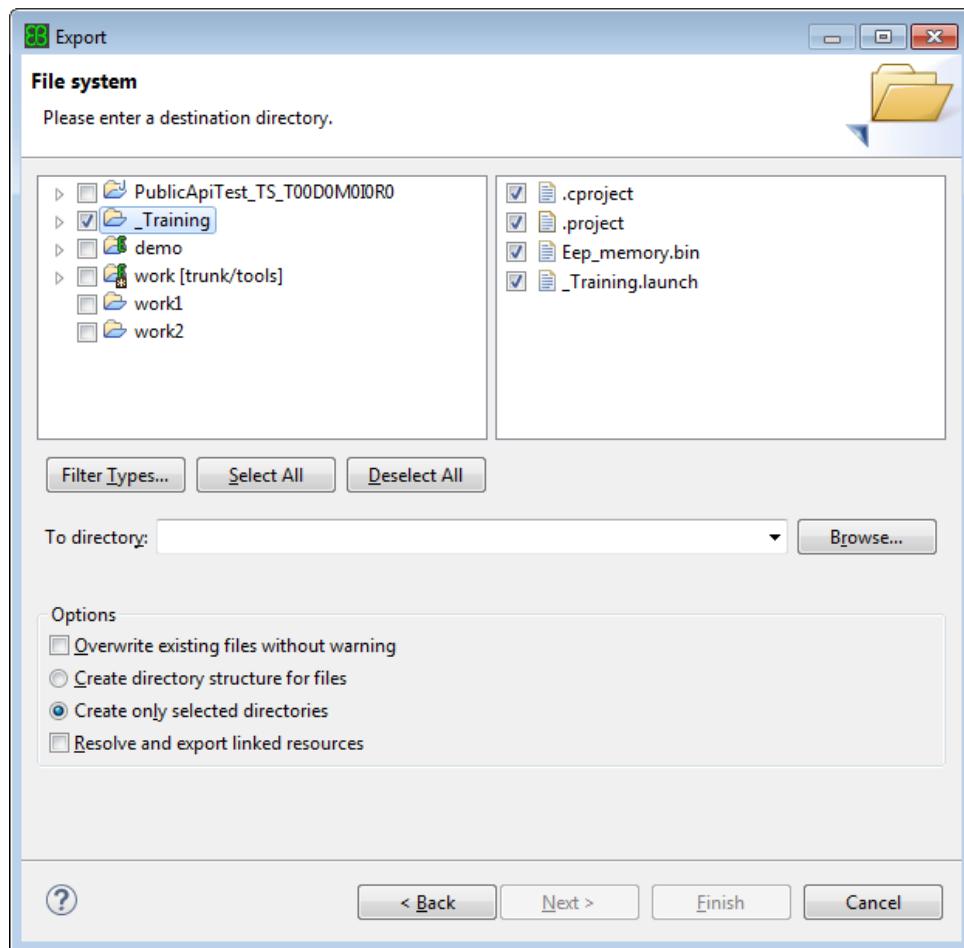
- ▶ [Section 6.10.1.2.1, "Exporting a project as copy"](#) exporting a project as copy,
- ▶ [Section 6.10.1.2.2, "Exporting a project as archive"](#) or exporting a project as archive.

6.10.1.2.1. Exporting a project as copy

To copy the EB tresos Studio workspace data to another location, export the project as directory structure.

- ▶ In the Select window, select **File System** as export destination.
- ▶ Click the **Next** button.

The **File system** page is shown.



- ▶ In the list on the left of the dialog you can see all projects of your current workspace. Check all the projects you would like to export. In case you want to exclude some parts of the project, you can also expand the project in the tree and check or uncheck the contained resources. This is not recommended, as partially exported projects are usually not fully functional.
- ▶ Enter the directory into which you want to export the projects into the drop-down text box **To directory**. You can also use the **Browse** button to choose a directory.
- ▶ In **Options**, select how you would like the project export to be performed.
- ▶ To export the projects, select **Finish**.

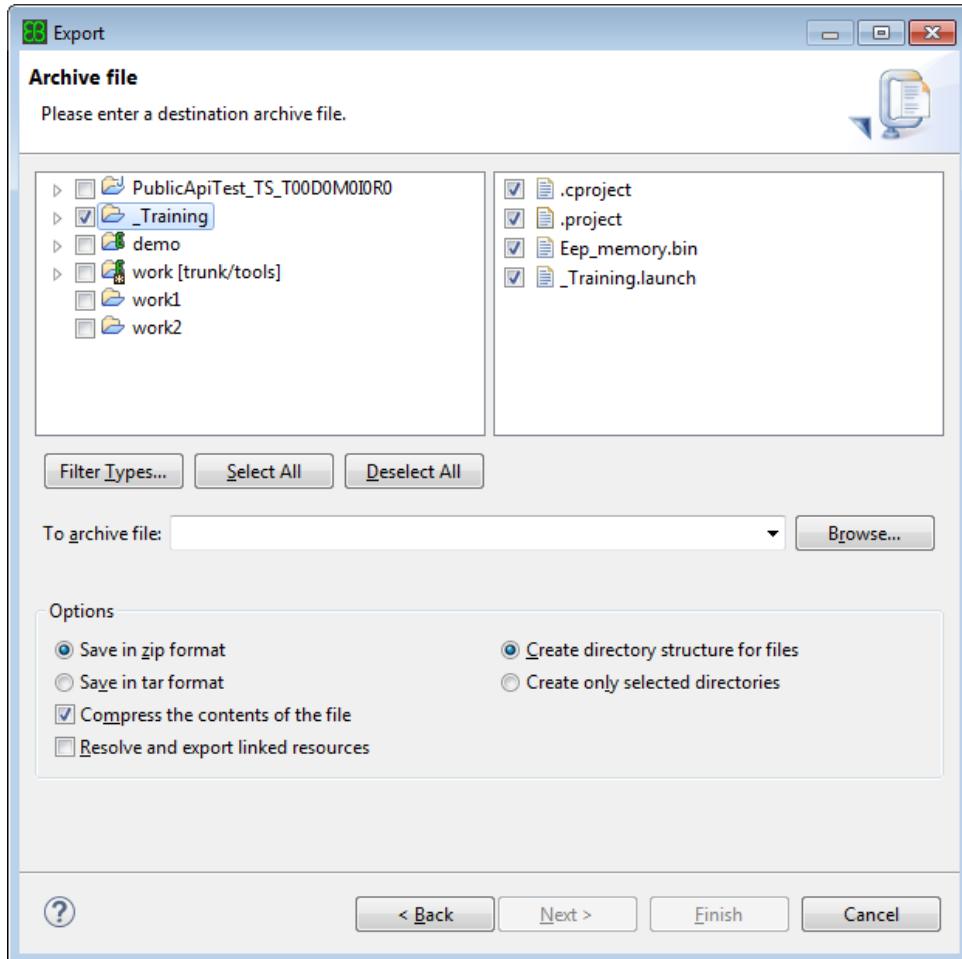
6.10.1.2.2. Exporting a project as archive

To store one ore more complete EB tresos Studio projects as compressed archive, export them as archive. This may be useful, e.g. if you want to exchange a project with another EB tresos Studio installation.

- ▶ In the **Select** window, select **Archive File** as export destination.
- ▶ Click the **Next** button.



The **Archive file** page is shown.



- ▶ In the list on the left of the dialog you can see all projects of your current workspace. Check all the projects you would like to export. In case you want to exclude some parts of the project, you can also expand the project in the tree and check or uncheck the contained resources. This is not recommended, as partially exported projects are usually not fully functional.
- ▶ Enter the filename of the archive into the drop-down text box **To archive file**. You can also use the **Browse** to choose a file.
- ▶ In **Options**, select how you would like the file system export to be performed.
- ▶ To export the projects, select **Finish**.

6.10.2. Importing and exporting configuration data

Importers and exporters are used to import and export from and to the configuration data of a project from file formats such as the AUTOSAR ECU Configuration, OIL-format, DBC-format, LDF-format, and Fibex-format.



Importing and exporting of complex project configurations includes selecting a file to store data to or load from, selecting a subset of the configuration parameters, selecting name prefixes, etc.

6.10.2.1. Step-by-step guide

Before you can export or import configuration data you need to create an export or import configuration for it. This configuration enables you to run the importer with the same settings later again. It stores all the settings you have entered for that particular import or export, and you can run that importer or exporter later again just by selecting it from a list.

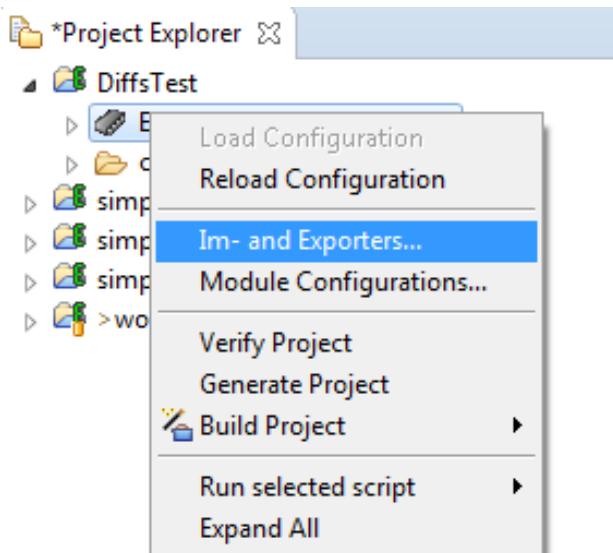
NOTE

Do not confuse import or export configurations with configuration projects. Importing and exporting of whole projects is covered in [Section 6.10.1, “Exchanging a complete EB tresos Studio project with other installations”](#)

6.10.2.1.1. Step 1: Using the Create, manage and run im- and exporters window

To create an import or export configuration for a configuration project:

- ▶ Select the ECU configuration node of the project you want to import to.
- ▶ Make sure your project is loaded. if not, load your project first (see [Section 6.3.6.3, “Loading a project”](#)).
- ▶ Right-click on the ECU configuration node.





► Click **Im- and Exporters**.

The window **Create, manage and run im- and exporters** opens up.

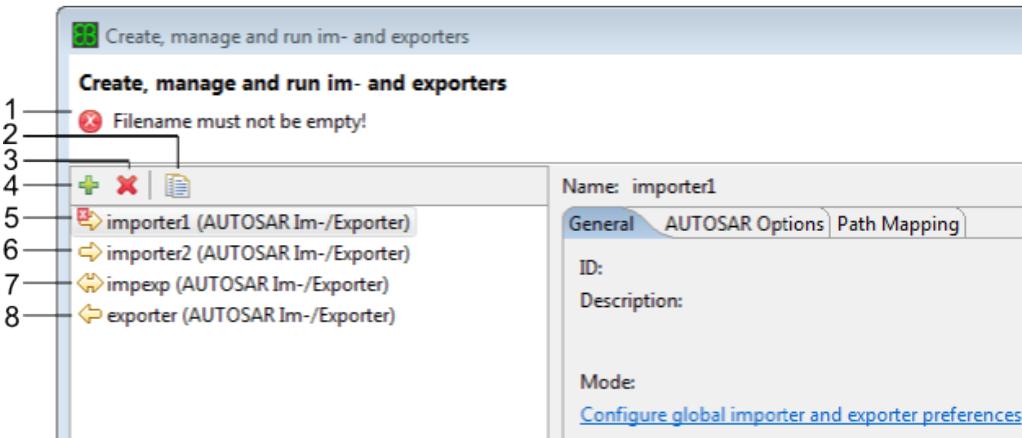
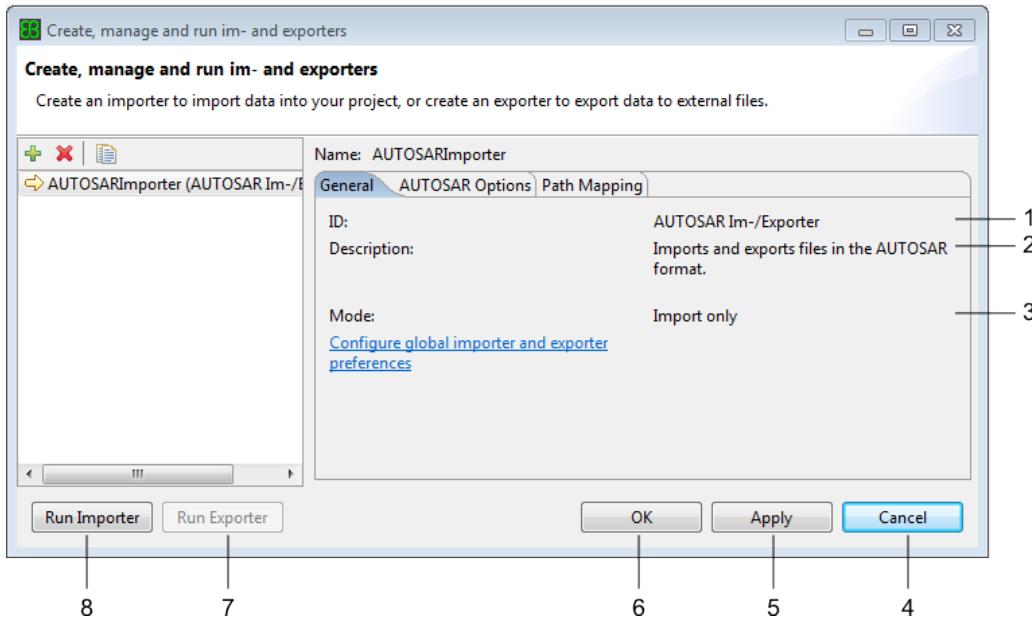


Figure 6.142. Top of the **Create, manage and run im- and exporters** window

Number	Icon	Explanation
1		If there is a problem with the importer or exporter configuration, it is shown here. If there are no problems, this section just contains a describing text.
2		Duplicate the selected import/export configuration.
3		Deletes the selected import/export configuration.
4		Adds a new import/export configuration.
5		The red cross in the top left corner indicates that this configuration contains an error.
6		This configuration will import data into EB tresos Studio.
7		This configuration will be able to do both: import and export data.
8		This configuration will export data from EB tresos Studio into other formats.

Figure 6.143. The **Create, manage and run im- and exporters** window

Number	Menu option/button	Explanation
1	ID	Type of the importer/exporter. Determines the format that is being imported or exported as well as the overall functionality of the importer/exporter.
2	Description	Describes the importer/exporter type that is currently selected.
3	Mode	The configuration mode. Modes can be: <ul style="list-style-type: none"> ▶ import ▶ export ▶ import and export
4	Cancel	Discards all changes that have been made since you opened the window or since you clicked the Apply button. Closes the window without asking if you want it closed.
5	Apply	Saves changes to the settings. The window stays open.
6	OK	Saves changes to the settings and closes the window.
7	Run Exporter	Exports data into another format by using the selected export configuration.
8	Run Importer	Imports data into EB tresos Studio by using the selected import configuration.

If you select one of the importer or exporter configuration on the left, the main part of the window displays the data of that configuration. The layout and the available GUI depends on the importer or exporter which you



selected. All importers and exporters have an **Overview** tab which displays general information. Depending on the type, there may also other tabs be present.

6.10.2.1.2. Step 2: Creating an export or import configuration

In the window **Create, manage and run im- and exporters**:

- ▶ Click the **Create a new im- or exporter** button.

The **New importer/exporter** window opens up.

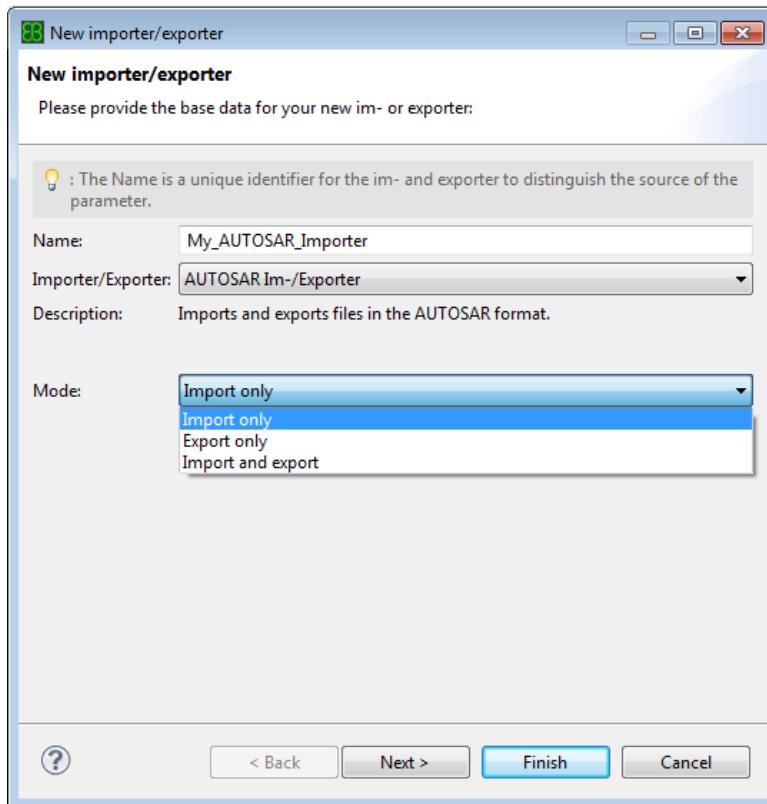


Figure 6.144. The **New importer/exporter** window

- ▶ Enter a name for your new configuration into the **Name** text box.
- ▶ Select the type of importer or exporter from the **Importer/Exporter** drop-down list box. The **Description** text displays a short text explaining what the selected importer or exporter does.
- ▶ Select the mode (import, export, or import and export) from the **Mode** drop-down list box.
- ▶ Click **Next**.
- ▶ Depending on the importer or exporter, further pages may open up into which you may enter importer- or exporter-specific data.



For further information, see the chapter of the respective importer or exporter.

- ▶ To switch between pages, click the **Back** button or the **Next** button.
- ▶ To finish creating the importer or exporter configuration, click the **Finish** button.
- ▶ To dismiss the entered configuration, click the **Cancel** button.

6.10.2.1.3. Step 3: Saving changes to an import or export configuration

After creating the importer or exporter configuration, you can still make changes to it in the **Create, manage and run im- and exporters** window. To save any changes you made:

- ▶ Click **Apply**.

To discard changes, click **Cancel** (instead of **Apply**).

6.10.2.1.4. Step 4: Importing or exporting project configuration data

To import or export configuration data:

- ▶ Select the import or export configuration which you want to run from the list on the left side, i.e. a configuration you just created.

Depending on whether the selected configuration is set to run in import, export, or in both modes, the **Run Importers** and **Run Exporters** buttons are enabled or disabled.

- ▶ Click **Run Importer** or **Run Exporter**, depending on the task you wish to perform.

A status window opens up to inform you about the progress of your task. Then a message box provides feedback whether the importer or exporter has run successfully or not.



Figure 6.145. Status window

Possible feedback types are:



- ▶ Success
- ▶ Warning
- ▶ Error

In case of error or warning feedback, EB tresos Studio logs the feedback to the error log (see [Section 5.3.5, "Error Log view"](#)).

NOTE**Execute the Update Service Component and BSWM Descriptions wizard after an importer run**

Note that configuration importer runs may reset the hardware dependent parts of your projects system description, for instance the compiler dependent properties of the AUTOSAR standard data types. The RTE Update Service Component and BSWM Descriptions wizard reconfigures the hardware dependent parts of the system description. Launch this wizard before you generate RTE configuration code if you have run an importer before.

For detailed information on the Update Service Component and BSWM Descriptions wizard, see the RTE or EB tresos AutoCore Generic documentation.

6.10.2.2. Importing and exporting configuration data from and to the AUTOSAR format

The EB tresos Studio GUI allows you to read and write files from and to the AUTOSAR format. For details about file content types and AUTOSAR content types, see the chapter [Section 3.5.3.2, "Content types for AUTOSAR module configuration files"](#).

To import or export project configuration data in the AUTOSAR format:

- ▶ Follow the steps in chapter [Section 6.10.2, "Importing and exporting configuration data"](#) to Step 2.
- ▶ In the **New importer/exporter** window, select the importer/exporter type **AUTOSAR Im-/Exporter**.
- ▶ Select whether you want to import, export or do both:

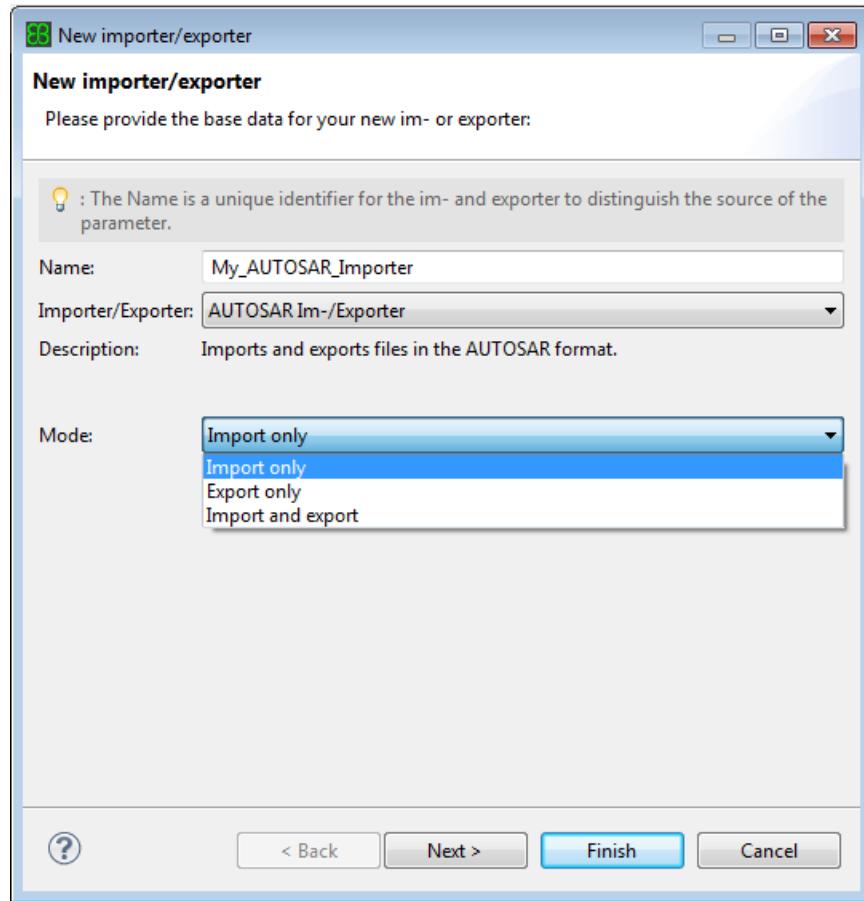


Figure 6.146. Selecting the import/export mode

- Click **Next**.

The **AUTOSAR Options** page opens up.

TIP



You may change these values later on as well

You may change the values of the **AUTOSAR Im-/Exporter** later on in the **Create, manage and run im- and exporters** dialog as well.

Depending on the selected import/export mode, the **AUTOSAR Options** page differs in some options. However, the first three options are always the same:

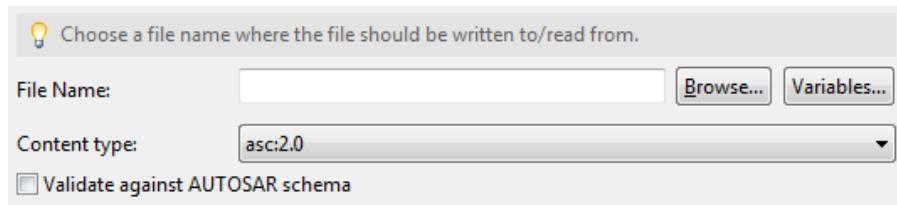


Figure 6.147. Common AUTOSAR Im-/Exporter options

- ▶ Enter the file name of the file on which the importer or exporter operates into the **File Name** text box. When you click the **Browse** button, a window appears that lets you choose a file. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, "Using path variables"](#).

If you select import mode, the file name specifies the file from which the data is imported. If you select export mode, the file name specifies the file into which the exported data is written.

- ▶ Select the content type of the selected file from the **Content type** drop-down list box. Available content types depend on the release version of your project and the importer/exporter mode:
 - ▶ Mixed release versions: If your project's release version is mixed, all content types are available.
 - ▶ **Import only** mode: Choose a content type lower or equal to your project's release version number.
 - ▶ **Export only** mode: Choose a content type equal or higher than your project's release version number.
 - ▶ **Import and export** mode: Use only the content type of your project's release version. This means for example if the release version of your project is 2.0, you cannot import files with content type asc:2.-1. However, you can export to this format.

For more information on content types, see [Section 3.5.3, "File content types"](#).

- ▶ To enable validation, select the check box **Validate**:
 - ▶ If you import a file, the file is validated against the XSD schema of the selected content type. If the file is not successfully validated, the file is not imported or exported and an error message opens up.
 - ▶ If you export a file, the exported file is written according to the selected content type. Additionally, the file is validated against the XSD schema of the selected content type.

On the lower part of the **AUTOSAR Options** page you can specify the import/export strategy and the configuration class handling.

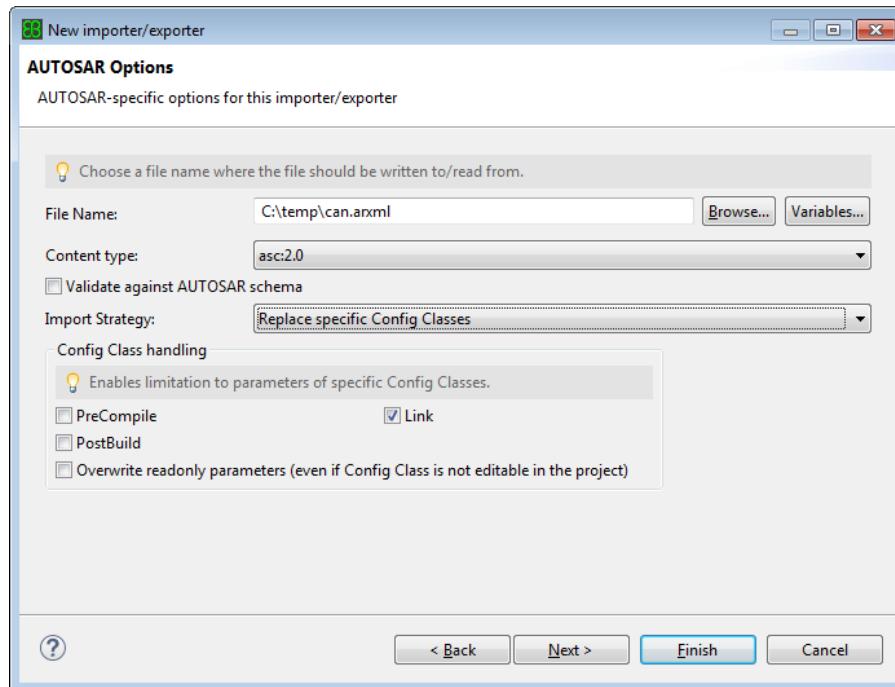


Figure 6.148. Configuring options for strategy and configuration classes

The available strategies depend on whether you have selected **Import only**, **Export only**, or **Import and export**. For more details, see [Section 6.10.2.2.2, “Strategies for importing and exporting configuration data”](#).

Strategies for import mode

- ▶ Replace, see [Section 6.10.2.2.1, “Replace”](#)
- ▶ Replace specific Config Classes, see [Section 6.10.2.2.2, “Replace specific Config Classes”](#)
- ▶ Merge, see [Section 6.10.2.2.3, “Merge”](#)
- ▶ Merge specific Config Classes, see [Section 6.10.2.2.4, “Merge specific Config Classes”](#)

Strategies for export mode

- ▶ Export, see [Section 6.10.2.2.5, “Export”](#)
- ▶ Export specific Config Classes, see [Section 6.10.2.2.6, “Export specific Config Classes”](#)

Strategies for import and export mode

- ▶ Import/Export, see [Section 6.10.2.2.7, “Import/Export”](#)
- ▶ Select the strategy and specify the configuration class handling.

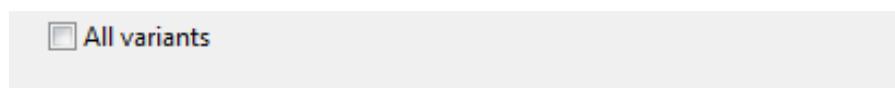


Figure 6.149. **AUTOSAR Im-/Exporter** All variants checkbox



The AUTOSAR Im-/Exporter dialog shows a checkbox to export all variants. The **All variants** checkbox is only visible in the following situations:

- ▶ If you use the **Export only** mode.
- ▶ If you use the **Import and export** mode.
- ▶ If you select a post-build selectable variant in the EcuC module of your project.

Select the checkbox **All variants** to export the complete configuration including all variants and their variation point information. Clear the checkbox **All variants** to export only the configuration of the currently active selectable variant without variation point information.

Every exported arxml file will only contain the configuration of the currently active loadable variant.

If the checkbox **All variants** is selected, then all post-build selectable PredefinedVariants including Criterions and CompuMethod definitions are exported to the file `PostBuildSelectableVariants.arxml`, which is located beneath the exported ECU configuration file.

- ▶ Click **Next**.

The **Path Mapping** page opens up.

NOTE**Path mapping is only available if configuration data is imported**

You can enable the path mapping feature only for the following modes:

- ▶ **Import only**
 - ▶ **Import and export**
-

6.10.2.2.1. Path mapping in import mode

In this section you find instructions for mapping AUTOSAR SHORT-NAME paths and package structures from the input file to the paths used internally in EB tresos Studio. Configuring path mappings might be necessary if the following conditions are met:

- ▶ The configuration which shall be imported is not created using the same AUTOSAR schema file (ECUC-MODULE-DEF) as the one from the module used in EB tresos Studio. In this case, all DEFINITION-REF elements defined within the input files are wrong and must be mapped.
- ▶ The configuration which shall be imported is contained within a package structure that is different from the structure created by EB tresos Studio. In this case, the package structure must be mapped to the structure used internally in EB tresos Studio.
- ▶ The SHORT-NAME in the configuration is different from the one that already exists in EB tresos Studio.
- ▶ The configuration which shall be imported contains references to another module configuration. The package structure of this module configuration is different from the structure created by EB tresos Studio or whose SHORT-NAME is different from the module configuration that already exists in EB tresos Studio.

- The configuration which shall be imported contains references to a system model element. The package structure of this system model element is different from the structure created by EB tresos Studio or whose SHORT-NAME is different from the system model element that already exists in EB tresos Studio.

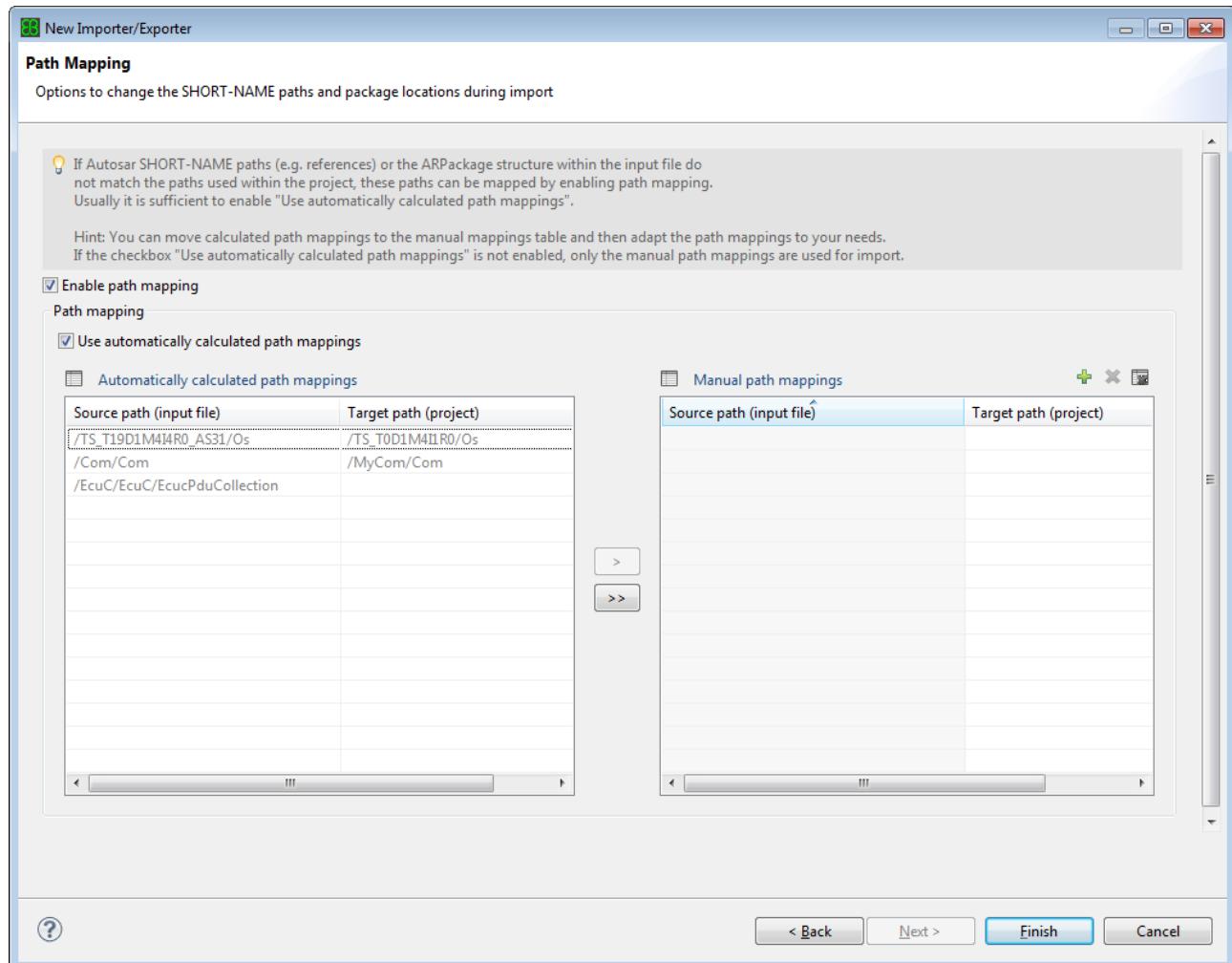


Figure 6.150. The **Path Mapping** page

With the **Enable path mapping** check box you can enable or disable the path mapping feature. When you create a new importer, this check box is disabled by default. If the path mapping feature is disabled, all widgets in the path mapping section are grayed out. You can enable the path mapping feature only for the following modes:

- ▶ Import only
 - ▶ Import and export

The table **Automatically calculated path mappings** shows all source paths of the current input file with the corresponding calculated target path. If the target path is empty, the target path is not ambiguous or cannot be calculated.



The **Automatically calculated path mappings** table is always read-only. To change path mappings, move one or more path mappings with the **>** and **>>** buttons to the **Manual path mappings** table. There you can adapt the source path and change the target path. This is necessary for mappings where no target path could be calculated automatically.

With the check box **Use automatically calculated path mappings** you can manage which path mappings shall be used for an importer run. The following use cases exist:

► **Use automatically calculated path mappings**

Only path mappings available in the **Manual path mappings** table are used for an importer run. An error is shown if the table is empty while the path mapping feature is enabled.

► **Use automatically calculated path mappings**

In addition to the path mappings in the **Manual path mappings** table, the path mappings in the **Automatically calculated path mappings** table are used during import.

NOTE**Path format**

One path mapping entry consists of a source path (from the input file) and a target path (from the current project).

TIP**Working with path mappings**

A common use case to work with the path mappings page is to enable path mapping and to check the **Use automatically calculated path mappings** check box. Then move at least all mappings with an empty target path to the **Manual path mappings** table and choose an available target path for the project.

6.10.2.2.2. Strategies for importing and exporting configuration data

6.10.2.2.2.1. Replace

Use this strategy to import the complete module configuration and replace the existing one completely.

6.10.2.2.2.2. Replace specific Config Classes

To import only parameters and references of specific configuration classes, select **Replace specific Config Classes**. Then select the configuration classes you want to import in the **Config Class handling** panel of the



AUTOSAR Options page. Parameters and references with the selected configuration classes are imported and replace the existing parameters and references. You can only import configuration classes that are editable in your project.

Select the **Overwrite read-only parameters** check box to also import and overwrite not editable parameters of the ECU configuration. For instructions for changing the configuration time of your project, see [Section 6.3.7.3, “Editing the ECU Configuration settings”](#).

6.10.2.2.2.3. Merge

If you choose this import strategy, EB tresos Studio combines the data from the AUTOSAR import file with the existing configuration data of the project.

- ▶ Not yet existing elements are added.
- ▶ Already existing elements with the same AUTOSAR `SHORT-NAME` path are overwritten.
- ▶ If a list of elements without `SHORT-NAME` (e.g. a list of integer values) is overwritten, each new element overwrites the existing element at the same position within the list. If the existing list contains more elements, the last elements which are not overwritten stay unchanged.
- ▶ If an AUTOSAR importer with the same name is used again, the importer removes or disables elements, or sets elements to their default value, that are not contained in the import file any longer.

NOTE**Multiple Configuration Container not affected**

This does not apply to existing *Multiple Configuration Container* elements. They are kept, even if they did not originate from an AUTOSAR importer with the same name.

The AUTOSAR importer merge strategy cannot do any of the following things:

- ▶ Remove or disable elements, or set elements to their default value, which have not been imported before by an AUTOSAR importer with the same name.
- ▶ Enable an optional disabled element without changing its value.

6.10.2.2.2.4. Merge specific Config Classes

To combine the existing module configuration with specific configuration classes of the module configuration you want to import, select **Merge specific Config Classes**. Then select the configuration classes you want to import in the **Config Class handling** panel of the **AUTOSAR Options** page. Parameters and references with the selected configuration classes are imported and merged with the existing parameters and references. You can only import configuration classes that are editable in your project.



Select the **Overwrite read-only parameters** check box to also import and overwrite non-editable parameters of the ECU configuration. For instructions on how to change the configuration time of your project, see [Section 6.3.7.3, “Editing the ECU Configuration settings”](#).

6.10.2.2.2.5. Export

To export the complete module configuration into the specified file, select **Export**.

6.10.2.2.2.6. Export specific Config Classes

To export only parameters and references of specific configuration classes, select **Export specific Config Classes**. Then select the configuration classes you want to export in the **Config Class handling** panel of the **AUTOSAR Options** page. The selected configuration classes are exported into the specified file.

6.10.2.2.2.7. Import/Export

An importer configuration for import and export mode can be used for both importing and exporting configuration data. If you import configuration data, the *Merge* strategy is used, see [Section 6.10.2.2.2.3, “Merge”](#). During export of configuration data, the *Export* strategy is used. For details, see [Section 6.10.2.2.2.5, “Export”](#).

NOTE**Configuration classes are always disabled in the Import and export mode**

In the **Import and export** mode of the **AUTOSAR Im-/Exporter**, the configuration class handling option is always disabled.

6.10.2.2.3. Using path variables

EB tresos Studio supports using the predefined variables defined by Eclipse as well as creating own variables. All created variables are workspace specific. If you import a project, which uses a created path variable, into another workspace you need to define the variable again. If a variable cannot be resolved then an error appears.

To define your own variables:

- ▶ In the **Window** menu, select **Preferences...**
- ▶ The **Preferences** dialog opens up
- ▶ On the left side, navigate to **Run/Debug / String Substitution** page.



► Create your own variable

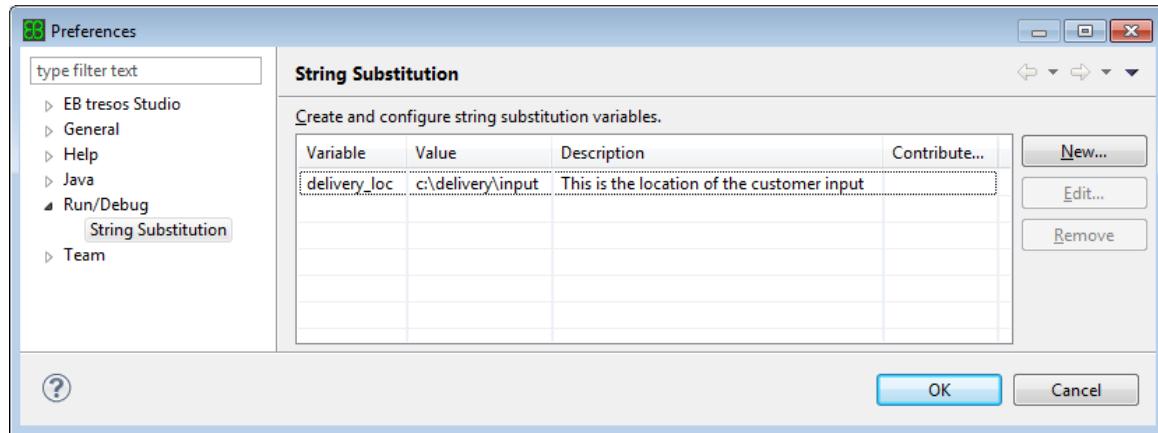


Figure 6.151. Define a new variable

You can use your variable e.g. in all importers/exporters and for the location of a module configuration file when adding a module configuration to the project.

For example in the AUTOSAR Im-/Exporter on the **AUTOSAR Options** page press the **Variables...** button. The standard Eclipse Variables dialog opens up. Here you can select either the predefined variables or your own newly created variable.

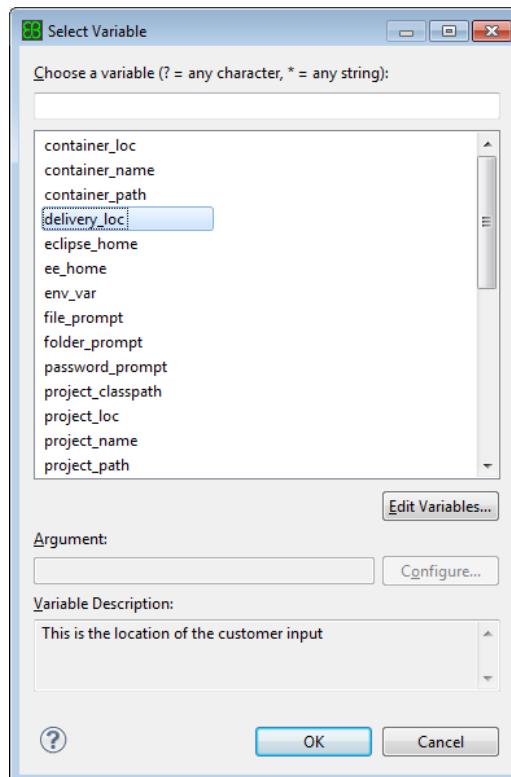


Figure 6.152. Select a variable



Select a variable and complete the filename, so it points to an existing file.

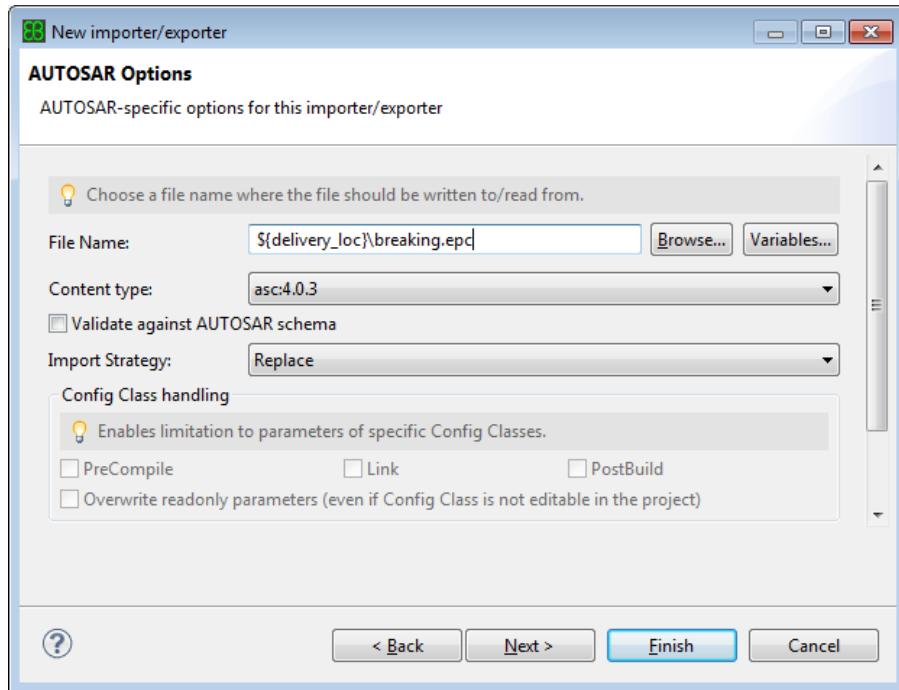


Figure 6.153. AUTOSAR importer using path variable

NOTE



Using several variables

It is possible to use several variables for one path. But you can only select one variable at once. To add several variables, move the cursor to the position where to insert the variable and press **Variables** button again, to add the next variable.

6.10.2.2.4. Importing post build variants

EB tresos Studio allows you to import post build variants from an AUTOSAR arxml input file. For details about variants, see the chapter [Section 6.13.1, “Variant handling concept by AUTOSAR”](#).

6.10.2.2.4.1. Importing post build selectable variants

If you want to import a file containing post build selectable variants, the target project needs to be configured for those selectable variants prior the import.

If you import a file without any post build selectable variants into a project which is configured for selectable variants, the data of the file will be imported to the currently active selectable variant.



6.10.2.2.4.2. Importing post build loadable variants

According to AUTOSAR specification the import file must not contain any post build loadable variants. If you import a file with post build loadable variants, then an error is reported.

6.10.2.3. Importing AUTOSAR system descriptions

6.10.2.3.1. Overview

EB tresos Studio provides an importer that allows to import AUTOSAR system description (Sys-D) and software component description files (SWC-D). In this chapter you learn the following:

- ▶ How to import system information in [Section 6.10.2.3.3, “Importing system information”](#).

6.10.2.3.2. Background information

Supported AUTOSAR meta model formats are:

- ▶ AUTOSAR 2.1 (XSD Document Version 2.1.3 Revision 0017, xmlns="http://autosar.org/2.1.2")
- ▶ AUTOSAR 2.1 (XSD Document Version 2.1.4 Revision 0018, xmlns="http://autosar.org/2.1.4")
- ▶ AUTOSAR 3.0 (XSD Document Version 3.0.2 Revision 0003, xmlns="http://autosar.org/3.0.2")
- ▶ AUTOSAR 3.1 (XSD Document Version 3.1.0 Revision 0001, xmlns="http://autosar.org/3.1.0")
- ▶ AUTOSAR 3.1 (XSD Document Version 3.2.1 Revision 0003, xmlns="http://autosar.org/3.1.2")
- ▶ AUTOSAR 3.1 (XSD Document Version 3.3.0 Revision 0004, xmlns="http://autosar.org/3.1.4")
- ▶ AUTOSAR 3.1 (XSD Document Version 3.3.0 Revision 0004 (Daimler-specific version 2), xmlns="http://autosar.org/3.1.4.DAI.2")
- ▶ AUTOSAR 3.1 (XSD Document Version 3.3.0 Revision 0004 (Daimler-specific version 3), xmlns="http://autosar.org/3.1.4.DAI.3")
- ▶ AUTOSAR 3.1 (XSD Document Version 3.3.0 Revision 0004 (Daimler-specific version 4), xmlns="http://autosar.org/3.1.4.DAI.4")
- ▶ AUTOSAR 3.2.1 (XSD Document Version 3.5.0 Revision 0001, xmlns="http://autosar.org/3.2.1")
- ▶ AUTOSAR 3.2.2 (XSD Document Version 3.6.0 Revision 0002, xmlns="http://autosar.org/3.2.2")
- ▶ AUTOSAR 4.0.2 (XSD Document Version 4.1.0 Revision 0002, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.0.3 (XSD Document Version 4.2.1 Revision 0003, xmlns="http://autosar.org/r4.0")



- ▶ AUTOSAR 4.1.0 (snapshot 2012-08-09, XSD Document Version 4.3.0 Revision 0000, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.1.1 (XSD Document Version 4.3.0 Revision 0001, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.1.2 (XSD Document Version 4.4.0 Revision 0002, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.1.3 (XSD Document Version 4.5.0 Revision 0003, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.2.1 (XSD Document Version 4.6.0 Revision 0001, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.2.2 (XSD Document Version 4.6.0 Revision 0002, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.3.0 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_4-3-0.xsd")
- ▶ AUTOSAR 4.3.1 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00044.xsd")
- ▶ AUTOSAR 4.3.1 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00045.xsd")
- ▶ AUTOSAR 4.4.0 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00046.xsd")
- ▶ AUTOSAR 4.4.0 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00047.xsd")

NOTE**Import files with different meta model versions in AUTOSAR 3.2 and newer**

It is possible to import files with different meta model versions within one single importer. But the files must either be all of meta model version 3.2.x, or meta model version 4.x.

6.10.2.3.3. Importing system information

To import configurations from the AUTOSAR Sys-D and SWC-D format, take the following steps:

1. Follow the instructions in [Section 6.10.2, “Importing and exporting configuration data”](#) to open the **Create, manage and run im- and exporters** window.
2. Click the **Create a new im- or exporter** button in the **Create, manage and run im- and exporters** window. The **New importer/exporter** dialog opens up.

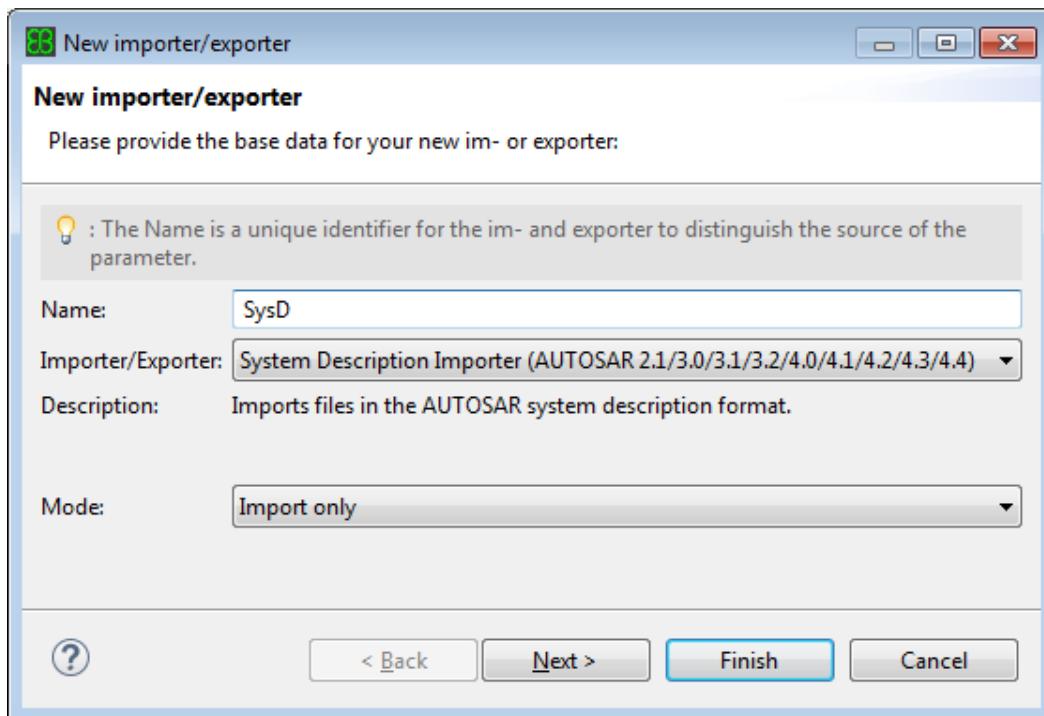


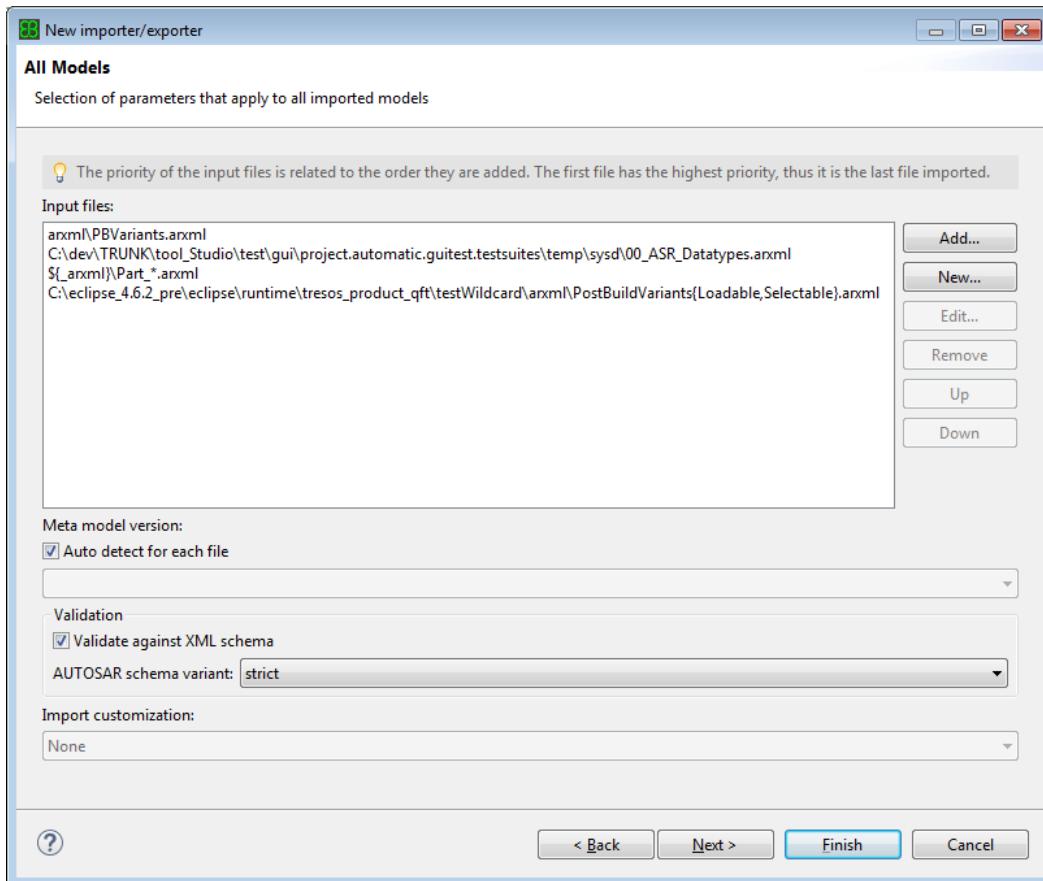
Figure 6.154. The **New Importer/Exporter** dialog

3. Enter the name of the importer configuration into the **Name** text box.
4. Select **System Description Importer (AUTOSAR 2.1/3.0/3.1/3.2/4.0/4.1/4.2/4.3/4.4)** from the **Importer/Exporter** drop-down list box.

In the **Mode** drop-down list box the value **Import only** is preselected. No other value is available.

The **System Description Importer** can only be used to import files. For information to export to the AUTOSAR format, see [Section 6.10.2.4, “Exporting AUTOSAR system description”](#).

5. To proceed to the next page, click the **Next** button. On the next page you may enter generic information.

Figure 6.155. The **All Models** page

6. You may add one or several files to the list of files by clicking the **Add...** button on the right of the **Input Files** text box.
7. You may create a new entry in the file list by clicking the **New...** button on the right of the **Input Files** text box. A dialog opens where you can enter an absolute or relative file path using variables and glob wildcards.

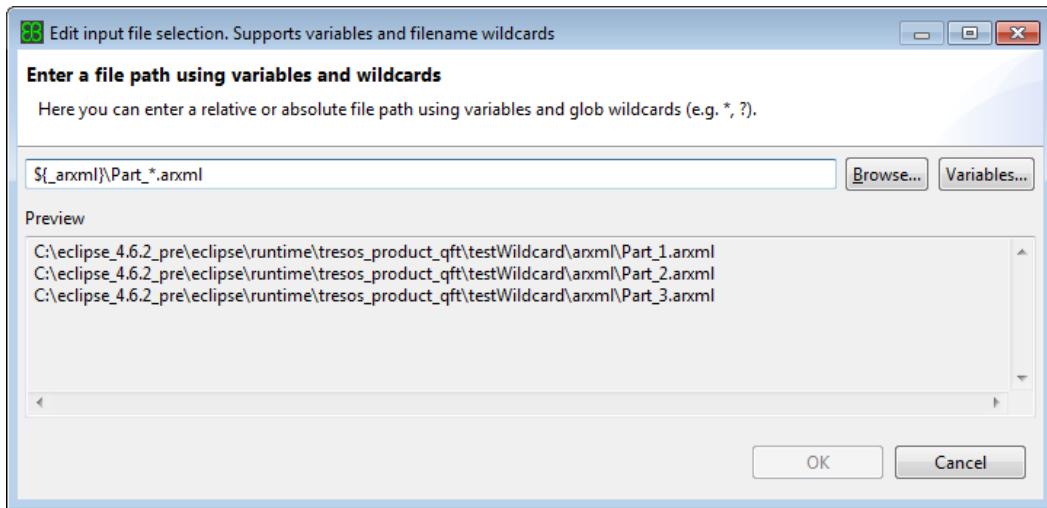


Figure 6.156. The **Input file selection** dialog supporting variables and glob wildcards

In the preview you can see which files match the given file path.

To insert a file path you can

- ▶ directly type an absolute or relative path in the text box
- ▶ browse to an existing file using the **Browse...** button
- ▶ adding a variable by using the **Variables...** button. When you click the **Variables...** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#)

NOTE

Glob wildcard patterns



For further information about glob wildcard patterns, see <http://docs.oracle.com/javase/tutorial/essential/io/fileOps.html#glob>

NOTE

Relative paths



Relative paths are always resolved to the project location.

8. You may edit an existing entry in the file list by clicking the **Edit...** button on the right of the **Input Files** text box. A dialog opens where you can change the file path using variables and glob wildcards.
9. You may remove files from the list of files with **Remove** button on the right of the **Input Files** text box.
10. You may change the order of the list of files to be loaded with the **Up** and **Down** buttons on the right of the **Input Files** text box.



NOTE**Specify import order of files**

If one system model element is available in several files, it is necessary to specify which file has the highest priority to be imported. The uppermost entry in the list is the one with the highest priority.

To define an entry with the highest priority during the import, move it up to the very top of the list.

Files resolved from an entry that uses wildcards are imported in alphabetical order.

If an input file matches several entries which use wildcards, then this file is only imported once from the uppermost entry in the list and a warning is logged in the **Error Log** view.

The list of imported files is logged as an info message in the **Error Log** view when the importer is running.

-
11. The **Meta model version** is set to **Auto detect for each file** by default when a new importer is created. Then every file is parsed with the meta model version specified in the file.

If the meta model version of a file cannot be auto detected, then an error is shown on top of the dialog and you need to manually choose the meta model version. This behavior occurs e.g. in AUTOSAR version 3.-1.5 which is currently not supported by EB tresos Studio.

To choose the AUTOSAR version of the input files manually, you need to deselect the **Auto detect for each file** check box and select one item in the drop-down list box **Meta model version**.

Then every file is read using the selected meta model version.

NOTE**If the importer has errors, you cannot run the importer anymore**

When the list of input files or the meta model version has changed, then the selected files are parsed and some consistency checks are performed. For example, the files are validated against the XML schema.

If an error has occurred, the error message is shown on top of the dialog. Then it is not possible to run the importer.

-
12. To validate the imported files against the XML schema, select the **Validate against XML Schema** check box. If this option is used in combination with **Auto detect for each file** option, then each file is validated against its own schema.

If more than one XML schema variant is available for the AUTOSAR meta model versions of the imported files, choose one from the **AUTOSAR schema variant** combo box.

**NOTE****Validation errors**

If the input files have different meta model versions and auto detection is not selected, the validation is likely to fail. In this case, it is recommended to disable the **Validate against XML Schema** check box.

13. If the **Import customization** drop-down list box is enabled, select an import customization.

The **Import customization** drop-down list box provides available import customizations registered for this kind of importer in combination with the selected meta model version. An import customization affects the import process and modifies the imported data. In most cases this list is empty and thus grayed out.

14. After you configured the **All Models** page, click the **Next** button to configure the system model import.

The **System Model Import** page opens up.

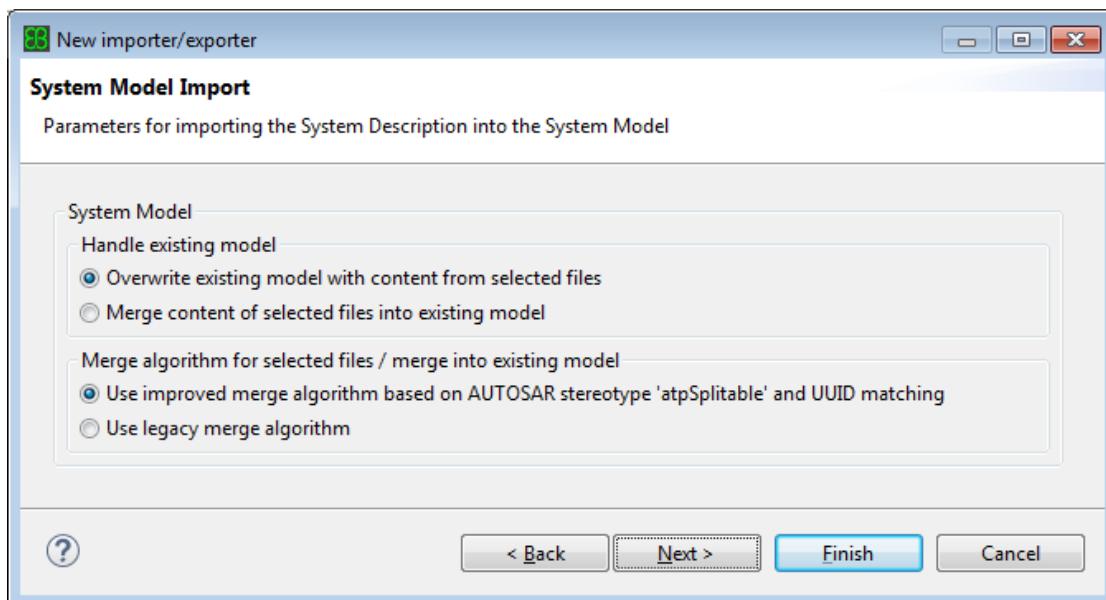


Figure 6.157. The **System Model Import** page

- ▶ Choose how to handle the existing model:
 - ▶ To completely replace the system model in the project, select **Overwrite existing model with content of selected files**.
 - ▶ To add the system model information of the selected files to the existing model, select **Merge content of selected files into existing model**.
- ▶ Choose the merge algorithm:
 - ▶ **Use improved merge algorithm based on AUTOSAR stereotype 'atpSplittable' and UUID matching**



The AUTOSAR meta model stereotypes <splitable> (3.x) / <atpSplitable> (4.x) define which elements can be split up over several physical files. The new improved merge algorithm is based on this concept.

In addition to this, the new merge algorithm uses UUID matching to find e.g. renamed elements. For example: you imported a file into the existing project model and rename an element in the file afterwards. If the element still has the same UUID, then the merge algorithm will find the correct entity and rename it in the project model when importing the file again.

For newly created System Description importer, this option is set as default.

► **Use legacy merge algorithm**

If you want to get the same result as in previous EB tresos Studio releases, then select the legacy merge algorithm.

NOTE



Use of the merge algorithm

The selected merge algorithm is used to merge multiple selected input files as well as to merge the content of the selected files into the existing model.

This means that even if you select **Overwrite existing model with content from selected files**, you need to choose **Merge algorithm for selected files / merge into exiting model** if you import more than one file. In this case these files are merged and the merged model replaces the existing system model in the project.

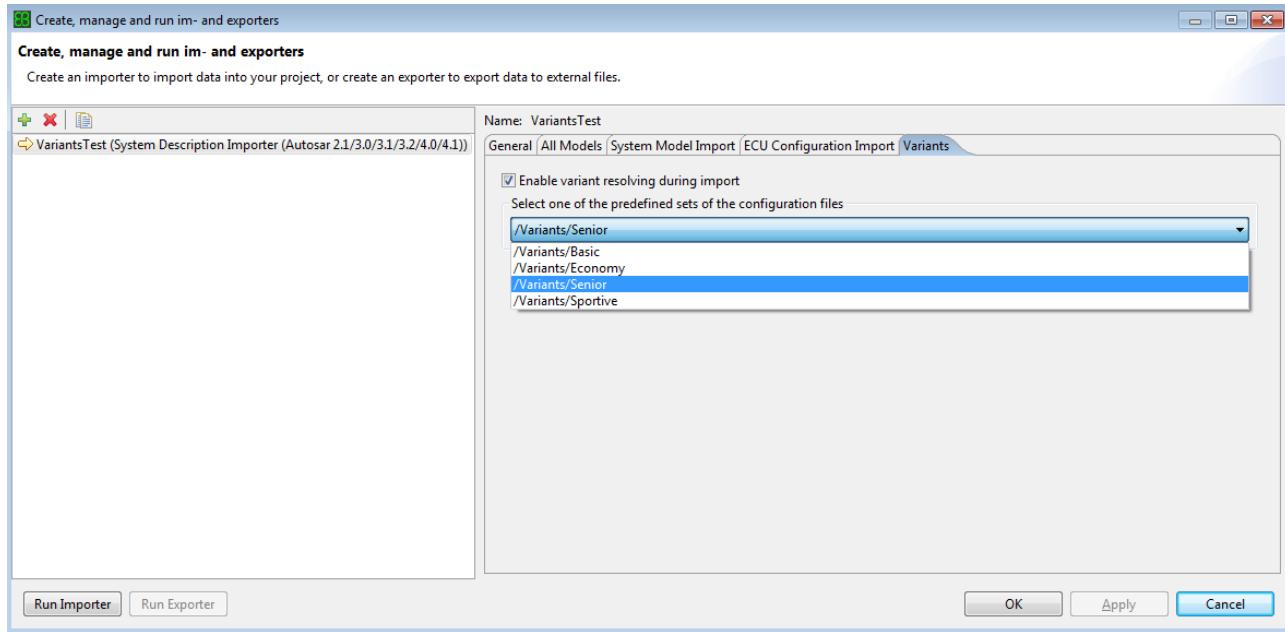
-
15. To end the importer configuration, click the **Finish** button. This closes the **New importer/exporter** dialog.
 16. To run the Sys-D Importer, select the importer name you just created on the left side of the **Create, manage, and run im- and exporters** window and click the **Run Importer** button.

6.10.2.3.4. References

Details on the ECU extract to AUTOSAR parameter mapping are available in the EB tresos AutoCore documentation.

6.10.2.3.5. Variants tab

Variant resolving can be enabled with the **Variants** tab from the System Description Importer wizard. The tab contains a check box that will allow the user to enable variant resolving, and a combo box that will allow selecting a predefined variant from a list. The predefined variants are defined in the input files.

Figure 6.158. The **Variants** tab

More details about the variant handling are available in [Section 6.13, “Working with variants and Post-build loadable”](#).

6.10.2.4. Exporting AUTOSAR system description

6.10.2.4.1. Overview

EB tresos Studio provides an exporter with which you may export the AUTOSAR system description (Sys-D) and the software component description (SWC-D) of a configuration project into a file format of your choice. In this chapter you learn:

- ▶ which AUTOSAR meta model formats are supported (at [Section 6.10.2.4.2, “Background information”](#)), and
- ▶ how to export system information (at [Section 6.10.2.4.3, “Exporting system information”](#)).

6.10.2.4.2. Background information

Supported AUTOSAR meta model formats for export are:

- ▶ AUTOSAR 4.0.2 (XSD Document Version 4.1.0 Revision 0002, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.0.3 (XSD Document Version 4.2.1 Revision 0003, xmlns="http://autosar.org/r4.0")



- ▶ AUTOSAR 4.1.0 (snapshot 2012-08-09, XSD Document Version 4.3.0 Revision 0000, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.1.1 (XSD Document Version 4.3.0 Revision 0001, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.1.2 (XSD Document Version 4.4.0 Revision 0002, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.1.3 (XSD Document Version 4.5.0 Revision 0003, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.2.1 (XSD Document Version 4.6.0 Revision 0001, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.2.2 (XSD Document Version 4.6.0 Revision 0002, xmlns="http://autosar.org/r4.0")
- ▶ AUTOSAR 4.3.0 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_4-3-0.xsd")
- ▶ AUTOSAR 4.3.1 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00044.xsd")
- ▶ AUTOSAR 4.4.0 (xmlns="http://autosar.org/r4.0" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00046.xsd")

6.10.2.4.3. Exporting system information

To export configurations into an AUTOSAR Sys-D format:

- ▶ Open the **Create, manage and run im- and exporters** dialog by right-clicking the ECU node of your project in the **Project Explorer** view and selecting **Im- and Exporters....**

The **Create, manage and run im- and exporters** dialog opens up.

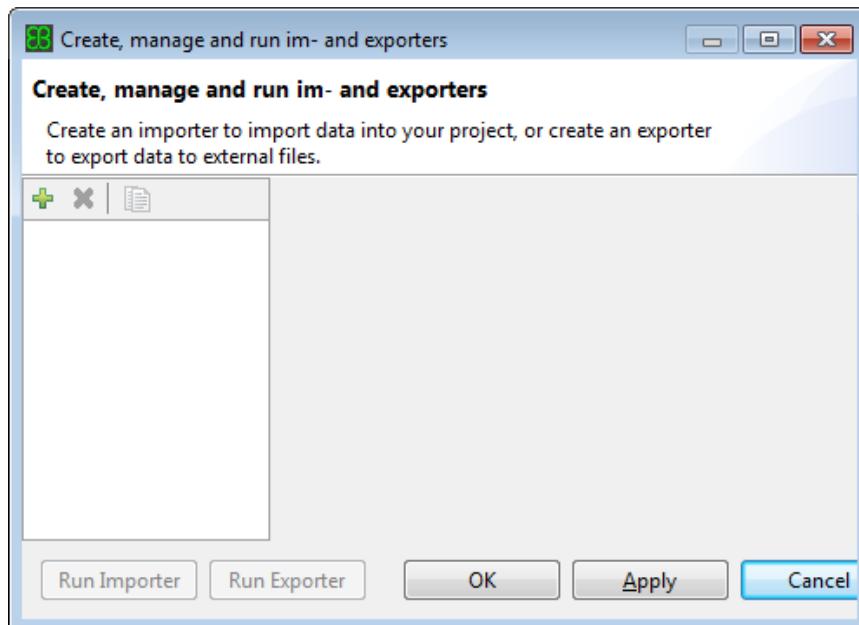


Figure 6.159. The **Create, manage and run im- and exporters** dialog



- To add a new exporter, click the **Create a new im- or exporter** button .

The **New Importer/Exporter** dialog opens up.

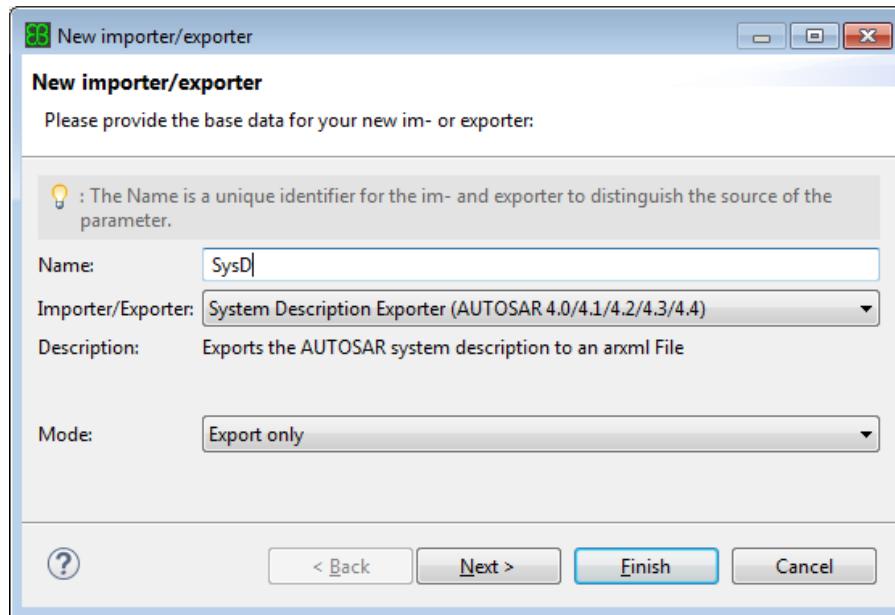


Figure 6.160. Configuring your new exporter

- In the **Name** text box, enter the name of your exporter configuration.
- In the **Importer/Exporter** drop-down list box, select **System Description Exporter (AUTOSAR 4.0/4.-1/4.2/4.3/4.4)**.
- In the **Mode** drop-down list box the value **Export only** is preselected. No other value is available.

With the **System Description Exporter** you can only export into one single file.

- To proceed to the next page, click **Next**. The **System Model Export** page opens up.

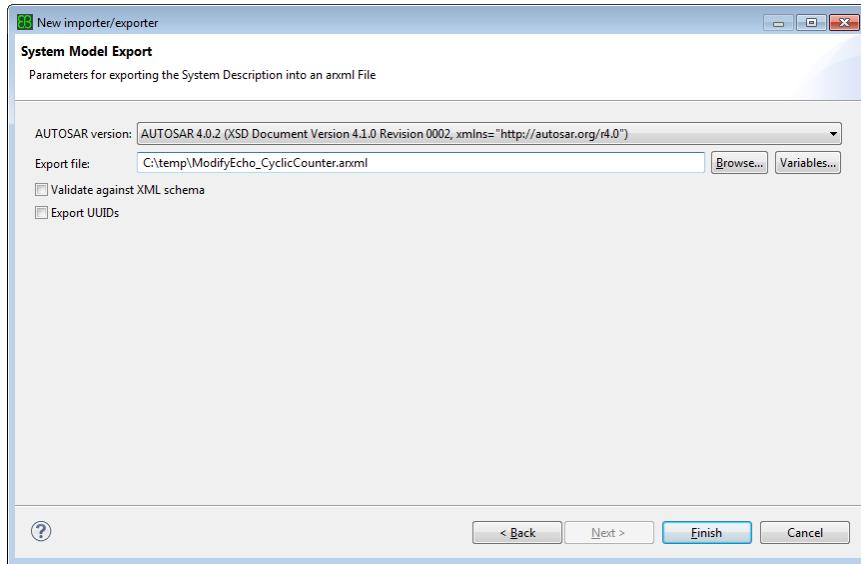


Figure 6.161. Specifying your exporter

- ▶ In the **AUTOSAR version** drop-down list box, select the AUTOSAR version you want to export to.
- ▶ In the **Export file** text box, browse to the file you want to export to or browse to a directory of your choice and enter a new file name. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, "Using path variables"](#)

NOTE**Selected files are overwritten**

If you select a file that already exists, this file will be overwritten by the exported file.

- ▶ If you want to validate the exported file against the Sys-D XML schema of the selected AUTOSAR version, select the **Validate against XML Schema** check box.
- ▶ If you want to export UUID attributes, select the **Export UUIDs** check box.
- ▶ Click **Finish**. The configuration of the exporter is saved and the **New Importer/Exporter** dialog closes.
- ▶ To run your exporter, select the exporter name you just created on the left side of the **Create, manage, and run im- and exporters** dialog.
- ▶ Select **Run Exporter**.
- ▶ When the export is finished, you find the exported Sys-D file in the directory you have specified.



6.10.2.5. Importing from the OIL format

6.10.2.5.1. Overview

EB tresos Studio provides an integrated OIL Importer to import Os configurations from OIL files. If an OIL file is available for import, the OIL Importer simplifies and speeds up the configuration of the Os module. It is also possible to import OIL files via the command line. This chapter describes the tool's support for the OIL format from the GUI and the command line.

6.10.2.5.2. OIL Importer and file format

The OIL Importer can read an OIL file and is able to configure the Os module.

The OIL Importer creates default configurations for hardware-related options. It creates one Os configuration.

During the import, the OIL Importer creates several containers for frames and signals. The names chosen for these containers are based on the names defined in the OIL file.

The mapping of the OIL format to the AUTOSAR parameters is described in the chapter `The Oil attribute naming translation` of the Os module documentation.

6.10.2.5.3. Importing OIL configurations

Since the OIL Importer does not create module configurations, you need to assure that an Os module configuration is present.

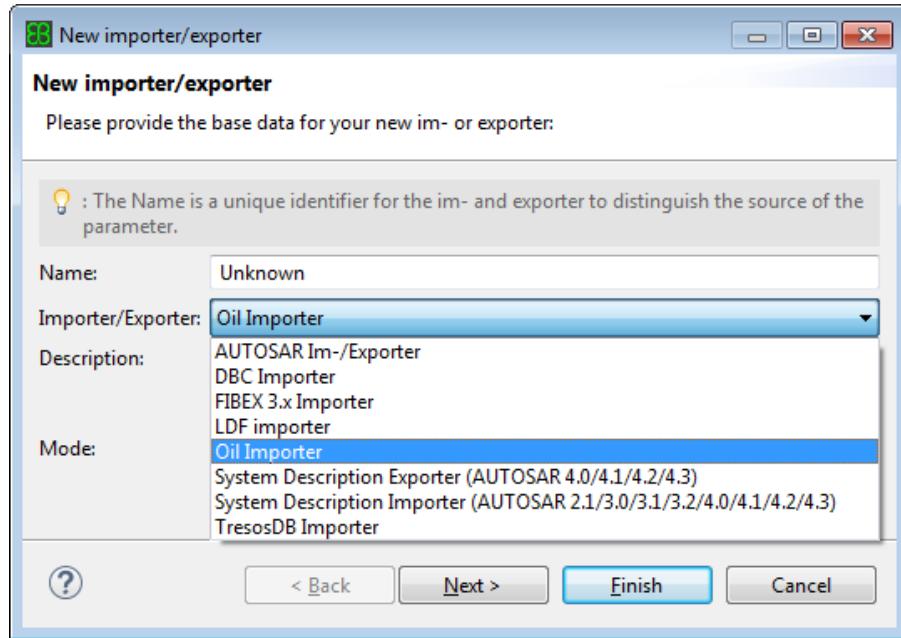
To import a configuration from the OIL format:

- ▶ Refer to [Section 6.10.2, “Importing and exporting configuration data”](#).
- ▶ Follow the instructions of Step 1 until you arrive at the window **Create, manage and run im- and exporters**.

In the window Create, manage and run im- and exporters:

- ▶ Click the **Create a new im- or exporter** button.

The **New Importer/Exporter** window opens up.



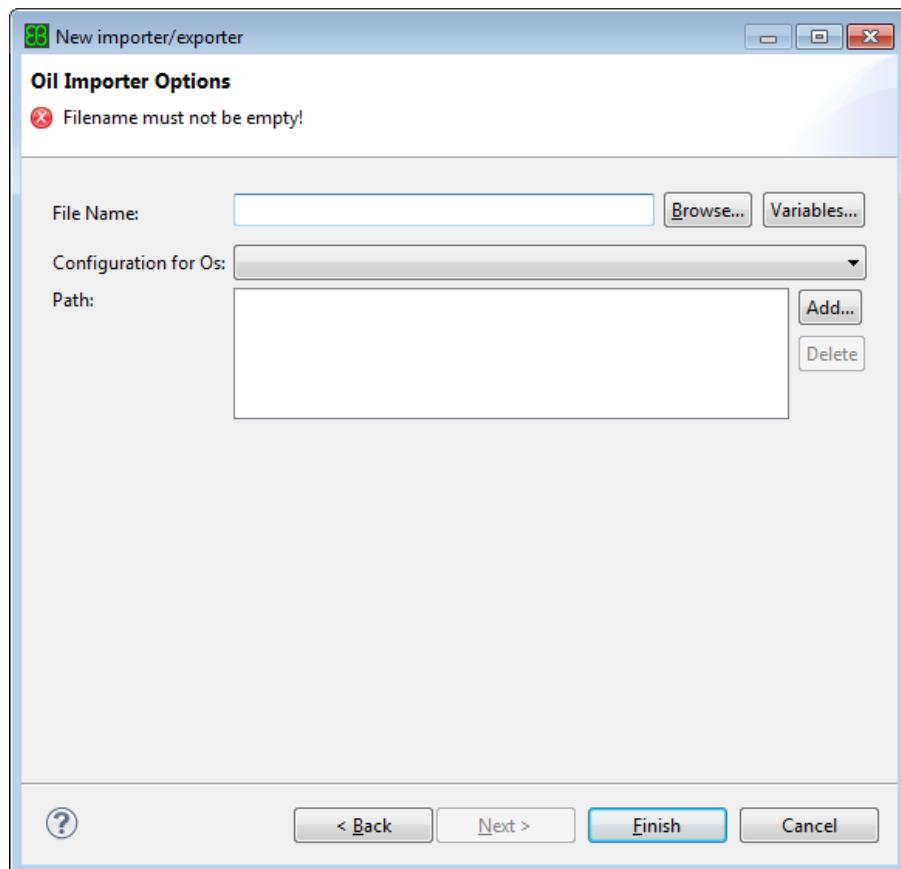
To enter the basic information for the new importer or exporter configuration:

- ▶ Select the **Name** text box and enter a name for your new configuration.
- ▶ Select **OIL Importer** from the **Importer/Exporter** drop-down list box.
- ▶ Select the **Mode** line drop-down list box.
- ▶ Select **Import only** from the drop-down list box.

The OIL Importer can only be used to import files. Exporting to OIL format is not possible.

- ▶ Click the **Next** button.

The **OIL Importer Options** page is shown.



- ▶ In the **File Name** text box browse for the OIL file you want to import into EB tresos Studio. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, "Using path variables"](#)
- ▶ In the **Configuration for Os** menu line, select the target Os configuration for the import.
- ▶ In the **List of Path names** menu line, select a list of path names as include paths to additional OIL files.

To save the importer configuration:

- ▶ Click **Finish**.

This saves your importer configuration and closes the **OIL Importer Options** window.

To run the OIL Importer, select the file name you just created on the left side of the **Create, manage, and run im- and exporters** window.

- ▶ Click **Run Importer**.

The OIL configuration is being imported.



6.10.2.5.4. Importing OIL files via the command line

The OIL Importer of EB tresos Studio can also be used on the command line, e.g. to import OIL files in batch mode. This import function is encapsulated into the **legacy convert** command.

You may need to edit the configuration that has been created during the import before you use it to configure an ECU.

6.10.2.5.4.1. OIL Importer parameters for the command line

The commandline OIL Importer requires passing some system properties to the command line via the **-D** option. These are:

- ▶ **target** (mandatory): Specifies the target for which to import.
- ▶ **oil.include** (optional): List of pathnames, where possible oil include files are located.
- ▶ **oil.package** (mandatory) ARPackage name of the selected Os configuration.
- ▶ **oil.config** (mandatory) ConfigName of the Os configuration selected .

6.10.2.5.4.2. OIL import example

The following example shows a typical use of the OIL Importer (all on one line):

```
tresos_cmd.bat -Dtarger=V850
    -Doil.include=C:\;D:\parser
    -Doil.package=Os
    -Doil.config=Os
    -Dmsg1648=false legacy convert
    network.oil output.epc
```

This line will import all frames and signals from the file `network.oil` into the Os module for the V850 architecture.

The `-Dmsg1648=false` parameter disables a warning message that would otherwise appear. The warning - if not disabled - informs users that all nodes that are required are generated automatically. Since this applies only to nodes that are not extracted from the OIL file, you may safely ignore this warning.

6.10.2.6. Importing CAN configurations from the DBC format



6.10.2.6.1. Overview

EB tresos Studio provides an integrated DBC Importer to ease and speed up the configuration of the CAN stack if an appropriate DBC file is provided. To import LIN configurations, continue reading at [Section 6.10.2.7, "Importing from the LDF format".](#)

6.10.2.6.2. Background information

The DBC Importer can read a DBC file and fill in the system model of a project.

The DBC file format is a proprietary file format that has a lot of restrictions, e.g.:

- ▶ The definition of multiple networks is not supported.
- ▶ The definition of a network of a bus type other than CAN is not supported.

NOTE**Import from CAN networks**

The DBC Importer can only import CAN networks from a DBC file.



AUTOSAR system description files do not have these drawbacks. AUTOSAR system description files provide all the data a DBC file can provide and in addition support:

- ▶ multiple networks
- ▶ gateways
- ▶ higher level protocol layers such as Tp or Nm

NOTE**Use of AUTOSAR system description files recommended**

If the supplier of a system model also offers AUTOSAR system description files, it is highly recommended to use AUTOSAR system description files.



DBC files provide a feature that allows the creator of the DBC file to attach any number of additional so-called *attributes*. Attributes are uniquely identified by a freely configurable name and are assigned to one of the object types a DBC file supports: CAN Network, CAN Message, Signal, or CAN Communication Controller.

Since the meaning and the physical unit of the attribute values is not available in the DBC file, the DBC Importer has to store the attribute information in a generic way.



If the custom attributes correspond to standardized AUTOSAR parameters, an *import customization* can implement the mapping from attribute values to standardized AUTOSAR parameter values. Tools that are executed after the mapping took place, for instance the components that configure the various BSW modules, can use the translated AUTOSAR parameters as input.

How to obtain an *Import customization* is described in [Section 6.10.2.6.4.1, “Import customizations”](#).

NOTE**Modules are not configured automatically by the DBC Importer**

The DBC Importer does not provide the data to import ECU configuration for CanTp, CanNm, and Nm modules unless you select an **Import customization** which detects CanTp N-PDUs, or CanNm NM-PDUs by using customer-specific properties in the imported .DBC file. If you do not select **Import customization**, the modules CanTp, CanNm, and Nm are not configured when you run the ECU configuration importer.

6.10.2.6.2.1. Limitations

This section lists the restrictions of the EB tresos Studio DBC Importer.

- ▶ No LIN configurations can be imported with the DBC Importer. (There is an LDF Importer to do this, see [Section 6.10.2.7, “Importing from the LDF format”](#))
- ▶ Not every DBC file may be understood by the DBC Importer, because DBC is a proprietary format and the DBC Importer can only interpret a subset of it (which was up to now sufficient to extract the information required). See [Section 6.10.2.6.4.2.1, “DBC grammar”](#) for more information on this subject.

6.10.2.6.3. Importing DBC configurations

The DBC Importer enables the import into the system model.

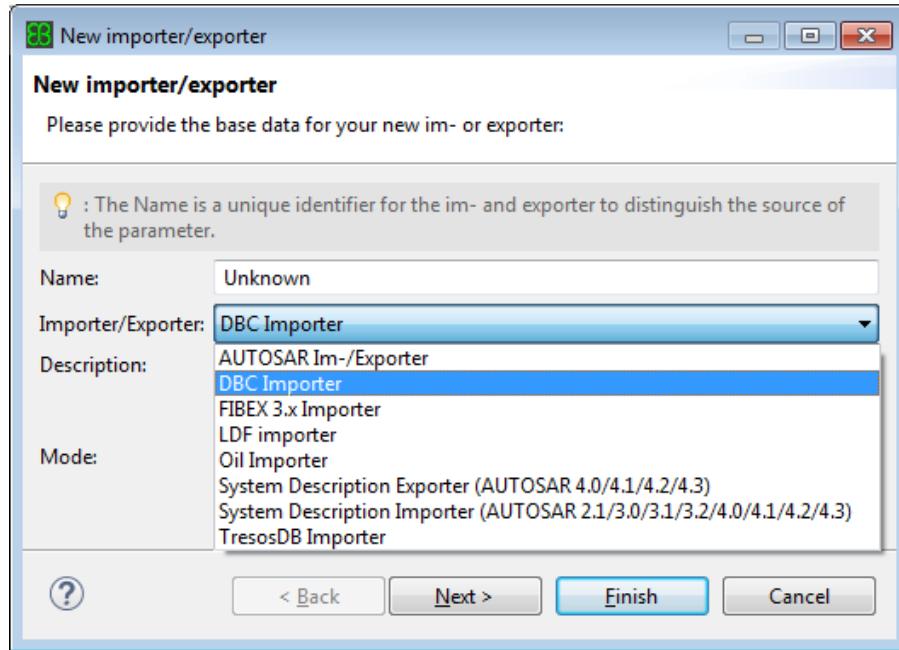
To import configurations from the DBC format:

- ▶ Refer to [Section 6.10.2, “Importing and exporting configuration data”](#).
- ▶ Follow the instructions of Step 1 until you arrive at the window **Create, manage and run im- and exporters**.

In the window Create, manage and run im- and exporters:

- ▶ Click the **Create a new im- or exporter** button.

The **New Importer/Exporter** window opens up.



To enter the basic information for the new importer or exporter configuration:

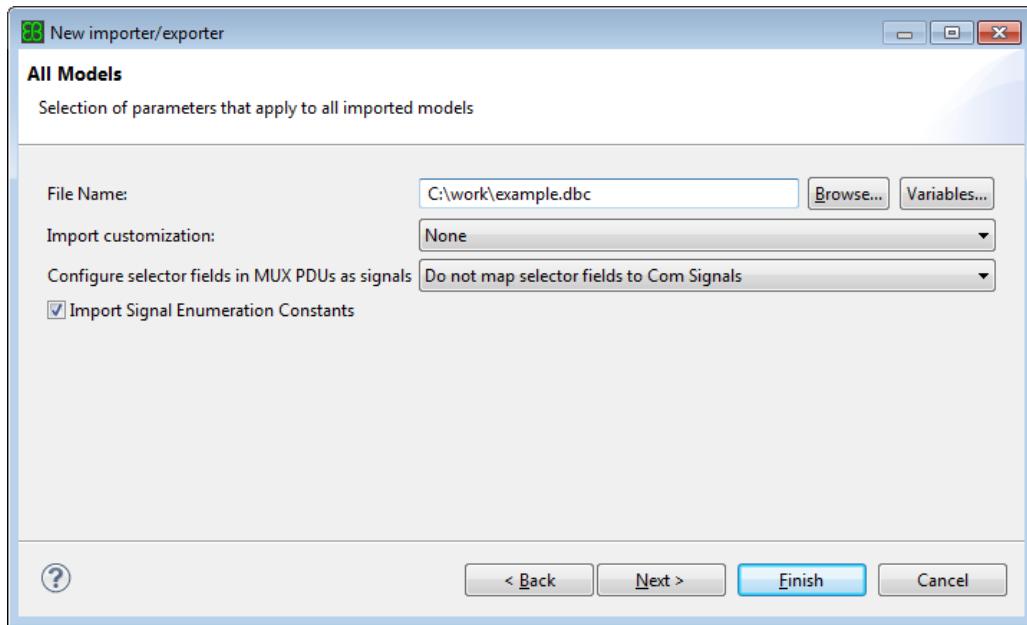
- ▶ Select the **Name** text box.
- ▶ Enter a name for your new configuration.
- ▶ Select **DBC Importer** from the **Importer/Exporter** drop-down list box.
- ▶ Select the **Mode** drop-down list box.
- ▶ Select **Import only** from the drop-down list box.

The DBC Importer can only be used to import files. Exporting to DBC format is not possible.

- ▶ Click **Next**.

The **All Models** window opens up.

The **All Models** window is provided for entering general information.



- ▶ Select the **File Name** file selection entry. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#)
- ▶ Select the DBC file that you would like to import.
- ▶ If the **Import customization** drop-down list box is enabled, select an import customization or select **None** if you don't want to execute any import customization.

The **Import customization** drop-down list box provides available import customizations registered for this kind of importer. An import customization affects the import process and modifies the imported data. In most cases this list will be empty and thus grayed out.

- ▶ With the **Configure selector fields in MUX PDUs as signals** drop down list box, you may specify how multiplexed PDUs and their contained selector fields shall be imported. Three modes are available: **Do not map selector fields to Com Signals**, **Map selector fields to Com Signals**, and **AUTOSAR compliant mapping**. For detailed information about this option, see [Section 6.10.2.6.4.3, “I-Pdu Multiplexer Configuration”](#).
- ▶ To import the content of `VAL_` tokens from the DBC file, select the **Import Signal Enumeration Constants** check box. Further information on the `VAL_` tokens is available at (see [Section 6.10.2.6.4.2.1, “DBC grammar”](#)).

For each imported signal enumeration, the generated `Rte` code contains a C language macro that maps a symbolic name to a specific value. The symbolic name can be used by the application code. The generated code, however, is only valid if the following conditions meet:

- ▶ All symbolic names must be valid C language identifiers.
- ▶ Every symbolic name must represent exactly one numeric value.



If you require the generated Rte code and the imported file violates any of these conditions, clear the **Import Signal Enumeration Constants** check box.

- ▶ Click **Next**.

The **System Model Import** window opens up.

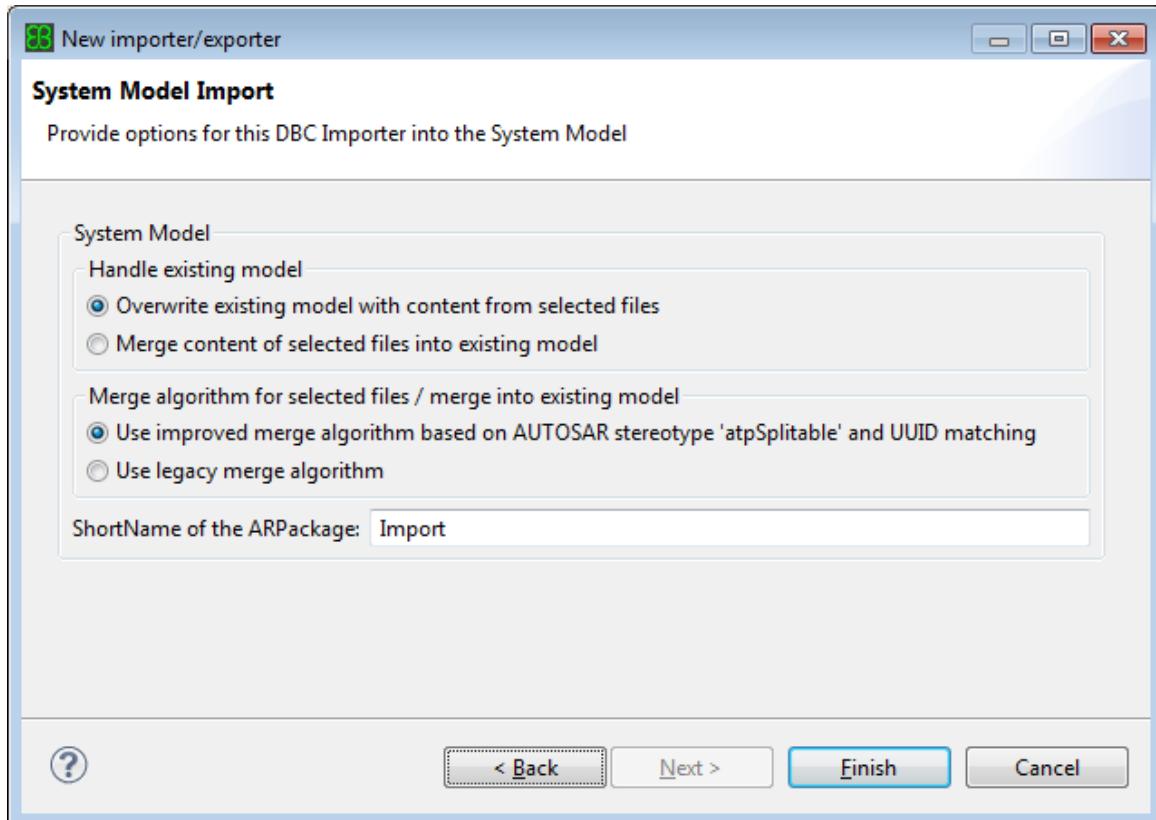


Figure 6.162. The **System Model Import** page

- ▶ Choose how to handle the existing model:
 - ▶ To completely replace the system model in the project, select **Overwrite existing model with content from selected files**.
 - ▶ To add the system model information of the selected files to the existing model, select **Merge content of selected files into existing model**.
- ▶ Choose the merge algorithm:
 - ▶ **Use improved merge algorithm based on AUTOSAR stereotype 'atpSplittable' and UUID matching**

The AUTOSAR meta model stereotypes <splitable> (3.x) / <atpSplittable> (4.x) define which elements can be split up over several physical files. The new improved merge algorithm is based on this concept.



In addition to this, the new merge algorithm uses UUID matching to find e.g. renamed elements. For example: you imported a file into the existing project model and rename an element in the file afterwards. If the element still has the same UUID, then the merge algorithm will find the correct entity and rename it in the project model when importing the file again.

For newly created System Description importer, this option is set as default.

► **Use legacy merge algorithm**

If you want to get the same result as in previous EB tresos Studio releases, then select the legacy merge algorithm.

- To select the AUTOSAR short-name of the AUTOSAR AR-PACKAGE into which the system model of the source file is to be imported, enter the AUTOSAR short-name into the **ShortName of the ARPackage** text box.

To save the importer configuration:

- Click **Finish**.

This saves your importer configuration and closes the window.

- To run the DBC Importer, select the importer name you just created on the left side of the **Create, manage, and run im- and exporters** window.
- Click **Run Importer**.

The DBC configuration is being imported.

6.10.2.6.4. References

6.10.2.6.4.1. Import customizations

An *import customization* is an extension to the DBC Importer which translates DBC attributes into standard AUTOSAR parameters. Since this translation happens before the various BSW modules are configured, the BSW module configurations can take advantage of the translated parameters.

You cannot implement import customizations by yourself because they read from and write into the EB tresos Studio internal data model. If you require an import customization for your DBC file, contact the support hot-line to order an implementation of the import customization by means of an engineering service. Enclose a rough description on which attributes you want to have processed and, if possible, an example DBC file. The effort to set up such an *Import customization* depends on the amount of translated attributes and the involved complexity.



6.10.2.6.4.2. The DBC parser

DBC is not an open file format, but a proprietary format defined by Vector Informatik. For this reason, not every DBC file may be importable by the DBC Importer. The following chapter describes which DBC files the DBC Importer can read.

For reading DBC files a DBC parser is applied. Every parser is based on a *grammar*. Since the grammar of the DBC format is not freely available, EB deduced a subset of their own, which allows parsing the information needed by the DBC Importer.

6.10.2.6.4.2.1. DBC grammar

The DBC Importer uses the following (*EBNF*) grammar to parse DBC files. The DBC Importer ignores lines that start with tags, which are not specified here.

```

dbc = version, ns, bs, controllers, {message}, {definition} ;

definition = comment | attrdef | attrinit | attribute | envVarDef | valueTable |
    senderList;

version = 'VERSION', quotedString ;

ns = 'NS_-', ':', { UC | '_' } ;

bs = 'BS_-', ':' | 'BS_-', ':' integer : integer, integer; (* values ignored *)

controllers = 'BU_-', ':', identifier, {identifier} (*controller names*) ;

message = 'BO_-', integer (*CAN-id*), identifier (*message name*), ':',
    integer (*size of data in bytes*), identifier (*sender*),
    signal, {signal} ;

senderList = 'BO_TX_BU_-, integer (*CAN-id*),
    identifier {',', identifier} (*sender*) ;

signal = 'SG_-', identifier (*signal name*), ['M'|('m', integer)], ':',
    integer (*startbit*), '|', integer (*bit length*), '@-',
    ('0' | '1') (*endianess*), ('+' | '-') (*type*),
    '(', integer (*factor*), ',', integer (*offset*), ')',
    '[', integer (*minimum*), '|', integer (*maximum*), ']',
    quotedString (*unit [km, h, kg, ...]*),
    identifier {',', identifier} (*receiver*) ;

signalGroup = 'SIG_GROUP_-, integer (*CAN-id*), identifier
    (*signal group name*), integer (*not used*),

```



```

':' identifier {',', identifier} (*signal*) ;

sigValType = 'SIG_VALTYPE_ ', integer (*CAN-id*), identifier (*signal name*),
':', integer (*signal type*);

valueTable = 'VAL_-', integer (*CAN-id*), identifier (*signal*),
    valueList {',', valueList};"

valueList = integer (*value*), quotedString (*symbol*);

valueTableEnvVar = 'VAL_-', integer (*CAN-id*),
    valueListEnvVar {',', valueListEnvVar};"

valueListEnvVar ::= /* empty */
|
integer, quotedString; (* values ignored *)

comment = 'CM_-', comment-body ';' ;

comment-body = message-comment | signal-comment | envVar-comment |
    controller-comment | net-Comment | otherComment ;

message-comment = 'BO_-', integer (*CAN-id*), quotedString (*comment*) ;

signal-comment = 'SG_-', integer (*CAN-id*), identifier (*signal name*),
    quotedString (*comment*) ;

envVar-comment = 'EV_-', identifier, quotedString (*comment*) ;

controller-comment = 'BU_-', identifier, quotedString (*comment*) ;

net-comment = quotedString (*comment*) ;

otherComment = ? This rule should eat all comment for
objects which are not supported ? ;

attrdef = 'BA_DEF_-', {'BO_-' | 'BU_-' | 'SG_-' | 'EV_-'},
    quotedString (*attribute name*), dataType, ';' ;

dataType = 'STRING' |
    'ENUM', quotedString, {',', quotedString} |
    'INT', integer (*lower bound*), integer (*upper bound*) |
    'HEX', integer (*lower bound*), integer (*upper bound*) |
    'FLOAT', float (*lower bound*), float (*upper bound*) ;

attrinit = 'BA_DEF_DEF_-', quotedString (*attribute name*), attrVal ';' ;

```



```

attrVal = float | integer | quotedString ;

attribute = 'BA_-' , quotedString (*attribute name*), attrDest, attrVal, ';' ;

attrDest = 'BU_-' , identifier (*controller name*) |
           'BO_-' , integer (*CAN-id*) |
           'SG_-' , integer (*CAN-id*), identifier (*signal name*) |
           'EV_-' , identifier {*env. var. name*} ;

envVarDef = 'EV_-' , identifier (*variable name*), ':',
            integer (*value type*),
            '[' , integer (*minimum*), '|', integer (*maximum*), ']',
            quotedString (*unit*), integer (*default value*),
            integer (*number of envVarDef*),
            identifier (*contains access information*),
            identifier {',' , identifier} (*controllers*), ';' ;

exponent = [Ee], ['+' | '-'], digit, {digit} ;
integer = ['-'] digit, {digit} ;
float = ['-'], digit, {digit}, '.', {digit}, [exponent] | ['-'], digit,
        {digit}, exponent ;
identifier = (letter | '_-'), {letter | digit | '_-'} ;
digit = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' ;
letter = LC | UC ;
nl = ? new line ? ;
UC = 'A' | 'B' | 'C' | 'D' | 'E' | 'F' | 'G' | 'H' | 'I' | 'J' | 'K' |
     'L' | 'M' | 'N' | 'O' | 'P' | 'Q' | 'R' | 'S' | 'T' | 'U' | 'V' |
     'W' | 'X' | 'Y' | 'Z' ;
LC = 'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j' | 'k' |
     'l' | 'm' | 'n' | 'o' | 'p' | 'q' | 'r' | 's' | 't' | 'u' | 'v' |
     'w' | 'x' | 'y' | 'z' ;
quotedString = "'", any - "'", "' ;
any = ? any character ? ;
ws = ? any white space character, i.e. space or tab ? ;

```

The code uses the following conventions:

left --> right

left is composed of right

A , B

A is followed by B

{ A }

A may be present one or more times



[A]

A may be present zero or one times

A | B

either A or B must be present

single quotes ''

denote literals i.e. strings that need to appear in this form in the DBC file

? A ?

A is a non-formal description

(A)

are used to group other expressions (as parentheses do in maths)

6.10.2.6.4.3. I-Pdu Multiplexer Configuration

The DBC file format defines Multiplexer configuration information differently than the AUTOSAR system template does. Multiplexer selector fields in the DBC file format are defined as a special kind of signals, so-called *mode signals*.

If a PDU contains such a mode signal, it usually also contains signals depending on this mode signal. Such signals are called *mode dependent signals* and are only transmitted if the mode signal contains a certain value.

The AUTOSAR system template defines so called *DynamicPart* bit ranges which depend on the value of the selector field and *StaticPart* bit ranges which are independent from the selector field value.

There are many possible mappings between the DBC and the AUTOSAR I-Pdu Multiplexer configuration representations. For instance, a straight forward mapping between the DBC signals and AUTOSAR bit ranges would be to map a mode signal onto the selector field, to map any mode dependent signal onto a single *DynamicPart*, and to map all mode independent signals to one single *StaticPart*. This mapping very likely leads to suboptimal Ipdum configurations with many *DynamicParts* involving unnecessary memory and runtime overhead.

The DBC Importer provides an option with which you can specify how multiplexer selector fields, *StaticParts*, and *DynamicParts* shall be configured. Three modes are available:

- ▶ **Do not map selector fields to Com Signals:** In this mode, the multiplexer selector field is *not* included as signal in any static or dynamic SignalIPdu.
- ▶ **Map selector fields to Com Signals:** In this mode, the multiplexer selector field is included as *mode dependent signal* in every dynamic SignalIPdu. Since the value of this signal must not be changed, the use is limited.
- ▶ **AUTOSAR compliant mapping:** As in the previous mode, this mode includes the multiplexer selector field as *mode dependent signal*. Moreover, the bit ranges of the static and dynamic parts are not *stuffed*, i.e. the bit offsets of the bit ranges in the Multiplexed PDUs and in the contained static and dynamic SignalIPdus are identical. Additionally all dynamic SignalIPdus of belonging to the same Multiplexed PDU own identical bit ranges.



- ▶ If you need low RAM and ROM consumption, and your `IpduM` does not need to comply to the AUTOSAR standard, select the first mode.
- ▶ If you require that the values of the selector fields of the dynamic `SignalIPdus` can be retrieved at runtime and stuffing of the bit ranges for low RAM and ROM consumption still shall be performed, select the second mode.
- ▶ If you require that the `IpduM` shall be configured in an AUTOSAR compliant way, and/or better runtime behavior at the expense of higher RAM/ROM consumption select the third mode.

The following example outlines how the DBC Importer performs the `IPDU Multiplexer` configuration mapping if the first mode (**Do not map selector fields to Com Signals**) was selected:

The CAN message defined in the example contains a number of *big endian* and *little endian* signals which either depend on the mode signal or not. The mode signal itself is assumed to lie on bits `[0..1]` of the message and is not explicitly shown. Dynamic signals depending on different values usually overlap each other; however the outlined example does not show such overlaps to keep the example simple.

	7	6	5	4	3	2	1	0
0	SL	SL	SL					
1	SL		SL	SL		SL	SL	SL
2	DL2		DL1	DL1	DL1		SL	SL
3						DB1	DB1	DB1
4	DB1	DB1	DB1				SB	SB
5	SB		DL3	DL3	DL3			
6								
7	DB2	DB2	DB2					

The first character on the label of each bit indicates whether the signal occupying the bit is either mode independent/static (S), or mode-dependent/dynamic (D). The second character describes the endianess of the occupying signal, which is either big endian (B), or little endian (L). Bits, which belong to mode-dependent signals, additionally display the mode signals value they depend on: For instance `DL3` indicates that the signal of the bit is only transmitted if the mode signals has a value of 3.

In the first step, the algorithm traverses all bits of the PDU beginning with the least significant bit of the PDU (bit 0) and ends with its most significant bit, bit 63. The algorithm assigns the signal bits either to static or to dynamic bit ranges depending on whether they are occupied by static or dynamic signals.

- ▶ A bit range starts and ends with a bit which is actually occupied by a signal.
- ▶ If a static bit follows a dynamic bit, the current dynamic bit range ends, and a new static bit range begins.
- ▶ If a dynamic bit follows a static bit, the current static bit range ends, and a new dynamic bit range begins.



- ▶ A sequence of unused bits is added to a static or dynamic part if it is not longer than seven bits and surrounded by two static or dynamic bits. In all other cases, the first empty bit succeeding a bit range confines it, as well as the last empty bit before a bit range.

7	6	5	4	3	2	1	0
SL	SL	SL					
SL		SL	SL		SL	SL	SL
DL2		DL1	DL1	DL1		SL	SL
					DL1	DL1	DL1
DB1	DB1	DB1				SB	SB
SB		DL3	DL3	DL3			
DB2	DB2	DB2					

In the second step, the algorithm groups the bit ranges according to the values the mode signal may occupy. If a bit range does not contain any bit depending on a certain mode signal value, the bit range is removed from the value group. The following figure displays the static bit ranges of the CAN message and the dynamic bit ranges assigned to value 1 of the mode signal.

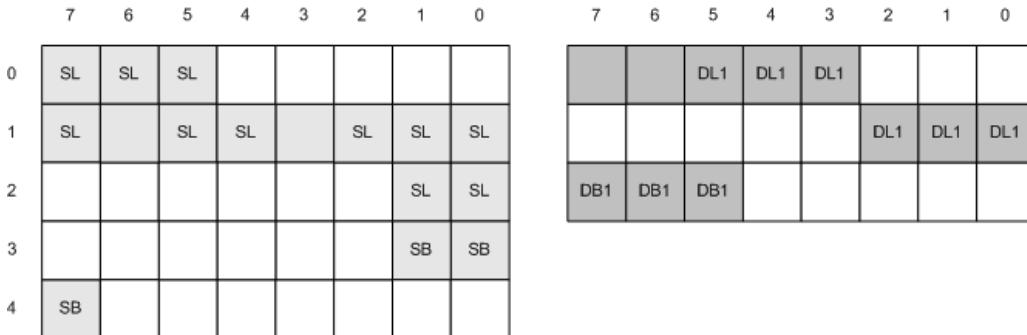
7	6	5	4	3	2	1	0
0	SL	SL	SL				
1	SL		SL	SL		SL	SL
2						SL	SL
3							
4						SB	SB
5	SB						
6							
7							

7	6	5	4	3	2	1	0
			DL1	DL1	DL1		
						DL1	DL1
							DL1
DB1	DB1	DB1					

The last step performs the byte stuffing for the static bit range set and all dynamic bit range sets. If there is an interval $[n*8 .. (n + 1)*8 - 1]$ which does not occupy any bit range, the interval is removed completely, i.e. all following bit ranges are shifted up by one byte. If remaining full bytes of a CAN message do not occupy any bit range, they are also truncated. The next figure displays the result of the byte stuffing step.



The static part on the left hand side makes up the static `SignalIPdu`, which is five bytes long. The dynamic part on the right hand side makes up the dynamic `SignalIPdu` depending on the switch value 1.



The tables below show the properties of the `StaticPart` and entities which are the result of the transformation.

<code>startPosition</code>	<code>bitLength</code>	<code>pduMultiplexerDataPartByteOrder</code>
5	13	MOSTSIGNIFICANTBYTELAST
24	2	MOSTSIGNIFICANTBYTELAST
39	1	MOSTSIGNIFICANTBYTELAST

The tables below show the properties of the `DynamicPart` entities depending on the switch value 1 which are the result of the transformation.

<code>startPosition</code>	<code>bitLength</code>	<code>pduMultiplexerDataPartByteOrder</code>	<code>selector-FieldCode</code>
3	8	MOSTSIGNIFICANTBYTELAST	1
21	3	MOSTSIGNIFICANTBYTELAST	1

Moreover, the algorithm creates one `SignalIPdu` entity for every selector field code value to which the corresponding `DynamicPart` entities refer. If one or more `StaticPart` entities have been created, a corresponding `SignalIPdu` is created to which all `StaticPart` entities refer. If any of the `StaticPart` and/or `DynamicPart` entities were shifted during the byte stuffing step, the contained signals are shifted by the same amount of bytes to ensure consistency.

In the second mode **Map selector fields to Com Signals**, the selector field code of a Multiplexer PDU is handled as an ordinary *mode dependent signal*, thus the configured example differs in two aspects:

- ▶ Every dynamic `SignalIPdu` owns a bit range occupying the bits in which the selector field resides.
- ▶ Every dynamic `SignalIPdu` additionally contains the selector field as dedicated signal.

The following paragraphs outline the layout of the static and dynamic PDUs if the third mode (**AUTOSAR compliant mapping**) is selected.

The example below displays the original Multiplexer configuration. In addition, the selector field code "SFC" is displayed occupying bits 0 and 1.



	7	6	5	4	3	2	1	0
0	SL	SL	SL				SFC	SFC
1	SL		SL	SL		SL	SL	SL
2	DL2		DL1	DL1	DL1		SL	SL
3						DB1	DB1	DB1
4	DB1	DB1	DB1				SB	SB
5	SB		DL3	DL3	DL3			
6								
7	DB2	DB2	DB2					

This mode leads to the following layout of the contained static and dynamic SignalIPdus. The static SignalIPdu is shown on the left hand side, the layout of all dynamic SignalIPdus is displayed on the right hand side:

	7	6	5	4	3	2	1	0
0	SL	SL	SL					
1	SL		SL	SL		SL	SL	SL
2							SL	SL
3								
4						SB	SB	
5	SB							

	7	6	5	4	3	2	1	0
0							SFC	SFC
1								
2	DL2		DL1	DL1	DL1			
3							DB1	DB1
4	DB1	DB1	DB1					
5			DL3	DL3	DL3			
6								
7	DB2	DB2	DB2					

The bit range stuffing step as described above is omitted which causes empty bytes in the SignalIPdus. Since the bit ranges of all dynamic parts must be identical, this import mode does not take into account whether a bit range is actually required for a dynamic SignalIPdu or not. This leads to identical layouts of all dynamic SignalIPdus. In comparison to the other import modes, this mode may lead to a significant higher amount of required RAM, especially if there are many dynamic SignalIPdus (i.e. many different selector field codes) per Multiplexed PDU.

6.10.2.7. Importing from the LDF format



EB tresos Studio provides an integrated LDF Importer to import LIN configurations from LDF files. If an LDF file is available for import, the LDF Importer simplifies and speeds up the configuration of the LIN stack.

6.10.2.7.1. Overview

The LDF Importer can read an LDF file and is able to import the data into the system model of the project.

6.10.2.7.2. LDF Importer and file format

LDF (Lin Description File) is a standardized and open file format for LIN networks. LDF files describe the nodes connected to the LIN network, frames and signals transmitted on the network, and Schedule Tables describing the transmission pattern. A LIN network contains one master node and one or more Slave nodes. The master node is the only node for which configuration data is imported since Slave nodes are not considered to be AUTOSAR nodes.

Supported LDF versions are: 2.0, 2.1, and 2.2.

A specification on the file format and on the LIN protocol itself can be obtained from <http://www.lin-subbus.org>.

6.10.2.7.3. Importing LDF configurations

The LDF Importer allows to import into system model of the project.

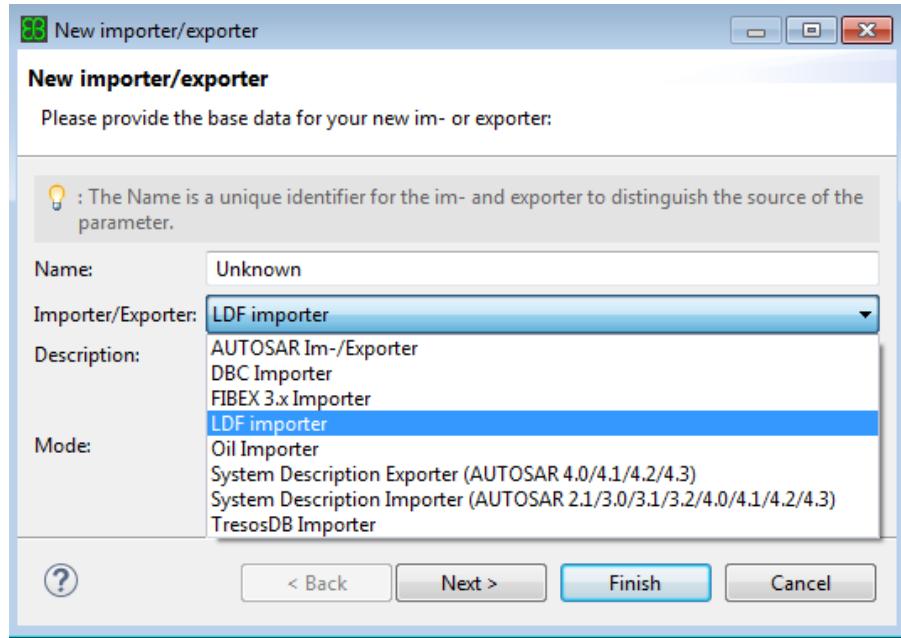
To import configurations from the LDF format:

- ▶ Refer to chapter [Section 6.10.2, “Importing and exporting configuration data”](#).
- ▶ Follow the instructions of Step 1 until you arrive at the window **Create, manage and run im- and exporters**.

In the window **Create, manage and run im- and exporters**:

- ▶ Click the **Create a new im- or exporter** button.

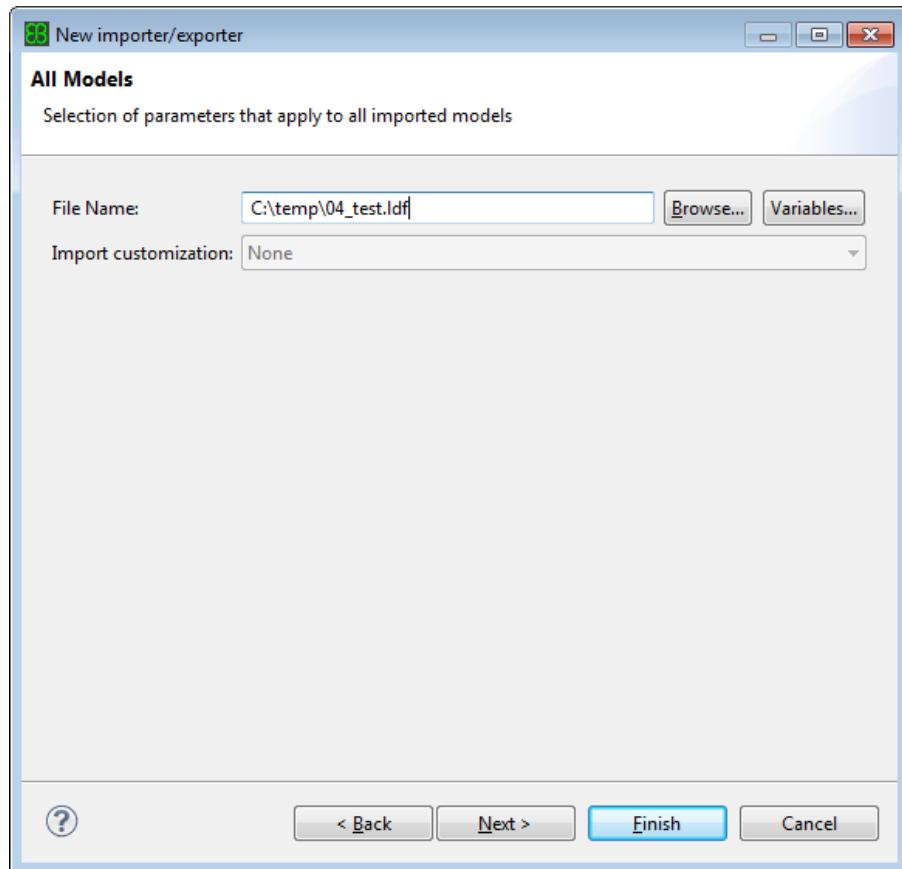
The **New Importer/Exporter** window opens up.



To enter the basic information for the new importer or exporter configuration:

- ▶ Select the **Name** text box.
- ▶ Enter a name for your new configuration.
- ▶ Select **LDF Importer** from the **Importer/Exporter** drop-down list box.
- ▶ Select the **Mode** drop-down list box.
- ▶ Select **Import only** from the drop-down list box.
- ▶ Click **Next**.

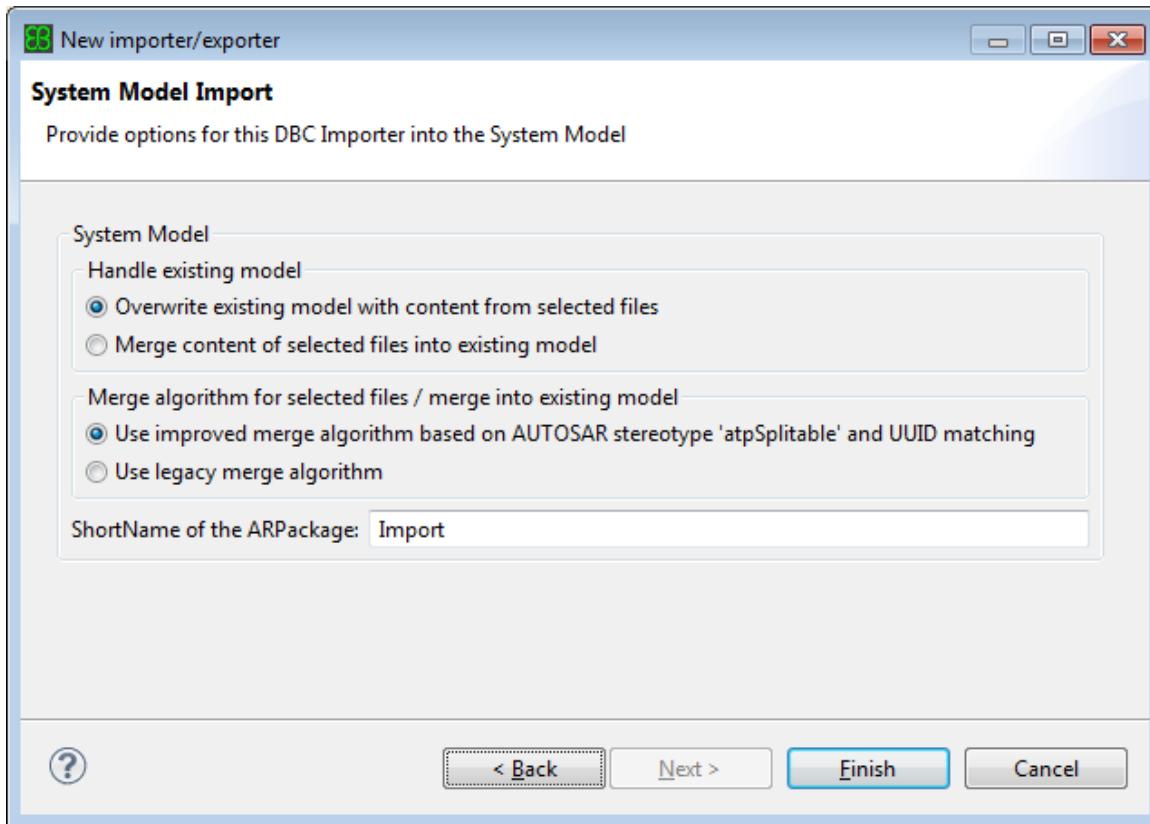
After clicking the **Next** button you enter a page, in which you may enter general parameters.



- ▶ In the **File Name** file selection entry enter the name of an LDF file. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#)
- ▶ If the **Import customization** drop-down list box is enabled, select an import customization.

The **Import customization** drop-down list box provides available import customizations registered for this kind of importer. An import customization affects the import process and modifies the imported data. In most cases this list will be empty and thus grayed out.

Press the **Next** button to switch to a page, in which you may enter parameters for the system model import.

Figure 6.163. The **System Model Import** page

- ▶ Choose how to handle the existing model:
 - ▶ To completely replace the system model in the project, select **Overwrite existing model with content from selected files**.
 - ▶ To add the system model information of the selected files to the existing model, select **Merge content of selected files into existing model**.
- ▶ Choose the merge algorithm:
 - ▶ **Use improved merge algorithm based on AUTOSAR stereotype 'atpSplittable' and UUID matching**

The AUTOSAR meta model stereotypes <splitable> (3.x) / <atpSplittable> (4.x) define which elements can be split up over several physical files. The new improved merge algorithm is based on this concept.

In addition to this, the new merge algorithm uses UUID matching to find e.g. renamed elements. For example: you imported a file into the existing project model and rename an element in the file afterwards. If the element still has the same UUID, then the merge algorithm will find the correct entity and rename it in the project model when importing the file again.

For newly created System Description importer, this option is set as default.

- ▶ **Use legacy merge algorithm**



If you want to get the same result as in previous EB tresos Studio releases, then select the legacy merge algorithm.

- ▶ To select the AUTOSAR short-name of the AUTOSAR AR-PACKAGE into which the system model of the source file is to be imported, enter the AUTOSAR short-name into the **ShortName of the ARPackage** text box.

To save the importer configuration:

- ▶ Click **Finish**.

This saves your importer configuration and closes the **LDF Importer Options** window.

To run the LDF Importer, select the importer name you just created on the left side of the **Create, manage, and run im- and exporters** window.

- ▶ Click **Run Importer**.

The LDF configuration is being imported.

6.10.2.8. Importing from the Fibex format

6.10.2.8.1. Overview

EB tresos Studio provides an integrated Fibex Importer to import configurations from Fibex files.

The EB tresos Studio Fibex Importer is able to read a Fibex file and fill in the system model of a project.

Supported Fibex versions are: 3.0.0, 3.1.0, 3.1.1. For detailed background information see the Fibex specification available at <http://www.asam.net>.

6.10.2.8.2. Background information

6.10.2.8.2.1. Limitations

Known restrictions/limitations:

- ▶ Only FlexRay and CAN clusters are imported.
- ▶ When importing Fibex 3.x files, the Fibex Importer processes only the first **TIMING** parameter of each type (**CYCLIC**, **EVENT-CONTROLLED**, **REQUEST-CONTROLLED**) of a given PDU.



- ▶ For Fibex 3.x the schema has been extended to allow multiple **SEGMENT-POSITIONS** within a **STATIC-PART/DYNAMIC-PART**. Also, the optional tag **BIT-POSITION-DEMUX** has been added as child of a **SEGMENT-POSITION**.

These modifications do not affect the validity of existing Fibex 3.x instances.

6.10.2.8.2.2. EndToEndProtection related information

If the option to import **EndToEndProtection** related information is not disabled during import (see [Section 6.10.2.8.3, “Importing Fibex configurations”](#) or [Section 6.12.5.1.2, “System properties”](#) for the command line), additional entities are created for each E2E-protected PDU in the system model.

A PDU is considered E2E-protected if the following conditions hold:

- ▶ It contains a CRC signal. A signal is considered a CRC signal if **SIGNAL-TYPE/TYPE** exists and equals **CHECKSUM**, or if **SIGNAL-TYPE/TYPE** does not exist and **DESC/@TYPE** equals **CRC-ID**.
- ▶ The **CODED-TYPE/BIT-LENGTH** of the **CODING** referenced by the CRC signal is set to 8.
- ▶ One of the following combinations of values applies to the **SIGNAL-INSTANCE** that maps the CRC signal to the PDU:

CLUSTER's BIT-COUNTING-POLICY	BIT-POSITION	IS-HIGH-LOW-BYTE-ORDER
SAWTOOTH	0	false
SAWTOOTH	7	true
MONOTONE	7	false
MONOTONE	0	true

The following entities are created:

6.10.2.8.2.2.1. END-TO-END-PROTECTION-SET

A single **END-TO-END-PROTECTION-SET** is always created and added to the package **ENDTOENDPROTECTIONSETS**. **SHORT-NAME** is set to **E2EProtSet**.

6.10.2.8.2.2.2. END-TO-END-PROTECTION

One **END-TO-END-PROTECTION** element is created per E2E-protected PDU. It is a child element of the single **END-TO-END-PROTECTION-SET** and has the following parameters set:

SHORT-NAME is set to the **SHORT-NAME** of the PDU.



END-TO-END-PROFILE/CATEGORY **is set to** PROFILE _01.

END-TO-END-PROFILE/COUNTER-OFFSET **is set to** 8.

END-TO-END-PROFILE/CRC-OFFSET **is set to** 0.

END-TO-END-PROFILE/DATA-ID-MODE **is set to** 0.

END-TO-END-PROFILE/DATA-IDS/DATA-ID **is set to** SIGNAL-TYPE/ATTRIBUTES/ATTRIBUTE[@name = 'CRC-ID'] of the CRC signal.

END-TO-END-PROFILE/DATA-LENGTH **is set to** BYTE-LENGTH of the PDU, converted to bits.

END-TO-END-PROFILE/MAX-DELTA-COUNTER-INIT **is set to** 14.

6.10.2.8.2.2.1. END-TO-END-PROTECTION-I-SIGNAL-I-PDU

One END-TO-END-PROTECTION-I-SIGNAL-I-PDU element is created per E2E-protected PDU. It is a child element of its corresponding END-TO-END-PROTECTION element and has the following parameters set:

DATA-OFFSET **is set to** 0.

I-SIGNAL-GROUP-REF references the I-SIGNAL-GROUP created for the PDU, see [Section 6.10.2.8.2.2.3, "I-SIGNAL-GROUP"](#).

I-SIGNAL-I-PDU-REF references the E2E-protected PDU.

6.10.2.8.2.2.3. I-SIGNAL-GROUP

One I-SIGNAL-GROUP element is created per E2E-protected PDU. It is added to the ISIGNALS package and has the following parameters set:

SHORT-NAME **is set to** E2ESG_<name of PDU>.

I-SIGNAL-REFS: each SIGNAL of the E2E-protected PDU is referenced via a I-SIGNAL-REF.

SYSTEM-SIGNAL-GROUP-REF references the SYSTEM-SIGNAL-GROUP (see [Section 6.10.2.8.2.2.4, "SYSTEM-SIGNAL-GROUP"](#)).

6.10.2.8.2.2.4. SYSTEM-SIGNAL-GROUP

One SYSTEM-SIGNAL-GROUP element is created per E2E-protected PDU. It is added to the SYSSIGNALS package and has the following parameters set:



SHORT-NAME is set to E2ESG_<name of PDU>.

SYSTEM-SIGNAL-REFS: each SYSTEM-SIGNAL of the E2E-protected PDU is referenced via a SYSTEM-SIGNAL-REF.

6.10.2.8.2.2.5. I-SIGNAL-TRIGGERING

One I-SIGNAL-TRIGGERING is created for the I-SIGNAL-GROUP. I-SIGNAL-PORTS and I-SIGNAL-PORT-REFS are created in the same way as for all signals of the E2E-protected PDU, see [Section 6.10.2.8.2.2.6, "I-SIGNAL-PORTs".](#)

6.10.2.8.2.2.6. I-SIGNAL-PORTs

I-SIGNAL-PORTS and I-SIGNAL-PORT-REFS are created for the I-SIGNAL-GROUP and all contained signals so that they are sent/received via the same COMMUNICATION-CONNECTORS as the E2E-protected PDU.

6.10.2.8.3. Importing Fibex configurations

The Fibex Importer enables importing configurations into the system model of the project.

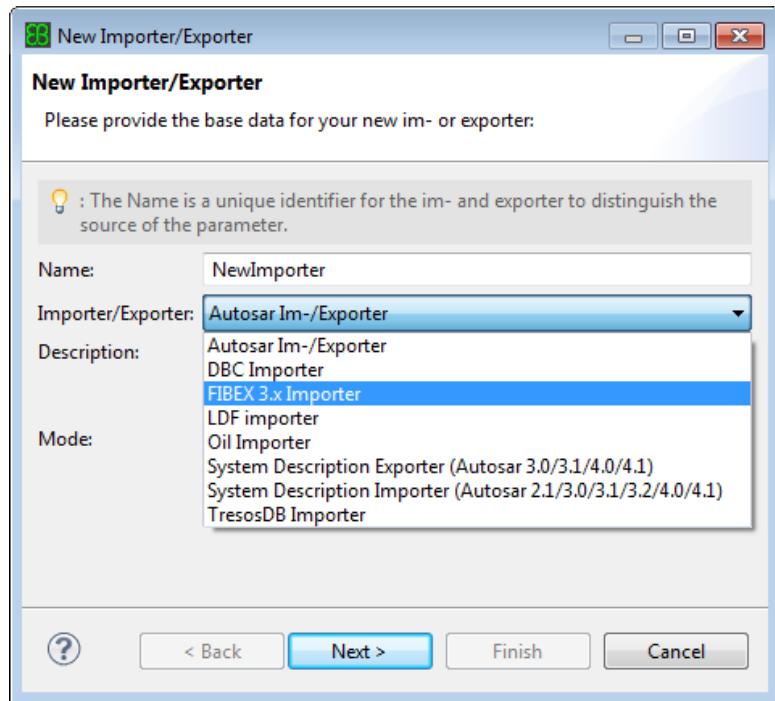
To import configurations from the Fibex format:

- ▶ Refer to [Section 6.10.2, "Importing and exporting configuration data".](#)
- ▶ Follow the instructions of Step 1 until you arrive at the window **Create, manage and run im- and exporters.**

In the window **Create, manage and run im- and exporters**:

- ▶ Click the **Create a new im- or exporter** button.

The **New Importer/Exporter** window opens up.



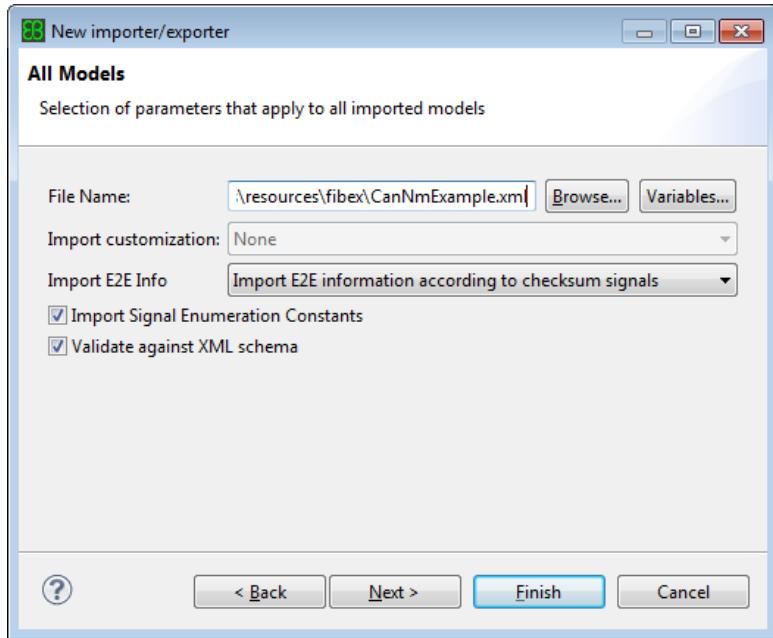
To enter the basic information for the new importer or exporter configuration:

- ▶ Select the **Name** text box.
- ▶ Enter a name for your new configuration.
- ▶ Select **Fibex 3.x Importer** from the **Importer/Exporter** drop-down listbox.
- ▶ Select **Import only** from the **Mode** drop-down listbox.

The Fibex Importer can only be used to import files. Exporting to Fibex format is not possible.

- ▶ Click **Next**.

On the following page generic information can be entered.



- ▶ Use the **File Name** file selection entry to select the Fibex file that should be imported. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#)
- ▶ If the **Import customization** drop-down list box is enabled, select an import customization.

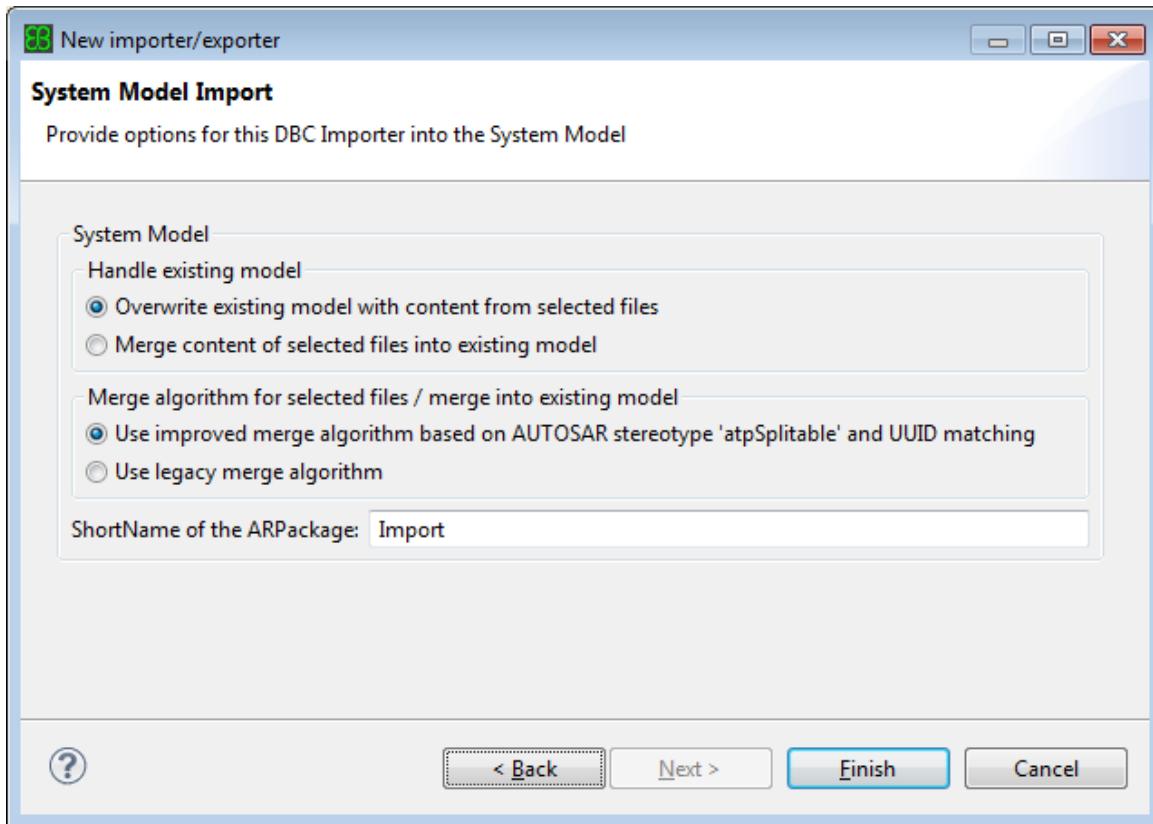
The **Import customization** drop-down list box provides available import customizations registered for this kind of importer. An import customization affects the import process and modifies the imported data. In most cases this list will be empty and thus grayed out.

- ▶ To import COMPU-CONST elements from the Fibex file, select the check box **Import Signal Enumeration Constants**.

For background information on this option see [Section 6.10.2.6.3, “Importing DBC configurations”](#).

- ▶ If you do not want to import EndToEndProtection related information, select **Do not import E2E information** from the **Import E2E Info** drop-down listbox.
- ▶ If parser errors occur during an import, you can turn off the XML validation by deselecting the check box **Validate against XML schema**.

Click the **Next** button to switch to a page where parameters for the system model import can be entered.

Figure 6.164. The **System Model Import** page

- ▶ Choose how to handle the existing model:
 - ▶ To completely replace the system model in the project, select **Overwrite existing model with content from selected files**.
 - ▶ To add the system model information of the selected files to the existing model, select **Merge content of selected files into existing model**.
- ▶ Choose the merge algorithm:
 - ▶ **Use improved merge algorithm based on AUTOSAR stereotype 'atpSplittable' and UUID matching**

The AUTOSAR meta model stereotypes <splitable> (3.x) / <atpSplittable> (4.x) define which elements can be split up over several physical files. The new improved merge algorithm is based on this concept.

In addition to this, the new merge algorithm uses UUID matching to find e.g. renamed elements. For example: you imported a file into the existing project model and rename an element in the file afterwards. If the element still has the same UUID, then the merge algorithm will find the correct entity and rename it in the project model when importing the file again.

For newly created System Description importer, this option is set as default.

- ▶ **Use legacy merge algorithm**



If you want to get the same result as in previous EB tresos Studio releases, then select the legacy merge algorithm.

- ▶ To select the AUTOSAR short-name of the AUTOSAR AR-PACKAGE into which the system model of the source file is to be imported, enter the AUTOSAR short-name into the **ShortName of the ARPackage** text box.

To save the importer configuration:

- ▶ Click **Finish**.

This saves your importer configuration and closes the **FIBEX Importer Options** window.

- ▶ To run the Fibex Importer, select the importer name you just created on the left side of the **Create, manage, and run im- and exporters** window.
- ▶ Click **Run Importer**.

The Fibex configuration is being imported.

6.10.2.9. Importing configurations from the TDB format

6.10.2.9.1. Overview

EB tresos Studio provides an integrated TDB Importer to import a system model from TDB files into the project.

6.10.2.9.2. Importing TDB configurations

The TDB Importer enables importing into system model of the project.

To import configurations from the TDB format:

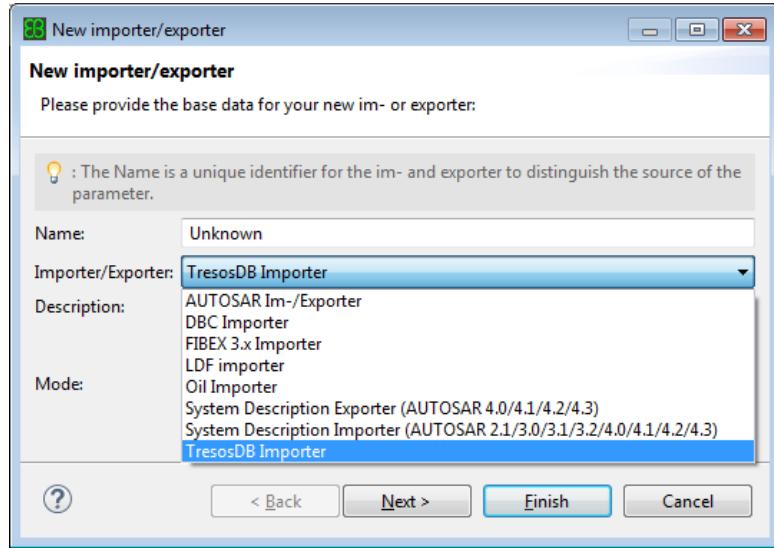
- ▶ Refer to [Section 6.10.2, “Importing and exporting configuration data”](#).
- ▶ Follow the instructions of Step 1 until you arrive at the window **Create, manage and run im- and exporters**.

In the window **Create, manage and run im- and exporters**:

- ▶ Click the **Create a new im- or exporter** button.



The **New Importer/Exporter** window opens up.



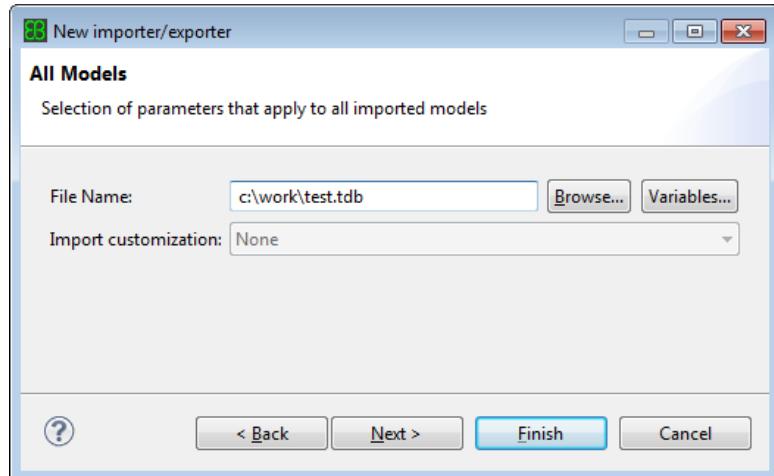
To enter the basic information for the new importer or exporter configuration:

- ▶ Select the **Name** text box.
- ▶ Enter a name for your new configuration.
- ▶ Select **TDB Importer** from the **Importer/Exporter** drop-down list box.
- ▶ Select **Import only** from the **Mode** drop-down list box.

The TDB Importer can only be used to import files. Exporting to TDB format is not possible.

- ▶ Click **Next**.

After pressing the **Next** button, you may enter general information.



- ▶ Use the **File Name** file selection entry to select the TDB file that you would like to import. When you click the **Variables** button, the standard Eclipse **Select Variable** dialog appears that lets you choose



either a predefined variable or an own defined variable. For more information about path variables, see [Section 6.10.2.2.3, "Using path variables"](#)

- ▶ If the **Import customization** drop-down list box is enabled, select an import customization.

The **Import customization** drop-down list box provides available import customizations registered for this kind of importer. An import customization affects the import process and modifies the imported data. In most cases this list will be empty and thus grayed out.

Click the **Next** button to switch to a page where parameters for the system model import can be entered.

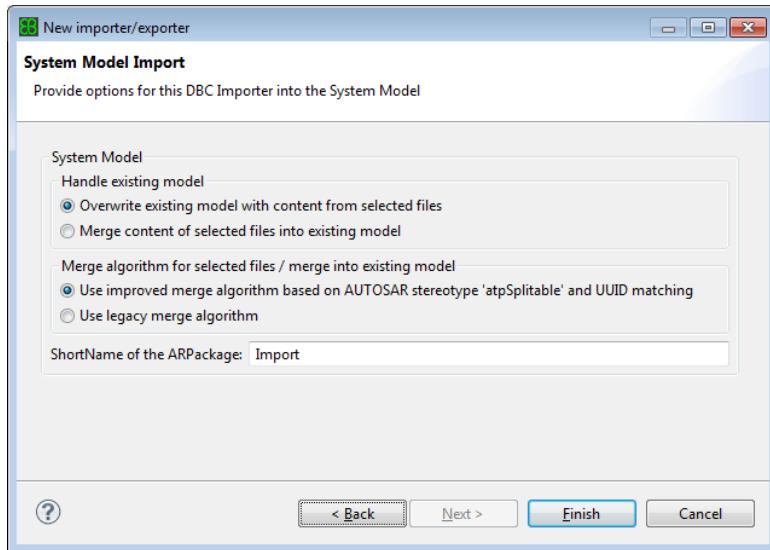


Figure 6.165. The **System Model Import** page

- ▶ Choose how to handle the existing model:
 - ▶ To completely replace the system model in the project, select **Overwrite existing model with content from selected files**.
 - ▶ To add the system model information of the selected files to the existing model, select **Merge content of selected files into existing model**.
- ▶ Choose the merge algorithm:
 - ▶ **Use improved merge algorithm based on AUTOSAR stereotype 'atpSplittable' and UUID matching**

The AUTOSAR meta model stereotypes `<splittable>` (3.x) / `<atpSplittable>` (4.x) define which elements can be split up over several physical files. The new improved merge algorithm is based on this concept.

In addition to this, the new merge algorithm uses UUID matching to find e.g. renamed elements. For example: you imported a file into the existing project model and rename an element in the file afterwards. If the element still has the same UUID, then the merge algorithm will find the correct entity and rename it in the project model when importing the file again.



For newly created System Description importer, this option is set as default.

► **Use legacy merge algorithm**

If you want to get the same result as in previous EB tresos Studio releases, then select the legacy merge algorithm.

- To select the AUTOSAR short-name of the AUTOSAR AR-PACKAGE into which the system model of the source file is to be imported, enter the AUTOSAR short-name into the **ShortName of the ARPackage** text box.

To save the importer configuration:

- Click **Finish**.

This saves your importer configuration and closes the **TDB Importer Options** window.

- To run the TDB Importer, select the importer name you just created on the left side of the **Create, manage, and run im- and exporters** window.
- Click **Run Importer**.

The TDB configuration is being imported.

6.10.2.10. Reusing importers and exporters across projects

6.10.2.10.1. Overview

EB tresos Studio saves the importer configurations in separate preferences XDM files. You can copy these individual preferences XDM files to the `prefs` folder of other projects and you can reuse them.

6.10.2.10.2. Reusing workflow files across projects

If you want to share the workflow of a project which has many importers configured with another project, you can do it easily just by copying these individual importer preference files to the other project along with the workflow.

6.10.2.10.3. Backward compatibility of importers

You can still import older projects with a single preferences XDM file for all importers. When you import a project from an earlier version of EB tresos Studio the preferences related to importers and exporters are automatically saved to individual preference files.



6.10.2.11. Importing ECU configurations from AUTOSAR system descriptions

6.10.2.11.1. Overview

EB tresos Studio provides an unattended wizard that allows to import AUTOSAR ECU configurations from current system information.

6.10.2.11.2. Background information

System information can be imported into the ECU configuration. This functionality is available in the **Create ECU Configuration** unattended wizard.

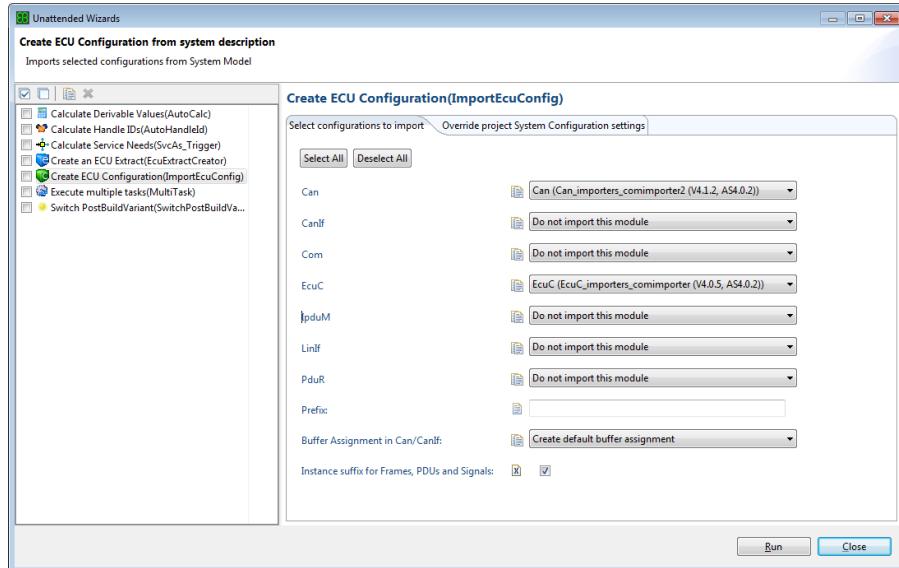


Figure 6.166. The unattended wizard to **Create ECU Configuration**

The **Create ECU Configuration** wizard offers several advantages:

- ▶ With this wizard it is possible to first import all system data from several files in different files formats and create the ECU Configuration based on the whole system model afterwards.
- ▶ The step to create the ECU Configuration can now be started also from the commandline.
- ▶ The creation of the ECU Configuration can now be triggered from a workflow.
- ▶ The creation of the ECU Configuration can be integrated into an **Execute multiple tasks** wizard.
- ▶ You do not need to select the System and EcuInstance each time again, but the unattended wizard takes the settings from the project properties, configured at the **System Configuration** page.

The **Create ECU Configuration** wizard can only be used if:



- ▶ the project contains system data, which contains either a System which refers at least one EcuInstance OR at least one TopLevelComposition (from which to create a System and EcuInstance).
- ▶ the project contains module configurations for which an ECU Configuration import is possible.

Unless you did not already configure the paths to the System and EcuInstance for the project, by either

- ▶ specifying the path during a previous wizard run or
- ▶ in the project properties

a selection dialog will appear when opening the **Create ECU Configuration** wizard in the **Unattended Wizards** dialog

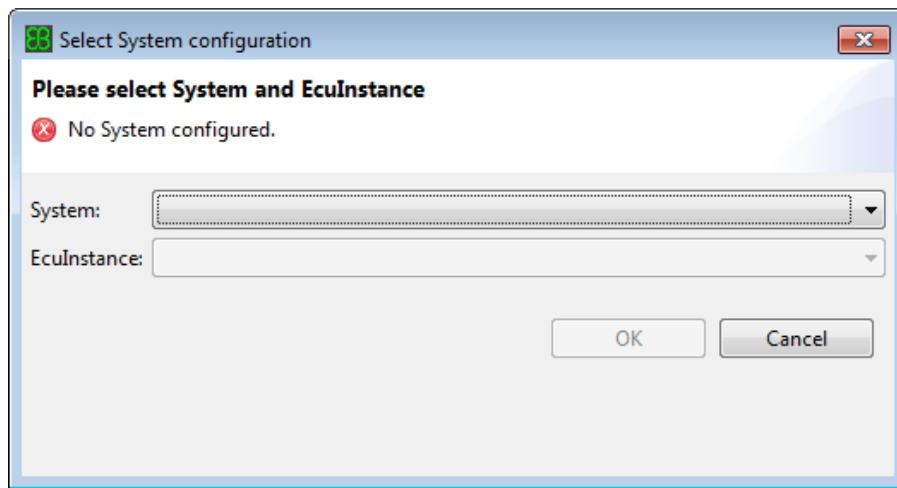


Figure 6.167. The **Select System and EcuInstance** dialog

Your selection is stored in the project settings and can be changed in the project properties dialog on the **System Configuration** page.

If there is no System available in your system model, but only a TopLevelComposition, then the following dialog will appear:

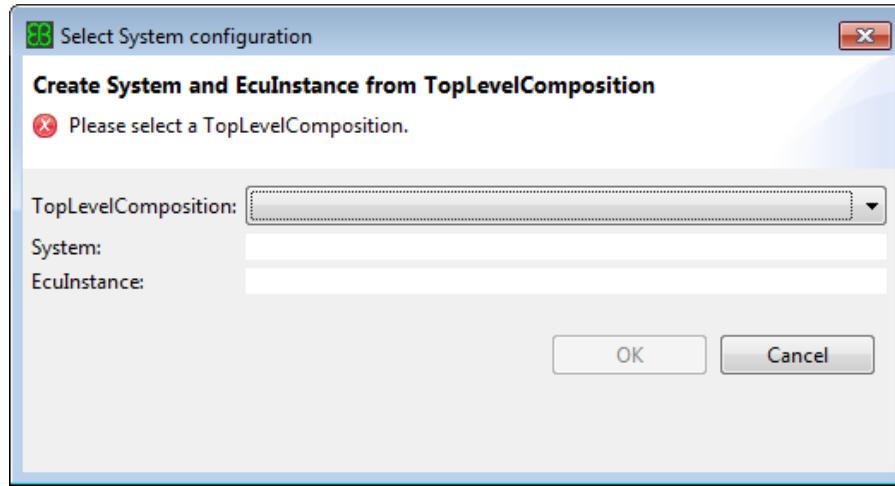


Figure 6.168. The **Create System and EcuInstance from TopLevelComposition** dialog

Here you can select one TopLevelComposition from which EB tresos Studio shall create one System and EcuInstance. You can specify the names in the dialog. The paths to the newly created System and EcuInstance are stored in the project settings.

It is also possible to open and use the **Create ECU Configuration** wizard without specifying System and EcuInstance in the project settings (e.g. by canceling the selection dialogs).

The wizard consists of two pages:

- ▶ Select configurations to import
- ▶ Override project System Configuration settings

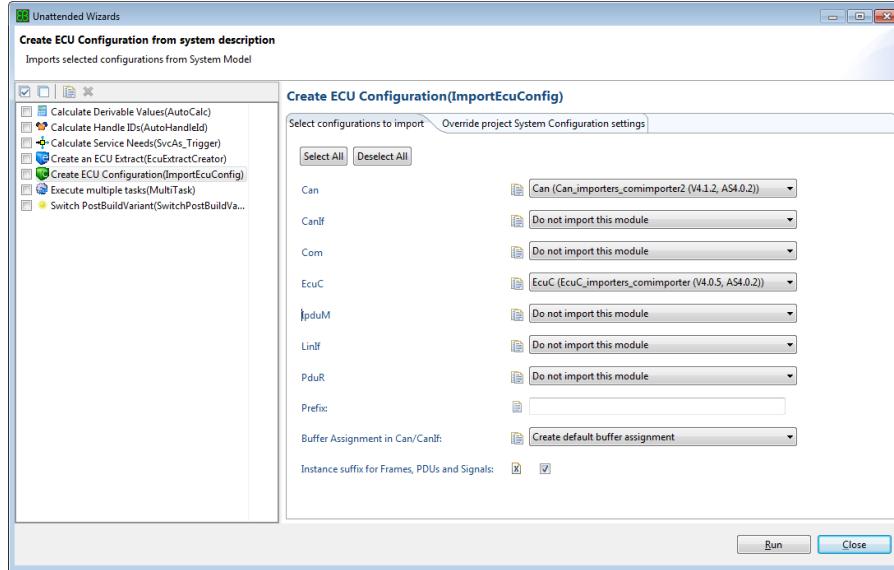
NOTE



Mapping aspects available in separate documentation

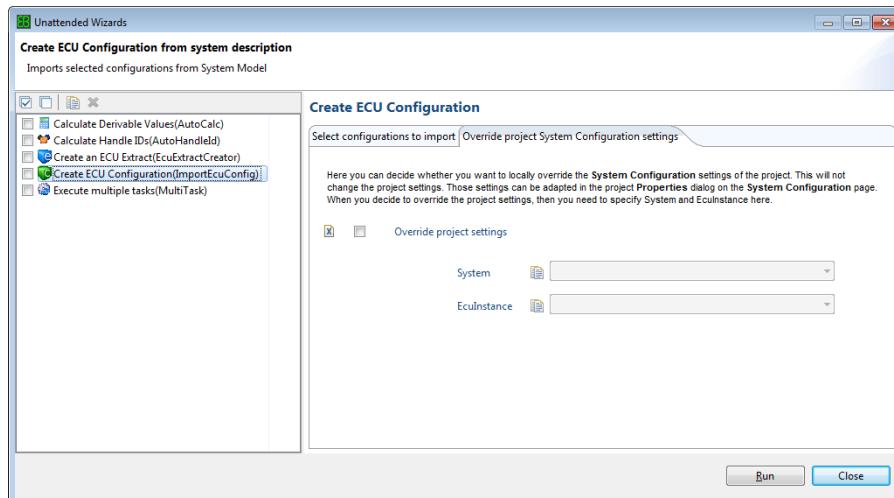
For more detailed information about e.g. naming rules or parameter mappings, see EB tresos Studio ECU Configuration Wizard documentation which is located at \$TRESOS_BASE/doc/2.0_EB_tresos_Studio with \$TRESOS_BASE being the location on your computer where you installed EB tresos Studio.

The **Select configurations to import** page displays a combobox for each module configuration of your project that the ECU Configuration Wizard supports. For the list of supported module types, see the ECU Configuration Wizard documentation. The **Buffer Assignment in Can/CanIf** combobox allows to specify whether or not to allocate hardware resources for sending and receiving PDUs in the **Can** and **CanIf** modules. By enabling the **Instance suffix for Frames, PDUs and Signals** checkbox, containers for Frames, PDUs and Signals will obtain dedicated suffixes. For detailed information on the latter two options, see the ECU Configuration Wizard documentation.

Figure 6.169. The **Select configurations to import** page

By default no module configuration is selected. You can use the **Select All** and **Deselect All** buttons to easily select or clear all module configurations. When you click the **Select All** button, the first available module configuration is automatically selected for every combo box, if you selected **Do not import this module** before.

The **Override project System Configuration settings** page may only be applicable if you imported system data from legacy file formats (e.g. dbc, fibex). Most users will not need to specify these settings, therefore those settings are displayed on a second page and the override checkbox is deselected by default.

Figure 6.170. The **Override project System Configuration settings** page

**NOTE****Does not change the project System Configuration properties**

The decision to override the project System Configuration settings and the selected System and EcuInstance are only valid for this instance of the unattended wizard. It will NOT change the global project settings!

NOTE**No System and EcuInstance specified**

If you neither specify System and EcuInstance in the project properties nor select to override the project settings, it is not possible to run the wizard.

NOTE**Selected module configuration not available**

If a previously selected module configuration is not available anymore (e.g. was removed from the project in the meantime), then this module configuration will be silently removed from the wizard configuration.

To import configuration data from the current system model:

1. Select the module configurations (e.g. EcuC,Com,...) that shall be configured in the corresponding drop-down list boxes.
2. For details on the **Prefix** text box, see the ECU Configuration Wizard documentation.
3. For details on the **Instance suffix for Frames, PDUs and Signals** check box, see the ECU Configuration Wizard documentation.
4. In the **Buffer Assignment in Can/CanIf** drop-down list box, select the buffer assignment policy for the imported Can and CanIf configuration. For detailed information, see the ECU Configuration Wizard documentation.
5. To start the import click on **Run** button or run the wizard from Unattended wizard menu.

NOTE**Import errors**

Problems discovered during the import will be shown in the errors view.

NOTE**Names for created containers**

During the import a lot of containers for e.g. messages or signals are created. For details about the created names, see the ECU Configuration Wizard documentation.

6.11. Creating an ECU Extract

**NOTE****AUTOSAR 3.2 and newer**

The following information applies only for AUTOSAR 3.2 and newer.

In this chapter you learn how you can create an ECU Extract and some details about how EB tresos Studio deals with short-names and especially short-name conflicts during ECU Extract creation.

6.11.1. The unattended wizard Create an ECU Extract(EcuExtractCreator)

You create an ECU Extract by running the unattended wizard **Create an ECU Extract(EcuExtractCreator)**. For details on how to run it, see [Section 6.8.10.3, “Running unattended wizards”](#).

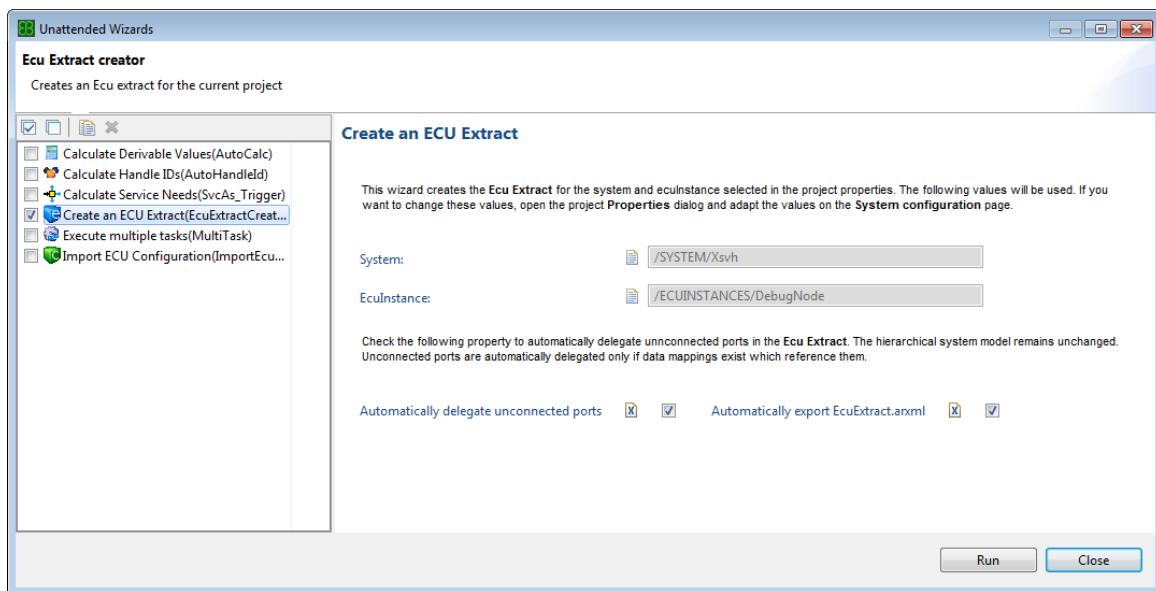


Figure 6.171. Unattended wizard for ECU Extract creation

The **EcuExtractCreator** does not require a specific configuration of its own, but uses the `System` and `EcuInstance` configured in the project properties instead to create the ECU Extract. If there are no valid values configured yet, EB tresos Studio prompts you with a dialog where you can select the `System` and `EcuInstance`. If no `System` and `EcuInstance` are available in the system data but just a `TopLevelComposition`, you can also create a `System` and `EcuInstance` from the `TopLevelComposition`.

For more information on how to configure the `System` and `EcuInstance` in the project properties, see [Section 6.3.7.5, “Editing the System Configuration settings”](#).

If the **Automatically delegate unconnected ports** property is set, all unconnected ports which have a data mapping attached are automatically delegated in the ECU Extract. The hierarchical model remains unchanged.



If the **Automatically export EcuExtract.arxml** property is set, the generated EcuExtract will be exported into the `EcuExtract.arxml` in the `systemmod` directory of the current project.

Once the `System` and `EcuInstance` are configured, you can also run the **EcuExtractCreator** from the command line: `tresos_cmd.bat autoconfigure <projectname> EcuExtractCreator`

6.11.2. Fixed AUTOSAR short-name paths

EB tresos Studio creates the ECU Extract within the following AUTOSAR short-name paths:

AUTOSAR short-name path	Content
<code>/EcuExtract</code>	Path of the Package that contains the ECU Extract for this ECU.
<code>/EcuExtract/TopLevelComposition</code>	Path of the top-level <code>CompositionSwComponentType</code> which contains the ECU flat view.
<code>/EcuExtract/<ECU_ID></code>	Path of the ECU Extract System with the category set to <code>ECU_EXTRACT</code> . The <code><ECU_ID></code> is taken over from the ECU configuration project. For information about the ECU ID, see Section 6.3.1.3, "Step 3: Configuring project properties" .

NOTE



<ECU ID> is adapted to short-name syntax restrictions

To comply to the restrictions for an AUTOSAR short-name, EB tresos Studio applies the following conversions to the ECU ID if necessary:

1. Replace every non-allowed character (e.g. blank) by an `_` underscore.
2. Replace consecutive underscores by single ones.
3. Trim from left to the maximum allowed length of 128 chars.
4. Replace the first character with `A` if it is not an alphabetic character.

6.11.3. SwComponentPrototype short-names within the ECU flat map

When EB tresos Studio creates the ECU flat map, it applies the following method to ensure unique short-names for elements of `SwComponentPrototype`:

1. Take the original short-names from the hierarchical model.
2. If short-name clashes occur, EB tresos Studio takes the following steps to resolve them:



- ▶ Alphabetically sort the `SwComponentPrototype` elements of the ECU flat view by the concatenated short-name paths of the original context `SwComponentPrototype`. These short-name paths are separated by ; and have the original short-name.
- ▶ Append _<index> which starts with _1 for these clashing short-names in the described order.
- ▶ If the maximum length for a short-name is exceeded by appending an index, the required number of characters is cut off from the end of the original short-name before the index is appended.

Consider the following structure in the hierarchical model as an example:

```

<AR-PACKAGE>
  <SHORT-NAME>CompPckg</SHORT-NAME>
  <ELEMENTS>
    <COMPOSITION-SW-COMPONENT-TYPE>
      <SHORT-NAME>ComposAB</SHORT-NAME>
      ...
      <COMPONENTS>
        <SW-COMPONENT-PROTOTYPE>
          <SHORT-NAME>compA</SHORT-NAME>
          <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/CompPckg/AtomicA</TYPE-TREF>
        </SW-COMPONENT-PROTOTYPE>
        <SW-COMPONENT-PROTOTYPE>
          <SHORT-NAME>compG</SHORT-NAME>
          <TYPE-TREF DEST="COMPOSITION-SW-COMPONENT-TYPE">/CompPckg/ComposABCD</TYPE-TREF>
        </SW-COMPONENT-PROTOTYPE>
        ...
      </COMPONENTS>
    </COMPOSITION-SW-COMPONENT-TYPE>
    <COMPOSITION-SW-COMPONENT-TYPE>
      <SHORT-NAME>ComposABCD</SHORT-NAME>
      ...
      <COMPONENTS>
        <SW-COMPONENT-PROTOTYPE>
          <SHORT-NAME>compA</SHORT-NAME>
          <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/CompPckg/AtomicA</TYPE-TREF>
        </SW-COMPONENT-PROTOTYPE>
        ...
      </COMPONENTS>
    </COMPOSITION-SW-COMPONENT-TYPE>
    <APPLICATION-SW-COMPONENT-TYPE>
      <SHORT-NAME>AtomicA</SHORT-NAME>
      ...
  </ELEMENTS>
</AR-PACKAGE>
```



```

<AR-PACKAGE>
  <SHORT-NAME>SysPckg</SHORT-NAME>
  <ELEMENTS>
    <SYSTEM>
      <SHORT-NAME>System</SHORT-NAME>
      <CATEGORY>SYSTEM_DESCRIPTION</CATEGORY>
      <ROOT-SOFTWARE-COMPOSITIONS>
        <ROOT-SW-COMPOSITION-PROTOTYPE>
          <SHORT-NAME>RootSwCompositionPrototype</SHORT-NAME>
          <SOFTWARE-COMPOSITION-TREF>
            <TYPE-TREF DEST="COMPOSITION-SW-COMPONENT-TYPE">/SysPckg/RootComp
          </SOFTWARE-COMPOSITION-TREF>
        </ROOT-SW-COMPOSITION-PROTOTYPE>
      </ROOT-SOFTWARE-COMPOSITIONS>
      ...
    </SYSTEM>
    <COMPOSITION-SW-COMPONENT-TYPE>
      <SHORT-NAME>RootComp</SHORT-NAME>
      ...
      <COMPONENTS>
        <SW-COMPONENT-PROTOTYPE>
          <SHORT-NAME>CompAB</SHORT-NAME>
          <TYPE-TREF DEST="COMPOSITION-SW-COMPONENT-TYPE">/CompPckg/ComposAB</TYPE-TREF>
        </SW-COMPONENT-PROTOTYPE>
        <SW-COMPONENT-PROTOTYPE>
          <SHORT-NAME>CompAB2</SHORT-NAME>
          <TYPE-TREF DEST="COMPOSITION-SW-COMPONENT-TYPE">/CompPckg/ComposAB</TYPE-TREF>
        </SW-COMPONENT-PROTOTYPE>
        <SW-COMPONENT-PROTOTYPE>
          <SHORT-NAME>CompABCD</SHORT-NAME>
          <TYPE-TREF DEST="COMPOSITION-SW-COMPONENT-TYPE">/CompPckg/ComposABCD</TYPE-TREF>
        </SW-COMPONENT-PROTOTYPE>
        ...
      </COMPONENTS>
      ...
    </COMPOSITION-SW-COMPONENT-TYPE>
  </ELEMENTS>
</AR-PACKAGE>

```

The short-name conflicts for the resulting `SwComponentPrototype` elements in the ECU flat view that arise from this example are therefore resolved after the sorting as in the following table:

Sort criteria (see above)	Short-name
/SysPckg/RootComp/CompAB2; /CompPckg/ComposAB/compG; compA	compA
/SysPckg/RootComp/CompAB2; compA	compA_1
/SysPckg/RootComp/CompAB; /CompPckg/ComposAB/compG; compA	compA_2
/SysPckg/RootComp/CompAB; compA	compA_3
/SysPckg/RootComp/CompABCD; compA	compA_4



6.12. Working with the command line

EB tresos Studio provides a commandline interface to trigger

- ▶ code generation and verification
 - ▶ for configuration projects (for details, see [Section 6.12.2, “Commands for project-based code generation and configuration”](#)),
 - ▶ independently from projects (for details, see [Section 6.12.3, “Commands for legacy code generation”](#)),
- ▶ module development functions (for details, see [Section 6.12.4, “Commands for module development”](#))
- ▶ file conversion functions (for details, see [Section 6.12.5, “Commands for converting and merging files”](#)), and
- ▶ execute multiple commands at once in batch mode (for details, see [Section 6.12.6, “Executing multiple commands \(batch mode\)”](#)).

6.12.1. General

6.12.1.1. Starting the commandline

To start the commandline:

- ▶ Open the Windows command prompt.
- ▶ Change into the directory, into which you have installed EB tresos Studio; e.g. C:\EB\tresos.
- ▶ Change into the bin subdirectory.
- ▶ You can execute a command by using tresos_cmd.bat; e.g. type tresos_cmd.bat --help to see a short help for the commandline options.

6.12.1.2. The commandline syntax

An example of a commandline is:

```
tresos_cmd.bat -Dinfo=false -data c:\workspace generate myproject
```

The EB tresos Studio commandline has a general syntax, which is used throughout all available commands. Each command is composed of specific blocks. Some of the blocks may be left out, but the overall order of the blocks is important for the command to be recognized properly. A commandline consists of the following blocks:



```
tresos_cmd.bat [<system_property>...] [-data <workspace_dir>]
<command>
```

The position of these blocks is important.

Block	Description
<system_property>	<p>System properties control the behavior of the command.</p> <p>Some system properties are understood by every command and some are only valid for a specific command. The list of system properties understood by all commands is available in Section 6.12.1.3, "System properties common to all commands".</p> <p>System properties are always the first block on the commandline. You may specify as many system properties as you want, or none at all.</p>
<workspace_dir>	<p>This block lets you control on which workspace the commandline operates. You may omit this block. Details of the contents of the block are available in Section 6.12.1.4, "Specifying the workspace on the commandline".</p>
<command>	<p>This block tells EB tresos Studio what to do, e.g. to generate code.</p> <p>This block consists of the command itself and additional parameters that control what the command does. The other sections of this chapter describe all the commands that are available and their parameters.</p>

6.12.1.3. System properties common to all commands

The following system properties can be used on all the commands listed below:

System property	Description
-Dinfo=false	Disables printing of any information messages.
-Dwarning=false	Disables printing of any warning messages.
-Dmsg<code>=false	Disables printing of any information or warning messages with the given numerical message code <code>.
-Ddebug.packages=<Package>:<Package>:...	Enables Log4J debugging for specific Java packages.
-Dvalidate=<boolean>	Enables or disables the validation of read and written files (e.g. against an AUTOSAR XSD schema-file).



System property	Description
	<p>If this property is not set, the behavior is context dependent: by default read files are not validated while written files are validated.</p> <p>NOTE  AUTOSAR configuration files written by a code generator are never validated, even if this property is set to <code>true</code>.</p>
<code>-Dbase.path=<directory></code>	The base directory for the command.
<code>-DrestrictShortName=true</code>	Enables the preference Use AUTOSAR-compliant maximum SHORT-NAME length (128 chars for AUTOSAR releases starting with 3.2, 32 chars for earlier releases) .

An example for the use of system properties is:

```
tresos_cmd.bat -Dinfo=false -Dmsg1234=false generate myproject
```

This code example generates code for a project called *myproject*.

When the commandline is executed, the output does not contain any informational messages (`-Dinfo=false`) and additionally the output of the message with the code `1234` is also suppressed (`-Dmsg1234=false`).

6.12.1.4. Specifying the workspace on the commandline

By default EB tresos Studio loads the workspace used in the last program run (for an explanation of the concept of workspaces, see [Section 3.5, “EB tresos Studio projects and workspaces”](#)).

When you execute commands on the commandline, you may want EB tresos Studio to use a specific workspace directory and not the one you used last.

- ▶ To specify another workspace directory, use the `-data` parameter.

This parameter changes the workspace directory for the current program run only. If you execute another command, or start the GUI, the workspace used previously is reused.

The `-data` option must be placed behind `tresos_cmd.bat`, after any system properties, and takes a directory as argument.

The following is an example for how you may change the workspace:

```
tresos_cmd.bat -Dinfo=false -data c:\workspace generate myproject
```



This code generates code for a project called *myproject*. As workspace, the directory `c:\workspace` is used. This implies that the project *myproject* is part of that workspace. When the commandline is executed, the output does not contain any informational messages.

TIP

You may also omit the system property `-Dinfo=false`. It is only present in the code example above to demonstrate the correct order of the commandline parameters.

The following example does the same as the example above, but uses the workspace which was used last:

```
tresos_cmd.bat -Dinfo=false generate myproject
```

6.12.2. Commands for project-based code generation and configuration

Use the project-based mode of the command line to generate code, verify configurations, or to change the configuration for an existing EB tresos Studio project stored in an EB tresos Studio workspace (see [Section 3.5, "EB tresos Studio projects and workspaces"](#)).

6.12.2.1. Importing projects

To access projects for generating code or to modify the ECU configuration, a project must be part of the current workspace. To get a project into the workspace, either create it in the workspace or import it into the workspace.

To support batch processing the EB tresos Studio command line allows to import projects found in the file system into the current workspace.

6.12.2.1.1. Syntax

To import a project into the workspace, use the command:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]  
importProject [-c] <project path>...
```



6.12.2.1.2. System properties

This command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.1.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:

Parameter	Description
<code>-c</code>	If you provide the <code>-c</code> commandline switch, a copy of the project directory is added to the workspace directly beneath the workspace root directory. The directory name is the same as the project name.
<code><project_path></code>	A path to an Eclipse project (a directory with a <code>.project</code> file).

6.12.2.1.4. Example

The following code is an example of how to use the command `importProject`:

```
tresos_cmd.bat importProject -c C:\temp\myproject
```

This command imports the project located in `C:\temp\myproject` into the current workspace. The `-c` command line option instructs EB tresos Studio to copy the project directory into the directory `myproject` beneath the workspace directory.

6.12.2.2. Deleting projects

To support batch processing, the EB tresos Studio commandline allows to remove projects from the current workspace.

6.12.2.2.1. Syntax

To remove a project from the workspace, use the command:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    deleteProject [-d] <project>...
```



6.12.2.2.2. System properties

This command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.2.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:

Parameter	Description
<code>-d</code>	If you provide the <code>-d</code> commandline switch, the content of the project directory is removed from the file system.
<code><project></code>	The name of an Eclipse project located in the current workspace.

6.12.2.2.4. Example

The following code is an example of how to use the command `deleteProject`:

```
tresos_cmd.bat deleteProject -d myproject
```

This command removes the project named `myproject` from the current workspace. The `-d` commandline option instructs EB tresos Studio to delete the project directory from the file system.

6.12.2.3. Renaming projects

The EB tresos Studio commandline allows to rename projects in the current workspace.

6.12.2.3.1. Syntax

To rename one project in the workspace, use the command:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    renameProject <project> <newProject>
```



6.12.2.3.2. System properties

This command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.3.3. Parameters

In addition to the common parameters <system_property> and <workspace>, this command also recognizes the following parameters:

Parameter	Description
<project>	The name of an Eclipse project located in the current workspace.
<newProject>	The new name of the project.

6.12.2.3.4. Example

The following code is an example of how to use the command renameProject:

```
tresos_cmd.bat renameProject myproject mynewproject
```

This command changes the project name from myproject to mynewproject in the current workspace.

6.12.2.4. Executing importers configured for a project

For each project you can configure importers via the GUI. These importers allow to import or export configuration data from or to ECU configuration formats or system formats such as DBC, LDF, etc. The commandline of EB tresos Studio allows to execute these configured importers.

6.12.2.4.1. Syntax

Importers can execute in import or export mode on the commandline.

To execute an importer in import mode use:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
```



```
import <project> <importer>...
```

To execute an importer in export mode use:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    export <project> <importer>...
```

6.12.2.4.2. System properties

The importer commands only recognize the system properties common to all commands. For a list of system properties common to all commands, see [Section 6.12.1.3, “System properties common to all commands”](#).

6.12.2.4.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, the importer command also recognizes the following parameters:

Parameter	Description
<code><project></code>	The name of the configuration project on which you want the execute importers.
<code><importer></code>	The name of the importer configuration to execute. You may provide this parameter multiple times to run multiple importers. An importer configuration with the given name must exist in the project.

For further information on how to configure importers for a project see [Section 6.10, “Importing and exporting”](#).

6.12.2.4.4. Example

The following commandline is an example of how to use the command `import`:

```
tresos_cmd.bat import myproject myimporter
```

The example executes the importer `myimporter` configured in the project `myproject`.

6.12.2.5. Listing importers configured for a project

For each project you can configure importers via the GUI. These importers allow to import or export configuration data from or to ECU configuration formats or system formats such as DBC, LDF, etc. The commandline of EB tresos Studio allows to list importers configured for a project.



6.12.2.5.1. Syntax

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    listimporter <project>
```

6.12.2.5.2. System properties

The **listimporter** command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.5.3. System properties

The importer commands only recognize the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.5.4. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, the **listimporter** command also recognizes the following parameters:

Parameter	Description
<code><project></code>	The name of the configuration project for which you want the list importers.

For further information on how to configure importers for a project see [Section 6.10, “Importing and exporting”](#).

6.12.2.5.5. Example

The following commandline is an example of how to use the command `listimporter`:

```
tresos_cmd.bat listimporter myproject
```

The example lists all importers for project *myproject*.

6.12.2.6. Verifying a project

To verify the configuration of all modules and to print out all warnings and errors, use the **verify** command.



6.12.2.6.1. Syntax

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]  
    verify <project>
```

6.12.2.6.2. System properties

This command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.6.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:

Parameter	Description
<code><project></code>	The name of the configuration project on which you want the command to run.

6.12.2.6.4. Example

The following code is an example of how to use the command `verify`:

```
tresos_cmd.bat -Dinfo=false -Dwarning=false verify myproject
```

This command verifies the configuration of the project `myproject` and does not write warning and info messages as output.

6.12.2.7. Generating code for a project

To verify the configuration and to generate code for the modules of a configuration project and to print out all warnings and errors, use the `generate` command.

6.12.2.7.1. Syntax



```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    generate [<option>...] <project>
```

6.12.2.7.2. System properties

This command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.7.3. Options

The following options are supported by this command:

Parameter	Description
-o <output_dir>	By default, the same output directory as in the GUI is used. If you want to generate your code into a different directory on the command-line, use this option and replace <output_dir> by the directory you want to generate the code to.
-g <module-id>	By default, all modules are generated, whose configurations are enabled for code generation in the project. For more information on editing module configurations, see Section 6.6, “Editing module configurations” . You can use the -g option to override that setting and to generate exactly these modules you specify on this command line, regardless whether their configurations are enabled for code generation. You can use the -g option multiple times if you want to generate code for more than one module. The argument <module-id> must be replaced by the ID of the module you want to generate code for, e.g. Com_TS_T16D4M2I0R0. By default, only the module configurations for which code is being generated are verified. To verify all module configurations, even if no code is generated for them, use the -a option.
<p>NOTE  You can only specify module-IDs of modules that are part of your project.</p>	
-v <config-name>	<p>Specifies the name of the module configuration you want to generate. This option can be used in combination with -g to generate only specific modules and configurations.</p> <p>If you use this option, module configurations which are not specified via -v or -g are not generated.</p>



Parameter	Description
	You may use this option multiple times.
-a	If you are using the -g option, only the configurations of the modules specified are verified before the code is generated. If you want all module configurations to be verified, even if no code is generated for them, use this option.

6.12.2.7.4. Parameters

In addition to the common parameters <system_property> and <workspace>, this command also recognizes the following parameters:

Parameter	Description
<project>	The name of the configuration project on which you want the command to run.

6.12.2.7.5. Example

The following code is an example of how to use the command generate:

```
tresos_cmd.bat -Dinfo=false -Dwarning=false generate myproject
```

This command generates code for the modules of the configuration project *myproject* and does not write warning and info messages as output.

6.12.2.8. Listing custom-built generation modes

Some modules may define additional generator modes, which you run with the make command (see [Section 6.12.2.9, “Executing custom-built generation modes for projects”](#)). The listmodes allows to list all modes supported by the available generators. Whether or not a generator is available, depends on whether the module that provides that generator is part of your project.

6.12.2.8.1. Syntax



```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    listmodes <project>
```

TIP

Some of the EB tresos AutoCore modules define additional generator modes. If you purchased these modules, Refer to the EB tresos AutoCore documentation to learn more about supported modes.

6.12.2.8.2. System properties

This command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.8.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:

Parameter	Description
<code><project></code>	The name of the configuration project on which you want the command to run.

6.12.2.8.4. Example

The following code is an example of how to use the command `listmodes`:

```
tresos_cmd.bat listmodes myproject
```

This command lists all the available modes for the configuration project *myproject* and writes them to the commandline.

6.12.2.9. Executing custom-built generation modes for projects

Some modules provide a custom-built generators. You may run these generators by calling a generator mode, other than **generate** and **verify**. The name of the generator mode and what it does depends on the module.



To find out if a module contributes a custom mode, see the documentation of the modules you are running with EB tresos Studio.

6.12.2.9.1. Syntax

To run a module with a custom-built generator in your project, use the following commandline:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
make [<option>...] <mode> <project>
```

6.12.2.9.2. System properties

By default, this command only recognizes the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)). But it is possible for custom-built generators to recognize other system properties. For further information, see the documentation of the generator you are planning to use.

6.12.2.9.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:

Parameter	Description
<code>-o <output_dir></code>	By default, the same output directory as in the GUI is used. If you want to generate your code into a different directory on the commandline, use this option and replace <code><output_dir></code> with the directory you want to generate the code to.
<code>-g <module-id></code>	By default, all modules are generated, whose configurations are enabled for code generation in the project. For more information on editing module configurations, see Section 6.6, “Editing module configurations” . You can use the <code>-g</code> option to override that setting and to generate exactly these modules you specify on this command line, regardless whether their configurations are enabled for code generation. You can use the <code>-g</code> option multiple times if you want to generate code for more than one module. The argument <code><module-id></code> must be replaced by the ID of the module you want to generate code for, e.g. <code>Com_TS_T16D4M2I0R0</code> . Only the module configurations for which code is being generated are verified.



Parameter	Description
<mode>	<p>The name of the custom-built generator. Refer to the documentation of the generator you are planning to use to find out more about the usage.</p> <p>NOTE  To use a custom-built generator on a project, the module which provides that generator must be part of the project.</p> <p>TIP  You can use the <listmodes> command (see Section 6.12.2.8, “Listing custom-built generation modes”) to find out which generator modes are available.</p>
<project>	The name of the configuration project on which you want the command to run.

6.12.2.9.4. Example

The following code is an example of how to use the command `make`:

```
tresos_cmd.bat make generate_SWC-T myproject
```

This command runs a custom-built generator called `generate_SWC-T` on the project *myproject*.

6.12.2.10. Executing unattended wizards on projects

If you have created a configuration for an unattended wizard of a project in the GUI, you can execute this wizard on the commandline. For more information about unattended wizard, see [Section 6.8.10, “Autoconfiguring routine jobs”](#).

6.12.2.10.1. Syntax

To execute an unattended wizard:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    autoconfigure <project> [<wizard>]...
```



6.12.2.10.2. System properties

This command does only recognize the system properties common to all commands (see [Section 6.12.1.3, "System properties common to all commands"](#)).

6.12.2.10.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command does also recognize the following parameters:

Parameter	Description
<code><project></code>	The name of the configuration project on which you want the unattended wizard to run.
<code><wizard></code>	The name of the unattended wizard that shall be run. You can run multiple wizards in a row by providing several names separated by spaces. If you do not specify any wizards, all configured wizards are executed. To retrieve a list of all available wizards for the given project, use the <code>listautoconfigure</code> command (see Section 6.12.2.11, "Listing all unattended wizards for a given project").

6.12.2.10.4. Example

The following code is an example of how to use the command `autoconfigure`:

```
tresos_cmd.bat autoconfigure myproject
```

This command runs all unattended wizards that are configured and checked in the GUI on the project *myproject*. The behaviour of these wizards is controlled by the settings provided in the GUI.

6.12.2.10.5. Running the Calculate Default Values wizard on the command line

To run the Calculate Default Values wizard as configured in the GUI on a project called *myproject*, use:

```
tresos_cmd.bat autoconfigure myproject AutoCalc
```

6.12.2.10.6. Running the Calculate Handle-Ids wizard on the command line

To run the Calculate Handle-Ids wizard as configured in the GUI on a project called *myproject*, use:



```
tresos_cmd.bat autoconfigure myproject AutoHandleId
```

6.12.2.11. Listing all unattended wizards for a given project

You can list all available unattended wizards for a given project on the commandline. For more information about unattended wizards, see [Section 6.8.10, “Autoconfiguring routine jobs”](#).

6.12.2.11.1. Syntax

To list all available unattended wizards for a special project:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    listautoconfigure <project>
```

6.12.2.11.2. System properties

This command does only recognize the system properties common to all commands (see [Section 6.12.1.3, “System properties common to all commands”](#)).

6.12.2.11.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command does also recognize the following parameter:

Parameter	Description
<code><project></code>	The name of the configuration project for which you want retrieve the list of available unattended wizards.

6.12.2.11.4. Example

The following code is an example of how to use the command `listautoconfigure`:

```
tresos_cmd.bat listautoconfigure myproject
```

This command lists all unattended wizards that are available for the project *myproject*.



6.12.2.12. Upgrading module configurations for a project

You can upgrade module configurations for an existing EB tresos Studio project on the command line using the `upgradeModuleConfigs` command. For more information on how to upgrade module configurations using the GUI, see [Section 6.7.2, “Upgrading module configurations to new module versions”](#). The `upgradeModuleConfigs` command upgrades all or specified module configurations of an existing EB tresos Studio configuration project, based on the given parameters.

There is a group of parameters for selecting the module configurations to be upgraded and one parameter for specifying target module IDs. All parameters are optional and if none are given, all module configurations of the project will be upgraded to their latest versions.

The command creates console output for each considered module configuration and whether it was upgraded to which target module ID or not upgraded at all.

WARNING Unspecified target module id may lead to unexpected results



The algorithm to automatically determine matching target modules is based on the module type, which is the AUTOSAR short-name of the module definition. If there is more than one module of the same type, target and derivate installed, then EB tresos Studio chooses the module with the highest software version. This may still not lead to the intended upgrade result. The matched module could still not be an actual successor to the original one. Refer to [Section 6.2.1.5, “Finding all installed modules”](#) on how to retrieve information about the installed modules.

Therefore it is recommended to specify the module configurations to be upgraded together with the respective target module IDs in each call.

6.12.2.12.1. Syntax

To upgrade module configurations for a project:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    upgradeModuleConfigs < projectName > [-onlyEnabled]
    [-type <moduleType>[;<moduleType>]*]
    [-id <moduleId>[;<moduleId>]*]
    [-cfgName <configName>[;<configName>]*]
    [-targetId <targetModuleId>[;<targetModuleId>]*]
```

6.12.2.12.2. System properties

This command only recognizes the system properties common to all commands, see [Section 6.12.1.3, “System properties common to all commands”](#).



6.12.2.12.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:

Parameter	Description
<code>-onlyEnabled</code>	By default, disabled module configurations will also be upgraded. This can be suppressed with the <code>-onlyEnabled</code> command line switch.
<code>-type <moduleType1;moduleType2;...></code>	Defines a module type or a semicolon-separated list of module types. Only these module configurations will be upgraded if they exist.
<code>-id <moduleId1;moduleId2;...></code>	Defines a module ID or a semicolon-separated list of module IDs. Only these module configurations will be upgraded if they exist.
<code>-cfgName <configName1;configName2;...></code>	Defines a module configuration name or a semicolon-separated list of module configuration names. Only these module configurations will be upgraded if they exist.
<code>-targetId <targetModuleId1;targetModuleId2;...></code>	Defines a target module ID or a semicolon-separated list of target module IDs. Only these are taken into consideration for the upgrade. If more than one is specified, they are tried in the given order, and the first one which can be used as a transformation target for a module configuration is taken. If not defined, the module configurations are upgraded to their latest versions.

6.12.2.12.4. Example

The following code is an example of how to use the command `upgradeModuleConfigs`:

```
tresos_cmd.bat upgradeModuleConfigs myproject -onlyEnabled -type Can;EcuC
```

This command upgrades all enabled module configurations of the types *Can* and *EcuC* for the project named *myproject* to their latest versions.

6.12.3. Commands for legacy code generation

In addition to the *project-based* mode, EB tresos Studio also support the *legacy* mode. This mode works independently from projects.



The following section describes all legacy mode commands that handle the generation of code. There are more legacy mode commands which will be explained in later sections.

TIP**Specify a workspace location in legacy mode only if you need it to store e.g. the error log file**

Specifying a workspace location in legacy mode does not have much effect, because there are no projects in legacy mode. However, the workspace is still used internally, e.g. as location for the error log file.

6.12.3.1. Listing all modules available in legacy mode

6.12.3.1.1. Syntax

To print out the names of all installed modules that support legacy code generation, use the `legacy list` command:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>] legacy list
```

6.12.3.1.2. System properties

This command only recognizes the system properties common to all commands. For a list of those commands, see [Section 6.12.1.3, “System properties common to all commands”](#).

6.12.3.1.3. Example

The following is an example of how to use the command `legacy list`:

```
tresos_cmd.bat legacy list
```

6.12.3.2. Generating code

6.12.3.2.1. Syntax

Use the `legacy generate` command to verify module configurations and if no errors occur, generate code for them:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
```



```
legacy generate [<options>]
<config-file>[@<type>] ...
```

6.12.3.2.2. System properties

In addition to the common system properties listed in [Section 6.12.1.3, “System properties common to all commands”](#) and the system properties for converting and merging files, see [Section 6.12.5.1.2, “System properties”](#), the legacy generate command also recognizes the following system properties:

System property	Description
-DmergeConfigs=(true false)	Merges the parameters of the module configurations. This is useful if you have split the parameters of a module configuration into several files. For example, one file contains standard parameters and one contains vendor-specific parameters. The default is <code>false</code> for this command. To merge the parameters of the module configurations, set to <code>true</code> . To not merge the parameters, set this property to <code>false</code> . Instead of merging parameters, the content of the file read last overwrites the previous content.
-DssuppressReadOnlyWarnings=(true false)	Suppresses warnings about read-only configuration parameters. The default is <code>false</code> for this command. To suppress warnings, and not log them or print them out, set to <code>true</code> . To enable warnings and log them or print them out, set to <code>false</code> .
-DpredefinedVariant=<Name>	This property enables the variant resolving and specifies the predefined Variant name that will be taken from the input files. If the parameter is not set, by default the variants resolving is disabled. For detailed information about variant handling, see Section 6.13, “Working with variants and Post-build loadable” .
-Dttarget=<target>	Defines the target architecture, e.g. <code>-Dttarget=WINDOWS</code> . This is necessary when working with module configurations, which use ECU Resources. Further information on ECU Resources is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API.



System property	Description
-Dderivate=<derivate>	<p>Defines the target's derivate, e.g. -Dderivate=Win32x86. This is necessary when working with module configurations, which use ECU Resources. Further information on ECU Resources is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API. If the ECU Resources require additional parameters, such as subderivate, you need to specify any additional parameter with -D<parameter-name>=<parameter-value>. Information on how to specify these parameters is available in the EB tresos Studio developer's guide, chapter ECU Resource Manager API/Characteristics when using the legacy commandline/Defining parameter values for the DefaultEcuResourceFinder.</p>
-DEcuResourceModuleIds=<Module1;Module2;...>	<p>Defines a comma- or semicolon-separated list of module IDs that are necessary to find relevant ECU Resources, e.g. -DEcuResourceModuleIds=Com_TS_T99D01M01I05R01;Rte_TS_T99D01M01I04R02. Information on how to define configured modules for the ECU Resource Finder is available in the EB tresos Studio developer's guide, chapter ECU Resource Manager API/Characteristics when using the legacy commandline/Defining configured modules for the DefaultEcuResourceFinder.</p>
-DpathMapping=auto	<p>Changes automatically</p> <ul style="list-style-type: none"> ▶ the mapping of DEFINITION-REF paths from an unknown path to the path of an installed module ▶ the mapping of reference paths <p>when the input file is read.</p> <p>This option has the same effect as the check box Use automatically calculated path mappings within the AUTOSAR importer mask. For details about the mapping functionality, see Section 6.10.2.2.1, “Path mapping in import mode”.</p>



System property	Description
	<p>NOTE</p>  <p>Restriction</p> <p>This system property only applies when the input file is read.</p>
<pre>-DpathMapping= <sourcePath>=<targetPath> [;...] [;auto]</pre>	<p>Defines paths which shall be changed when a module configuration file is read into the data model. This option has the same effect as the Manual path mappings table within the AUTOSAR importer mask. For details, see Section 6.10.2.2.1, “Path mapping in import mode”.</p> <p>Only changes</p> <ul style="list-style-type: none"> ▶ the mapping of DEFINITION-REF paths ▶ the mapping of reference paths <p>from the given sourcePath to the targetPath when the input file is read.</p> <p>You can combine this option with the automatic path mapping option if you add <code>;auto</code>.</p> <p>The given sourcePath and targetPath must be AUTOSAR SHORT-NAME paths that point to the SHORT-NAME path of an ECUC-MODULE-DEF (AUTOSAR schema) or an ECUC-MODULE-CONFIGURATION-VALUES (AUTOSAR module configuration)!</p> <p>NOTE</p>  <p>Restriction</p> <p>This system property only applies when the input file is read.</p>



System property	Description
	<p>NOTE</p>  <p>Use of -DpreConfig<ModuleID></p> <p>You can use <code>-DpreConfig<ModuleID></code> system properties only for module-IDs which you specify with the <code>-g</code> option. For details about the <code>-g</code> option, see Section 6.12.3.2.3, “Options”.</p>
<code>-DrecConfig<ModuleID>=<ConfigID></code>	<p>Applies the Recommended Configuration with the given <code><ConfigID></code> to the module with the given <code><ModuleID></code>, e.g. <code>-DrecConfigDcm_TS_TxDxM4I0R0=DcmRecConfigurationDflt</code>. Recommended Configuration data can be overwritten by configuration files you specified on the commandline, and therefore implies <code>_DmergeConfigs=true</code>.</p> <p>NOTE</p>  <p>Use of -DrecConfig<ModuleID></p> <p>You can use <code>-DrecConfig<ModuleID></code> system properties only for module-IDs which you specify with the <code>-g</code> option. For details about the <code>-g</code> option, see Section 6.12.3.2.3, “Options”.</p>
<code>-DVerify=false</code>	<p>Disables the verification of the configuration prior the code generation.</p> <p>NOTE</p>  <p>If this option is set to false, code generation is also triggered in case there are verification errors in the configuration. This might lead to wrong generated code!</p>

6.12.3.2.3. Options

The following options are supported by the `legacy generate` command:

Parameter	Description
<code>-a</code>	<p>Sets that all configurations are verified of all modules before generating code, even if code is not generated for all modules.</p> <p>The default is that only the module configurations for which code is being generated are verified.</p> <p>To verify all module configurations, even if no code is generated for them, use this option.</p>



Parameter	Description
	To verify only the configurations of the modules specified before generating code, use the <code>-g</code> option. This option is explained below.
<code>-c <config-path></code>	<p>Sets the directory in which the module configurations are searched as <code><config-path></code>.</p> <p>By default, the directory <code><base.path>\bin</code> is used, with <code>base.path</code> being the value of the system property <code>-Dbase.path</code>.</p> <p>For details about the system property <code>-Dbase.path</code>, see Section 6.12.1.3, “System properties common to all commands”. If the system property is not set, an error is displayed.</p>
<code>-d<var>=<value></code>	<p>Defines a variable called <code><var></code> with value <code><value></code>, which is then available to the code generators.</p> <p>If a code generator for a module supports this feature, refer to this generator's documentation for a list of the available variables and allowed values.</p>
<code>-g <module-id></code>	<p>By default, the code for all specified module configurations is generated. To override that default setting and generate code only for a subset of these modules, use the <code>-g</code> option.</p> <p>You may use this option multiple times if you want to generate code for more than one module.</p> <p>Replace the argument <code><module-id></code> with the ID of the module you want to generate code for, e.g. <code>Com_TS_T16D4M2IOR0</code>.</p> <p>By default, only the module configurations for which code is being generated are verified.</p> <p>To verify all module configurations, even if no code is generated for them, use the <code>-a</code> option.</p>
	<p>NOTE</p>  <p>You can only specify module-IDs of modules that are installed in the copy of EB tresos Studio you are running.</p>
<code>-n <config-name></code>	<p>Specifies the name of the module configurations of the configuration files to load.</p> <p>If you use this option, other module configurations contained in the specified files, are ignored. You may use this option multiple times.</p>



Parameter	Description
<code>-v <config-path></code>	<p>Specifies the SHORT-NAME path of the module configuration you want to generate. This option can be used in combination with <code>-g</code> to generate only specific modules and configurations.</p> <p>If you use this option, module configurations which are not specified via <code>-v</code> or <code>-g</code> are not generated.</p> <p>You may use this option multiple times.</p>
<code>-o <output-path></code>	<p>Sets the directory to which the generated files are written. The files are written to the directory specified as <code><output-path></code>.</p> <p>By default, the directory <code><base.path>\output</code> is used, with <code>base.path</code> being the value of the system property <code>-Dbase.path</code>.</p> <p>For details about the <code>-Dbase.path</code> system property, see Section 6.12.1.3, “System properties common to all commands”. If the system property is not set, an error is displayed.</p>
<code>-u</code>	Specifies that the generated files end with Unix line endings.

6.12.3.2.4. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, the legacy `generate` command also recognizes the following parameters:

Parameter	Description
<code><config-file>[@<type>]</code>	<p>Use <code><config-file></code> to specify the name of the file that contains the module configuration(s) you want to load.</p> <p>The parameter may contain wildcards to match multiple files with just one argument, e.g. <code><path>/*.epc</code>. note that no wildcard characters are supported in <code><path></code> itself.</p> <p>The parameter can be passed multiple times to load more than one file. You may use the optional argument <code>@<type></code> to explicitly specify the content type of the file.</p> <p>For information which content types are available, see Section 3.5.3, “File content types”.</p>



6.12.3.2.5. Example

The following code is an example of how to use the command `legacy generate`:

```
tresos_cmd.bat -DValidate=true legacy generate -n Com -g
    Com_TS_T16D4M2I0R0 -g Custom -u All.epc@asc:4.0.3
```

This example loads a module configuration called *Com* from the file *All.epc*. If *All.epc* contains other module configurations as well, they are ignored.

The *.epc* file is treated as AUTOSAR 4.0.3 file.

The module definition for the configuration is either contained in the module *Com_T16D4M2I0R0*.

The configuration is validated against its definition. The code is generated for the two modules *Com_T16D4M2I0R0* and *Custom* (provided that the module *Custom* does not need a configuration). The generated files contain Unix line feeds.

6.12.3.3. Verifying configurations

Use the `legacy verify` command to verify module configurations and to print out all information, warnings, and errors. This command basically executes the same generation steps as the `legacy generate` command. However there are two differences:

- ▶ No files are written to disk.
- ▶ The code generators also run if the configuration contains errors.

6.12.3.3.1. Syntax

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    legacy verify [<options>] <config-file>[@<type>]...
```

6.12.3.3.2. System properties

In addition to the common system properties listed in [Section 6.12.1.3, “System properties common to all commands”](#) and the system properties for converting and merging files, see [Section 6.12.5.1.2, “System properties”](#), the `legacy verify` command also recognizes the following system properties:

System property	Description
<code>-DMergeConfigs=(true false)</code>	Merges the parameters of the module configurations. This is useful if you have split the parameters of a module configuration into sever-



System property	Description
	<p>al files. For example, one file contains standard parameters and one contains vendor-specific parameters.</p> <p>The default is <code>false</code> for this command.</p> <p>To merge the parameters of the module configurations, set to <code>true</code>.</p> <p>To not merge the parameters but instead to overwrite the previous content with the content of the file read last, set to <code>false</code>.</p>
<code>-DsuppressReadOnlyWarnings=(true false)</code>	<p>Suppresses warnings about read-only configuration parameters. Warnings are not logged or printed out. The default is <code>false</code> for this command.</p> <p>To suppress warnings, set to <code>true</code>.</p> <p>To keep warnings logged or print them out, set to <code>false</code>.</p>
<code>-DpredefinedVariant=<Name></code>	<p>This property enables the variant resolving and specifies the predefined Variant name that will be taken from the input files.</p> <p>If the parameter is not set, by default the variants resolving is disabled.</p> <p>For detailed information about variant handling, see Section 6.13, "Working with variants and Post-build loadable".</p>
<code>-Dttarget=<target></code>	<p>Defines the target architecture, e.g. <code>-Dttarget=WINDOWS</code>. This is necessary when working with module configurations, which use ECU Resources. Information on ECU Resources is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API.</p>
<code>-Dderivate=<derivate></code>	<p>Defines the target's derivate, e.g. <code>-Dderivate=Win32x86</code>. This is necessary when working with module configurations, which use ECU Resources. Information on the ECU Resources Manager API is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API. If the ECU Resources require additional parameters, such as <code>subderivate</code>, you need to specify any additional parameter with <code>-D<parameter-name>=<parameter-value></code>. For information on how to specify these parameters, see the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API/Characteristics when using the legacy commandline/Defining parameter values for the DefaultEcuResourceFinder.</p>



System property	Description
<code>-DEcuResourceModuleIds=<Module1;Module2;...></code>	<p>Defines a comma- or semicolon-separated list of module-IDs, which are necessary for finding relevant ECU Resources, e.g. <code>-DEcuResourceModuleIds=Com_TS_T99D01M01I05R01;Rte_TS_T99D01M01I04R02</code>. For information on how to define the configured modules for the ECU Resource Finder, see the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API/Characteristics when using the legacy commandline/Defining configured modules for the DefaultEcuResourceFinder.</p>
<code>-DVerify=false</code>	<p>Disables the verification of the configuration prior the code generation.</p> <p>NOTE  If this option is set to false, code generation is also triggered in case there are verification errors in the configuration. This might lead to wrong generated code!</p>

6.12.3.3. Options

The following options are supported by the `legacy verify` command:

Parameter	Description
<code>-a</code>	<p>Sets that all configurations of all modules are verified before code is generated, even if code is not generated for all modules.</p> <p>The default is that only the module configurations, for which code is being generated, are verified.</p> <p>To verify all module configurations, even if no code is generated for them, use this option.</p> <p>To verify only the configurations of the modules specified before generating code, use the <code>-g</code> option. This option is explained below.</p>
<code>-c <config-path></code>	<p>Searches for the module configuration files in the directory you specified as <code><config-path></code>.</p> <p>By default, the files are searched in the directory <code><base.path>\bin</code>, with <code>base.path</code> being the value of the system property <code>-Dbase.path</code>.</p>



Parameter	Description
	<p>For details about the system property <code>-Dbase.path</code>, see Section 6.12.1.3, “System properties common to all commands”. If the system property is not set, an error is displayed.</p>
<code>-d<var>=<value></code>	<p>Defines a variable named <code><var></code> with value <code><value></code>, which is then available to the code generators.</p> <p>If a code generator for a module supports this feature, refer to this generator's documentation for a list of the available variables and allowed values.</p>
<code>-g <module-id></code>	<p>By default, the code for all specified module configurations is generated. To override that default setting and generate code only for a subset of these modules, use the <code>-g</code> option.</p> <p>You may use this option multiple times if you want to generate code for more than one module.</p> <p>Replace the argument <code><module-id></code> with the ID of the module you want to generate code for, e.g. <code>Com_TS_T16D4M2IOR0</code>.</p> <p>By default, only the module configurations for which code is being generated are verified.</p> <p>To verify all module configurations, even if no code is generated for them, use the <code>-a</code> option.</p>
	<p>NOTE</p>  <p>You can only specify module-IDs of modules that are installed in the copy of EB tresos Studio you are running.</p>
<code>-n <config-name></code>	<p>Specifies the name of the module configurations of the configuration files to load.</p> <p>If you use this option, other module configurations contained in the specified files, are ignored. You may use this option multiple times.</p>
<code>-v <config-path></code>	<p>Specifies the SHORT-NAME path of the module configuration you want to verify. This option can be used in combination with <code>-g</code> to verify only specific modules and configurations.</p> <p>If you use this option, module configurations which are not specified via <code>-v</code> or <code>-g</code> are not verified.</p> <p>You may use this option multiple times.</p>



6.12.3.3.4. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command does also recognize the following parameters:

Parameter	Description
<code><config-file>[@<type>]</code>	<p>Use <code><config-file></code> to specify the name of the file that contains the module configuration(s) you want to load.</p> <p>The parameter may contain wildcards to match multiple files with just one argument, e.g. <code><path>/*.epc</code>. note that no wildcard characters are supported in <code><path></code> itself.</p> <p>The parameter can be passed multiple times to load more than one file. You may use the optional argument <code>@<type></code> to explicitly specify the content type of the file.</p> <p>For information which content types are available, see Section 3.5.3, "File content types".</p>

6.12.3.3.5. Example

The following code is an example of how to use the command `legacy verify`:

```
tresos_cmd.bat -DValidate=true legacy verify -n Com -g
    Com_TS_T16D4M2I0R0 -g Custom -u All.epc@asc:4.0.3
```

This example loads a module configuration called `Com` from the file `All.epc`. If `All.epc` contains other module configurations as well, they are ignored.

The `.epc` file is treated as AUTOSAR 4.0.3 file.

The module definition for the configuration may be contained in the module `Com_T16D4M2I0R0`. The configuration is validated against its definition.

6.12.3.4. Listing all generation modes

It is possible that a module provides a custom-built generator mode other than `generate` and `verify`.

The name of these additional and module-specific generator modes and their functionality depend upon the individual module.

- ▶ To find out if a module provides any additional modes, refer to the documentation of the modules you are running with EB tresos Studio.



- ▶ To find out which generation modes the modules you use have, use the command **legacy listmodes**.
- ▶ To run such modes, call the generator mode from the commandline. For details, see [Section 6.12.3.5, “Running generation modes”](#).

6.12.3.4.1. Syntax

To list all available legacy generation modes for generators of all modules in legacy mode, use the following syntax:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]  
    legacy listmodes
```

6.12.3.4.2. System properties

The **legacy listmodes** command only recognizes the system properties common to all commands. For a list of these system properties, see [Section 6.12.1.3, “System properties common to all commands”](#).

6.12.3.4.3. Example

The following is an example of how to use the **legacy listmodes** command:

```
tresos_cmd.bat legacy listmodes
```

If you run this command you will see all module-IDs and their available generation modes.

6.12.3.5. Running generation modes

To run generator modes in legacy mode other than **generate** and **verify**, use the **legacy make** command.

The name of these additional and module-specific generator modes and their functionality depend upon the individual module. To list all available generation modes, see [Section 6.12.3.4, “Listing all generation modes”](#).

6.12.3.5.1. Syntax

If a module supports the legacy mode, you can run its custom-built generator on a configuration with the **legacy make** command:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]  
    legacy make <mode> [<options>]
```



<config-file>[@<type>] ...

6.12.3.5.2. System properties

In addition to the common system properties listed in [Section 6.12.1.3, “System properties common to all commands”](#) and the system properties for converting and merging files, see [Section 6.12.5.1.2, “System properties”](#), the **legacy make** command also recognizes the following system properties:

System property	Description
-DmergeConfigs=(true false)	<p>Merges the parameters of the module configurations. This is useful if you have split the parameters of a module configuration into several files (e.g. one file contains standard and another contains vendor-specific parameters).</p> <p>The default is <code>false</code> for this command.</p> <p>To merge the parameters of the module configurations, set to <code>true</code>.</p> <p>To not merge the parameters, set this property to <code>false</code>. Instead of merging parameters, the content of the file read last overwrites the previous content.</p>
-Dttarget=<target>	<p>Defines the target architecture, e.g. <code>-Dttarget=WINDOWS</code>. This is necessary when working with module configurations, which use ECU Resources. Information on the ECU Resource Manager API is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API.</p>
-Dderivate=<derivate>	<p>Defines the target's derivate, e.g. <code>-Dderivate=Win32x86</code>. This is necessary when working with module configurations, which use ECU Resources. Information on the ECU Resource Manager API is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API. If the ECU Resources require additional parameters, such as <code>subderivate</code>, you need to specify any additional parameter with <code>-D<parameter-name>=<parameter-value></code>. Information on how to specify these parameters is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API/Characteristics when using the legacy commandline/Defining parameter values for the DefaultEcuResourceFinder.</p>
-DEcuResourceModuleIds=<Module1;Module2;...>	<p>Defines a comma- or semicolon-separated list of module-IDs, which are necessary for finding relevant ECU Resources, e.g. -</p>



System property	Description
	<p>DECuResourceModuleIds=Com_TS_T99D01M01I05R01;Rte_TS_T99D01M01I04R02. Information on how to define these configured modules for the ECU Resource Finder is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API/Characteristics when using the legacy commandline/Defining configured modules for the DefaultEcuResourceFinder.</p>
-DpredefinedVariant=<Name>	<p>This property enables the variant resolving and specifies the predefined Variant name that will be taken from the input files.</p> <p>If the parameter is not set, by default the variants resolving is disabled.</p> <p>For detailed information about variant handling, see Section 6.13, "Working with variants and Post-build loadable".</p>
-DpreConfig<ModuleID>=<ConfigID>	<p>Applies the Pre-Configuration with the given <ConfigID> to the module with the given <ModuleID>, e.g. -DpreConfigDcm_TS_TxDxM4I0R0=DcmPreConfigurationDflt. Pre-Configuration data can not be overwritten by configuration files you specified on the commandline.</p> <p>NOTE  Use of -DpreConfig<ModuleID> You can use -DpreConfig<ModuleID> system properties only for module-IDs which you specify with the <u>-g</u> option. For details about the -g option, see Section 6.12.3.5.3, "Options".</p>
-DrecConfig<ModuleID>=<ConfigID>	<p>Applies the Recommended Configuration with the given <ConfigID> to the module with the given <ModuleID>, e.g. -DrecConfigDcm_TS_TxDxM4I0R0=DcmRecConfigurationDflt. Recommended Configuration data can be overwritten by configuration files you specified on the commandline, and therefore implies <u>-DmergeConfigs=true</u>.</p> <p>NOTE  Use of -DrecConfig<ModuleID> You can use -DrecConfig<ModuleID> system properties only for module-IDs which you specify with the <u>-g</u> option. For details about the -g option, see Section 6.12.3.5.3, "Options".</p>
-DVerify=false	<p>Disables the verification of the configuration prior the code generation.</p>



System property	Description
	<p>NOTE</p>  <p>If this option is set to false, code generation is also triggered in case there are verification errors in the configuration. This might lead to wrong generated code!</p>

6.12.3.5.3. Options

The following options are supported by the **legacy make** command:

Parameter	Description
-a	<p>Sets that all configurations are verified of all modules before generating code, even if code is not generated for all modules.</p> <p>The default is that only the module configurations for which code is being generated are verified.</p> <p>To verify all module configurations, even if no code is generated for them, use this option.</p> <p>To verify only the configurations of the modules specified before generating code, use the -g option. This option is explained below.</p>
-c <config-path>	<p>Sets the directory in which the module configurations are searched as <config-path>.</p> <p>By default, the directory <base.path>\bin is used, with base.path being the value of the system property -Dbase.path.</p> <p>For details about the system property -Dbase.path, see Section 6.12.1.3, “System properties common to all commands”. If the system property is not set, an error is displayed.</p>
-d<var>=<value>	<p>Defines a variable called <var> with value <value>, which is then available to the code generators.</p> <p>If a code generator for a module supports this feature, refer to this generator's documentation for a list of the available variables and allowed values.</p>
-g <module-id>	<p>By default, the code for all specified module configurations is generated. To override that default setting and generate code only for a subset of these modules, use the -g option.</p>



Parameter	Description
	<p>You may use this option multiple times if you want to generate code for more than one module.</p> <p>Replace the argument <module-id> with the ID of the module you want to generate code for, e.g. Com_TS_T16D4M2IOR0.</p> <p>By default, only the module configurations for which code is being generated are verified.</p> <p>To verify all module configurations, even if no code is generated for them, use the <code>-a</code> option.</p>
	<p>NOTE</p> 
	<p>You can only specify module-IDs of modules that are installed in the copy of EB tresos Studio you are running.</p>
<code>-n <config-name></code>	<p>Specifies the name of the module configurations of the configuration files to load.</p>
	<p>If you use this option, other module configurations contained in the specified files, are ignored. You may use this option multiple times.</p>
<code>-v <config-path></code>	<p>Specifies the SHORT-NAME path of the module configuration you want to generate. This option can be used in combination with <code>-g</code> to generate only specific modules and configurations.</p> <p>If you use this option, module configurations which are not specified via <code>-v</code> or <code>-g</code> are not generated.</p> <p>You may use this option multiple times.</p>
<code>-o <output-path></code>	<p>Sets the directory to which the generated files are written. The files are written to the directory specified as <code><output-path></code>.</p> <p>By default, the directory <code><base.path>\output</code> is used, with <code>base.path</code> being the value of the system property <code>-Dbase.path</code>.</p> <p>For details about the <code>-Dbase.path</code> system property, see Section 6.12.1.3, “System properties common to all commands”. If the system property is not set, an error is displayed.</p>
<code>-u</code>	<p>Specifies that the generated files end with Unix line endings.</p>



6.12.3.5.4. Parameters

In addition to the common parameters <system_property> and <workspace>, the **legacy make** command also recognizes the following parameters:

Parameter	Description
<config-file>[@<type>]	<p>Use <config-file> for the name of the file that contains the module configuration(s) that you want to load.</p> <p>You may pass the parameter multiple times to load more than one file.</p> <p>Use the optional argument @<type> to explicitly specify the content type of the file. For information, which content types are available, see Section 3.5.3, “File content types”.</p>

6.12.3.5.5. Example

The following code is an example of how to use the **legacy make** command:

```
tresos_cmd.bat -DmergeConfigs=true legacy make generate_SWC-T
    myEpc.epc myOther.epc
```

This example loads the content of the two files `myEpc.epc` and `myOther.epc`.

The module-configurations contained in `myEpc.epc` are merged with the ones in `myOther.epc`, but not overwritten.

A custom-built generator called `generate_SWC-T` is invoked. The above example assumes that the modules of the configurations contained in the loaded files support this generation mode.

6.12.4. Commands for module development

The commands in this chapter cover printing out data models or creating and checking vendor-specific module definitions, amongst others. The commands are useful for debugging purposes and module development.

NOTE

This chapter is for module developers only



The commands in this chapter are useful for debugging and module developing only. If you are using the GUI, you may not want to read this chapter.

The following topics are available:



- ▶ [Section 6.12.4.1, “Listing a data model”](#)
- ▶ [Section 6.12.4.2, “Creating vendor-specific module definitions \(VSMD\)”](#)
- ▶ [Section 6.12.4.3, “Checking vendor-specific against standard module definitions”](#)
- ▶ [Section 6.12.4.4, “Listing all rules used for VSMD checking”](#)
- ▶ [Section 6.12.4.5, “Encrypting and signing modules”](#)

6.12.4.1. Listing a data model

Use the `legacy tree` command to print out the data model loaded from modules and module configurations.

6.12.4.1.1. Syntax

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    legacy tree [<options>] <config-file>[@<type>]...
```

6.12.4.1.2. System properties

In addition to the common system properties (see [Section 6.12.1.3, “System properties common to all commands”](#)), this command also recognizes the following system properties:

System property	Description
<code>-DMergeConfigs=(true false)</code>	<p>Merges the parameters of the module configurations. This is useful if you have split the parameters of a module configuration into several files (e.g. one file containing standard- and one containing vendor-specific parameters).</p> <p>The default is <code>false</code> for this command.</p> <p>To merge the parameters of the module configurations, set the property to <code>true</code>.</p> <p>To not merge the parameters, set the property to <code>false</code>. The content of the file read last overwrites the previous content.</p>

6.12.4.1.3. Options

The following options are supported by this command:



Parameter	Description
-c <config-path>	<p>Use this option to set the directory in which module configuration files are searched to <config-path>.</p> <p>The default directory for module configuration files is <base.path>\bin, with base.path being the value of the system property -Dbase.path (for details on -Dbase.path, see Section 6.12.1.3, "System properties common to all commands"). If the system property is not set, an error is displayed.</p>
-n <config-name>	<p>Use this option to specify the name of the module configurations of the configuration files you wish to load. If you use this option, other module configurations, which are contained in the files you specified, are ignored. You can use this option multiple times.</p>

6.12.4.1.4. Parameters

In addition to the common parameters <system_property> and <workspace>, this command also recognizes the following parameters:

Parameter	Description
<config-file>[@<type>]	<p>Use <config-file> to specify the name of the file, which contains the module configuration(s) you wish to load. You may pass this parameter multiple times to load more than one file.</p> <p>Use the optional argument @<type> to explicitly specify the content type of the file. For information, which content types are available, see Section 3.5.3, "File content types".</p>

6.12.4.1.5. Example

The following code is an example of how to use the command legacy tree:

```
tresos_cmd.bat legacy tree -n Com All.epc@asc:4.0.3
```

This example loads a module configuration named *Com* from the file *All.epc* into the data model and prints the model as a tree. If this file contains other module configurations as well, they are ignored. The *.epc* file is treated as AUTOSAR 4.0.3 file.

6.12.4.2. Creating vendor-specific module definitions (VSMD)



According to the AUTOSAR specification Specification of ECU Configuration, a vendor has to derive a vendor-specific module definition (VSMD) from the corresponding AUTOSAR standard module definition (StMD). You can use the `vsmd` command to let EB tresos Studio derive a basic vendor-specific module definition for you.

TIP

Since it is possible to specify different content and file types for StMD and VSMD, you may also use the `vsmd` command to convert files from one format to the other.

For further information on content types, see [Section 3.5.3, “File content types”](#).

To derive a VSMD from a StMD, several rules apply. These rules are part of the AUTOSAR specification Specification of ECU Configuration. The most important one requires you to choose your own package name in the module definition file under which your module definition is located.

If you use the `vsmd` command, it creates a reference to the StMD using the REFINED-MODULE.DEF tag. In addition, new UUIDs are created for each element of the VSMD.

Additionally, a default ADMIN-DATA section is created for each contained MODULE-DEF.

6.12.4.2.1. Syntax

Use the `vsmd` command to create a basic vendor-specific module definition from a standard module definition as shown in the following example code:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    legacy vsmd [<options>] <stmd-file>[@<type>]...
    <vsmd-file>[@<type>] <package> [<origin>]
```

6.12.4.2.2. System properties

In addition to the common system properties (see [Section 6.12.1.3, “System properties common to all commands”](#)), this command also recognizes the following system properties:

System property	Description
<code>-DmergeConfigs= (true false)</code>	Merges the parameters of the module configurations. This is useful if you have split the parameters of a module configuration into several files, e.g. one file that contains standard- and one that contains vendor-specific parameters. The default is <code>true</code> for this command.



System property	Description
	<p>To merge the parameters of the module configurations, set the property to <code>true</code>.</p> <p>To not merge the parameters, set the property to <code>false</code>. The content of the file read last overwrites the content.</p>
<code>-DwriteDefaults= (true false)</code>	<p>Deprecated property.</p> <p>Forces the LOWER-MULTIPLICITY and UPPER-MULTIPLICITY tags to be printed to the output-file, even if their value is 1.</p> <p>The default value for this parameter is <code>false</code> and thus the tags are usually omitted.</p> <p>This property was used in previous versions to achieve conformance to XSD schema files. This property has been replaced by content types. For information about content types, see Section 3.5.3, “File content types”.</p>
<code>-Duuids=(true false)</code>	<p>Creates new random unique IDs for each node. The default is <code>true</code>.</p> <p>To create new random unique IDs for each node, set to <code>true</code>.</p>
<code>-DecuParamDef= (true false)</code>	<p>Generates a ECU-PARAMETER-DEFINITION tag into the output file. The default is <code>true</code>.</p> <p>To generate a ECU-PARAMETER-DEFINITION tag into the output file, set to <code>true</code>.</p> <p>To not generate a ECU-PARAMETER-DEFINITION tag into the output file, set to <code>false</code>.</p>
<code>-DecuParamDefName= <name></code>	<p>Changes the name of the ECU-PARAMETER-DEFINITION tag in the output file. The default value is <code>myEcuParameterDefinition</code>.</p>
<code>-DbswModuleDesc= (true false)</code>	<p>Generates a BSW-MODULE-DESCRIPTION tag into the output file. The default is <code>true</code>.</p> <p>To generate a BSW-MODULE-DESCRIPTION tag into the output file, set to <code>true</code>.</p> <p>To not generate a BSW-MODULE-DESCRIPTION tag into the output file, set to <code>false</code>.</p>
<code>-DbswModuleDescName= <name></code>	<p>Changes the name of the BSW-MODULE-DESCRIPTION tag in the output file. The default value is <code>myBswModuleDescription</code>.</p>
<code>-DvsmdPath= <as-path></code>	<p>The <code><as-path></code> argument explicitly sets the value of the VENDOR-SPECIFIC-MODULE-DEF-REF parameter inside the BSW-MOD-</p>



System property	Description
	<p>ULE-DESCRIPTION container of the output file. By default, this path is determined automatically.</p> <p>Precondition: To use <code>-DvsmdPath=<as-path></code>, set the property <code>-DbswModuleDesc=true</code>.</p>
<code>-DrecommendedConfigPath= <as-path></code>	<p>The <code><as-path></code> argument explicitly sets the value of the RECOMMENDED-CONFIGURATION-REF parameter inside the BSW-MODULE-DESCRIPTION container of the output file. If you do not set this property, no recommended configuration is loaded.</p> <p>Precondition: To use <code>-DrecommendedConfigPath=<as-path></code>, set the property <code>-DbswModuleDesc=true</code>.</p>
<code>-DpreConfigPath= <as-path></code>	<p>The <code><as-path></code> argument explicitly sets the value of the PRECONFIGURED-CONFIGURATION-REF parameter inside the BSW-MODULE-DESCRIPTION container of the output file. If you do not set this property, no preconfiguration is loaded.</p> <p>Precondition: To use <code>-DpreConfigPath=<as-path></code>, set the property <code>-DbswModuleDesc=true</code>.</p>
<code>-DssuppressReadOnlyWarnings= (true false)</code>	<p>Suppresses warnings regarding read-only configuration parameters; does not log them or print them out. The default is <code>false</code> for this command.</p> <p>To suppress warnings, not log them or print them out, set to <code>true</code>.</p> <p>To keep warnings, to log them or print them out, set to <code>false</code>.</p>
<code>-DMapOptionalAsList= (true false)</code>	<p>Creates a surrounding list for elements with LOWER-MULTIPLICITY= 0 and UPPER-MULTIPLICITY=1. For backward-compatibility reasons, the default value is <code>true</code>.</p> <p>Precondition: To use <code>-DMapOptionalAsList=(true false)</code>, the content type of the output file must be XDM.</p> <p>To create a surrounding list for elements with LOWER-MULTIPLICITY= 0 and UPPER-MULTIPLICITY=1, set to <code>true</code>.</p> <p>To not create a surrounding list, set to <code>false</code>. Instead, the affected elements are marked as optional.</p>



System property	Description
	<p>NOTE</p>  <p>Leaving this property at the default, or explicitly setting it to true can lead to problems!</p> <p>If you leave this property at the default value or explicitly set it to <code>true</code>, the XPaths of the affected elements depend on their LOWER-MULTIPLICITY value.</p> <p>If the LOWER-MULTIPLICITY is 0, a list is created. If the LOWER-MULTIPLICITY is later changed to 1, no list is created (with the UPPER-MULTIPLICITY always being at 1). The XPath is different in both cases.</p> <p>Code templates, which then wish to address this parameter in the configuration need to be able to handle both cases.</p> <p>However if you set this system property to <code>false</code>, the XPath stays the same in any case.</p>
	<p>TIP</p>  <p>If you are writing a code-template, you can use the following XPath function in the template to determine if the parameter is configured or not:</p> <pre>node:exists(OptionalVariable).</pre>

6.12.4.2.3. Options

The following options are supported by this command:

Parameter	Description
<code>-c <config-path></code>	<p>Use this option to perform the search for module configuration files in the directory you specify as <code><config-path></code>.</p> <p>By default, the directory <code><base.path>\bin</code> is used for the search, with <code>base.path</code> being the value of the system property <code>-Dbase.-path</code> (see Section 6.12.1.3, “System properties common to all commands”). If the system property is not set, an error is displayed.</p>
<code>-o <output-path></code>	<p>Use this option to write the generated files to the directory specified as <code><output-path></code>.</p>



Parameter	Description
	<p>By default, the directory <base.path>\output is used, with base.path being the value of the system property -Dbase.path. For details about this system property, see Section 6.12.1.3, "System properties common to all commands".</p> <p>If the system property is not set, an error is displayed.</p>
-u	Use this option, to generate all files with Unix line endings.
-n <config-name>	<p>Specifies the name of the module configurations of the configuration files to load.</p> <p>If you use this option, other module configurations contained in the specified files, are ignored. You may use this option multiple times.</p>

6.12.4.2.4. Parameters

In addition to the common parameters <system_property> and <workspace>, this command also recognizes the following parameters:

Parameter	Description
<stmd-file>[@<type>]	<p>Defines the file name that contains the standardized module definition (StMD) from which to create the vendor-specific module definition (VSMD). If the file contains more than one module definition, all files are loaded.</p> <p>Use the optional argument @<type> to explicitly specify the content type of the file. To find out which content types are available, see Section 3.5.3, "File content types".</p>
<vsmd-file>[@<type>]	<p>Defines the name of the file to which the created VSMD is written. Use the optional argument @<type> to explicitly specify the content type of the file. To find out which content types are available, see Section 3.5.3, "File content types".</p> <p>If you do not specify any content type, the file extension is interpreted. Possible extensions are: *.xdm, *.arxml, *.asc, *.epd, *.-bmd, *.epc.</p>
<package>	Defines the name of the AUTOSAR package under which your module definition is located.
<origin>	Defines the origin of the parameters. This parameter is only used if the content type of the output file is XDM. If that is so, each parameter



Parameter	Description
	in the XDM file gets an attribute named __ORIGIN with the text that you specified here as value.

6.12.4.2.5. Example

The following code is an example of how to use the command legacy vsmd:

```
tresos_cmd.bat -DMapOptionalAsList=false legacy vsmd
    Com.epd MyCom.xdm TS_T16D4M2I0R0
```

This example creates a VSMD file with the name MyCom.xdm from the StMD Com.epd. The created VSMD contains the AR-PACKAGE TS_T16D4M2I0R0. Optional elements are not surrounded by lists in the output file, which is the recommended way of doing this.

6.12.4.3. Checking vendor-specific against standard module definitions

There are some rules, you must follow when you derive a vendor-specific module definition (VSMD) from a standardized module definition (StMD). Refer to the AUTOSAR specification Specification of ECU Configuration for more information about these rules.

6.12.4.3.1. Syntax

To check if a VSMD conforms to a StMD, use the legacy vsmdcheck command as shown in the following code example:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    legacy vsmdcheck
    <stmd-file>[@<type>] <vsmd-file>[<type>] ...
    <rule|ruleset>[,<rule|ruleset>...]
    [-ignore <rule>]... [-xml <xmlfile>]
```

6.12.4.3.2. System properties

In addition to the common system properties listed in [Section 6.12.1.3, “System properties common to all commands”](#), this command also recognizes the following system properties:



System property	Description
<code>-DssuppressReadOnlyWarnings=</code> <code>(true false)</code>	<p>Suppresses warnings about read-only parameters; the warnings are not logged or printed out. The default is <code>false</code> for this command.</p> <p>To suppress warnings, set to <code>true</code>.</p> <p>To log or print out read-only parameters, set to <code>false</code>.</p>
<code>-Dttarget= <target></code>	<p>Defines the target architecture, e.g. <code>-Dttarget=WINDOWS</code>. This is necessary when working with module configurations, which use ECU Resources. Information on the ECU Resources Manager API is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API.</p>
<code>-Dderivate= <derivate></code>	<p>Defines the target's derivate, e.g. <code>-Dderivate=Win32x86</code>. This is necessary when working with module configurations which use ECU Resources. Information on ECU Resources is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API. If the ECU Resources require additional parameters such as <code>subderivate</code> then you need to specify any additional parameter with <code>-D<parameter-name>=<parameter-value></code>. For information on how to specify these parameters, refer to the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API/Characteristics when using the legacy commandline/Defining parameter values for the DefaultEcuResourceFinder.</p>
<code>-DEcuResourceModuleIds=</code> <code><Module1;Module2;...></code>	<p>Defines a comma- or semicolon-separated list of module-IDs that are necessary for finding relevant ECU Resources, e.g. <code>-DEcuResourceModuleIds=Com_TS_T99D01M01I05R01;Rte_TS_T99D01M01I04R02</code>. For information on how to define configured modules for the ECU Resource Finder, see the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API/Characteristics when using the legacy commandline/Defining configured modules for the DefaultEcuResourceFinder.</p>

6.12.4.3.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:



Parameter	Description
<stmd-file>[@<type>]	Defines the file name that contains the standardized module definition (StMD). If the file contains more than one module definition, all definitions are loaded. It is recommended to always load all StMDs to e.g. be able to check references pointing to other modules. A file containing all StMDs is provided with the EB tresos Studio installation inside the <code>autosar</code> folder. Use the optional argument <code>@<type></code> to explicitly specify the content type of the file. For information, which content types are available, see Section 3.5.3, “File content types” .
<vsmd-file>[@<type>]	Defines the file name that contains the vendor-specific module definition (VSMD). If the file contains more than one module definitions, all definitions are loaded. Use the optional argument <code>@<type></code> to explicitly specify the content type of the file. For information, which content types are available, see Section 3.5.3, “File content types” .
<rule ruleset>	Defines the rule or rule-set to be used. For a list of available rules, see the <code>legacy listrules</code> command in Section 6.12.4.4, “Listing all rules used for VSMD checking” . Available rule-sets are: <code>asc:2.-0</code> , <code>asc:2.1</code> , <code>asc:3.0</code> , <code>asc:3.1</code> , <code>asc:3.2</code> , <code>asc:4.0.2</code> , <code>asc:4.-0.3</code> , <code>asc:4.0.3.RFCs</code> , <code>asc:4.1.3</code> , <code>asc:4.2.1</code> , <code>asc:4.2.2</code> , <code>asc:4.3.0</code> , <code>asc:4.3.1</code> , and <code>asc:4.4.0</code>
<code>-ignore <rule></code>	Defines the name of a rule which shall not be checked. For a list of available rules, see the <code>legacy listrules</code> command in Section 6.12.4.4, “Listing all rules used for VSMD checking” .
<code>-xml <xmlfile></code>	Defines the name of a file into which the result shall be saved in XML format.

6.12.4.3.4. Example

The following code is an example of how to use the command `legacy vsmdcheck`:

```
tresos_cmd.bat -DValidate=true legacy vsmdcheck
    AUTOSAR_MOD_ECUConfigurationParameters.arxml@asc:4.0.3
    VSMD.epd@asc:4.0.3 asc:4.0.3 -ignore EcucSws_1035
```

This example validates the VSMD file with the name `VSMD.epd` against the StMD file with the name `StMD.bmd`. Both files are regarded as AUTOSAR 4.0.3 files. The rule-set used for the checks is `asc:4.0.3`, but the rule `EcucSws_1035` is not checked.



NOTE**Use all StMDs to check a VSMD**

It is recommended to always load all StMDs to e.g. be able to check references pointing to other modules. A file containing all StMDs is provided with the EB tresos Studio installation inside the `autosar` folder.

6.12.4.4. Listing all rules used for VSMD checking

6.12.4.4.1. Syntax

To list all rules that are used for the legacy `vsmdcheck` command, use the legacy `listrules` command:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>] legacy listrules
```

6.12.4.4.2. System properties

This command only recognizes the system properties common to all commands. For a list of these system properties, see [Section 6.12.1.3, “System properties common to all commands”](#).

6.12.4.4.3. Example

The following example lists all rules used for VSMD checking with the legacy `listrules` command:

```
tresos_cmd.bat legacy listrules
```

6.12.4.5. Encrypting and signing modules

If you are developing your own module for use in EB tresos Studio as an EB partner, you may use the `crypto` command to encrypt and sign modules. For more information about this command, see the EB tresos Studio developer's guide, chapter [Developing modules/How to sign modules with the crypto command](#).

6.12.5. Commands for converting and merging files

This section shows you how to convert files to different formats and how to merge the content of files.



6.12.5.1. Merging files and converting between different file formats

To merge and convert files, use the `convert` command. This command reads one or more files with any content type known to EB tresos Studio. The command merges the content of the read files and writes all read data to an output file with one of the following file formats:

- ▶ XDM
- ▶ AUTOSAR module configuration
- ▶ AUTOSAR system description

There are two different types of data that can be written to the output file: module configuration data and system description data. The input files must contain the same type of data that is supposed to be written to the output file. If it does not contain the same type of data, EB tresos Studio does not necessarily report an error, when you try to convert between the different types of data. However, the output file will probably be almost empty. For information, which file formats contain which kind of data, see [Section 3.5.3, “File content types”](#).

6.12.5.1.1. Syntax

To use the command `convert`, use the following syntax:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    legacy convert [<options>]
        <input-file>[@<type>]... <output-file>[@<type>]
```

6.12.5.1.2. System properties

The command `convert` recognizes the common system properties listed in [Section 6.12.1.3, “System properties common to all commands”](#). In addition to the common system properties, the `convert` command also recognizes additional system properties, depending on the kind of data to read and write.

Additional system properties when converting module configuration data:

System property	Description
<code>-DmergeConfigs=(true false)</code>	Merges parameters of module configurations. Merging parameters is useful if you have split the parameters of a module configuration into several files, e.g. if one file contains standard and another contains vendor-specific parameters. The default is <code>true</code> for this command. To merge the parameters of the module configurations, set the property to <code>true</code> . To not merge parameters, set this property to <code>false</code> . The content of the file read last overwrites the previous content.



System property	Description
-DwriteDefaults=(true false)	<p>Deprecated property. Use it to force the LOWER-MULTIPLICITY and UPPER-MULTIPLICITY tags to be printed to the output file, even if their value is 1. The default value for this parameter is false and thus the tags are usually omitted. This property was used in previous versions of EB tresos Studio to achieve conformance to XSD schema files. These schema files are now replaced by content types. For details about content files, see Section 3.5.3, “File content types”.</p>
-Duuids=(true false)	<p>To create new random unique IDs for each node, set this property to true. The default value is true.</p>
-DssuppressReadOnlyWarnings=(true false)	<p>To not log or print out warnings regarding read-only configuration parameters, set to true. The default value for this command is false, which logs and prints out warnings about read-only configuration parameters.</p>
-DMapOptionalAsList=(true false)	<p>This parameter can only be used if the content type of the output file is XDM.</p> <ul style="list-style-type: none"> ▶ Set to true to create a surrounding list for elements with LOWER-MULTIPLICITY= 0 and UPPER-MULTIPLICITY=1. ▶ Set to false to not create surrounding lists. Instead, the affected elements are marked as optional. <p>For backward-compatibility reasons, the default value is true.</p>



System property	Description
	<p>NOTE</p>  <p>Leaving this property at the default, or explicitly setting it to <code>true</code> can lead to problems!</p> <p>If you leave this property at the default or explicitly set it to <code>true</code>, the XPaths of the affected elements depend on their LOWER-MULTIPLICITY value. If the LOWER-MULTIPLICITY is 0, a list is created. If the LOWER-MULTIPLICITY is later changed to 1, no list is created (with the UPPER-MULTIPLICITY always being at 1). In both cases, the XPath is different. Code templates, which then try to address this parameter in the configuration, need to be able to handle both cases. However if you set this system property to <code>false</code>, the XPath always stays the same.</p>
	<p>TIP</p>  <p>If you are writing a code template, you may use the following XPath function in the template to determine if the parameter is configured or not: <code>node:exists(OptionalVariable)</code>.</p>
<code>-DmergeAttributes=(true false)</code>	<p>This property can only be used if you have set – <code>DmergeConfigs=true</code>. The property controls how to deal with the attributes of nodes that are present in more than one input file. The default value is <code>true</code>.</p> <ul style="list-style-type: none"> ▶ If this property is set to <code>true</code>, attributes with multiple values are merged so that the resulting attribute contains all values. How single-value attributes are dealt with, is controlled by the value of the property <code>-DoverwriteAttributes</code>. ▶ If this property is set to <code>false</code>, the result only contains those attributes and their values that were present when the node was read the first time.
<code>-DoverwriteAttributes=(true false)</code>	<p>This property can only be used if you have set – <code>DmergeAttributes=true</code>. The property controls how to deal with the attributes of nodes that are present in more than one input file. The default value is <code>true</code>.</p> <p>If more than one input file is used, this property controls how to handle the attributes of node present in several files.</p>



System property	Description
<p>-DwriteXPathAttributes=(true false)</p>	<p>▶ Set the property to <code>true</code>, to overwrite the values of the attributes with their last occurrence.</p> <p>▶ Set the property to <code>false</code>, to use the value of the first occurrence of that attribute.</p> <p>When you write an AUTOSAR schema file, not all XPath expressions used for attributes can be converted to a fixed value as needed by the AUTOSAR format. The commandline option <code>writeXPathAttributes</code> controls how to handle such non-computable attributes.</p> <p>The default value for <code>writeXPathAttributes</code> is <code>true</code>.</p> <p>If you set <code>writeXPathAttributes</code> to the value <code>true</code>, and:</p> <ul style="list-style-type: none"> ▶ If a value can be determined, the converter tries to evaluate the expressions of computable attributes and writes the resulting value. ▶ If an attribute cannot be evaluated, e.g. if it accesses a non-existing configuration value, the commandline operation fails with an error. <p>If you set <code>writeXPathAttributes</code> to the value <code>false</code>, computable attributes are never evaluated. Instead</p> <ul style="list-style-type: none"> ▶ if the corresponding AUTOSAR XML tag is optional, it is omitted ▶ or a reasonable default is written. <p>A warning is issued on the commandline for each computable attribute.</p> <p>When any of the above conversions has taken place: If the conversion succeeds with no error and the input to the conversion is an AUTOSAR-compatible model, an AUTOSAR-compatible output is produced.</p> <p>Depending on the value of <code>writeXPathAttributes</code>, the results of the conversion vary:</p> <ol style="list-style-type: none"> 1. If <code>writeXPathAttributes</code> is set to <code>true</code> and if a XDM parameter definition is converted to AUTOSAR format, AUTOSAR tags that result from a computable attribute, which could not be evaluated, are handled the following way: <ul style="list-style-type: none"> ▶ LOWER-MULTIPLICITY : Conversion fails with an error.



System property	Description
	<ul style="list-style-type: none"> ▶ UPPER-MULTIPLICITY : Conversion fails with an error. ▶ ENUM-LITERAL-DEF : Conversion fails with an error. ▶ DEFAULT-VALUE : Conversion fails with an error. ▶ MIN : <ul style="list-style-type: none"> ▶ If no <code>INVALID</code> attribute is used in the input model, no <code>MIN</code> tag is written. ▶ If an <code>INVALID</code> attribute is used that is not of type <code>Range</code>, a warning is issued on the commandline and a reasonable default is written to the resulting AUTOSAR file. ▶ If an <code>INVALID</code> attribute of type <code>Range</code> is used but no minimum value is set, no <code>MIN</code> tag is written. ▶ MAX : <ul style="list-style-type: none"> ▶ If no <code>INVALID</code> attribute is used in the input model, no <code>MAX</code> tag is written. ▶ If an <code>INVALID</code> attribute is used that is not of type <code>Range</code>, a warning is issued on the commandline and a reasonable default is written to the resulting AUTOSAR file. ▶ If an <code>INVALID</code> attribute of type <code>Range</code> is used but no maximum value is set, no <code>MAX</code> tag is written. <p>2. If <code>writeXPathAttributes</code> is set to <code>false</code> and if an XDM parameter definition is converted to AUTOSAR format, AUTOSAR tags that result from a computable attribute are handled the following way, no matter if the tag can be evaluated or not:</p> <ul style="list-style-type: none"> ▶ LOWER-MULTIPLICITY: A warning is printed in the commandline. A <code>LOWER-MULTIPLICITY</code> of <code>0</code> is written to the AUTOSAR file. ▶ UPPER-MULTIPLICITY: A warning is printed in the commandline. An <code>UPPER-MULTIPLICITY</code> of <code>*</code> is written to the AUTOSAR file. ▶ ENUM-LITERAL-DEF: A warning is printed and a single <code>ENUM-LITERAL-DEF</code> with the value <code>undefined</code> is written to the AUTOSAR file.



System property	Description
	<ul style="list-style-type: none"> ▶ DEFAULT-VALUE: A warning is printed. No DEFAULT-VALUE is written to the AUTOSAR file. ▶ MIN : <ul style="list-style-type: none"> ▶ If no INVALID attribute is used in the input model, no MIN tag is written. ▶ If an INVALID attribute is used that is not of type Range, a warning is issued on the commandline and a reasonable default is written to the resulting AUTOSAR file. ▶ If an INVALID attribute of type Range is used, but no minimum value is set, no MIN tag is written. ▶ MAX : <ul style="list-style-type: none"> ▶ If no INVALID attribute is used in the input model, no MAX tag is written. ▶ If an INVALID attribute is used that is not of type Range, a warning is issued on the commandline and a reasonable default is written to the resulting AUTOSAR file. ▶ If an INVALID attribute of type Range is used but no maximum value is set, no MAX tag is written.
-DwriteInvalidXPathAttributes=(true false)	<p>With this property set to true, you can write the XPath expressions that are part of the INVALID and WARNING attributes into the ADMIN-DATA section of the corresponding schema node.</p> <p>You can use this feature to migrate the XPath expressions from EB tresos Studio to other tools.</p> <p>NOTE</p>  <p>You need to adapt these expressions manually based on the tool in which you want to use these migrated XPath expressions.</p> <hr/> <p>The default value is false.</p>
-DwriteImporterInfoAttributes=(true false)	<p>With this property set to true , you may write the IMPORTER-INFO attributes into the ADMIN-DATA section of the corresponding container node.</p> <p>The default value is false.</p>



System property	Description
-DreadImporterInfoAttributes=(true false)	<p>With this property set to true , you may read the IMPORTER-INFO attributes into the ADMIN-DATA section from the corresponding container node.</p> <p>The default value is false.</p>
-DpreserveConfiguration=(true false)	<p>With this property you may convert a pre-configuration file without changing the values configured.</p> <p>The default value is false.</p> <ul style="list-style-type: none"> ▶ To add missing nodes and overwrite default values, keep the default value at false. ▶ To not add missing nodes and not overwrite values, set -DpreserveConfiguration to true. <p>The imported configuration is not changed in any way.</p>
-DpredefinedVariant=<Name>	<p>This property enables the variant resolving and specifies the predefined Variant name that will be taken from the input files.</p> <p>If the parameter is not set, by default the variants resolving is disabled.</p> <p>For detailed information about variant handling, see Section 6.13, "Working with variants and Post-build loadable".</p>
-Dttarget=<target>	<p>Defines the target architecture, e.g. -Dttarget=WINDOWS. This is necessary when working with module configurations, which use ECU Resources. Further information on ECU Resources is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API.</p>
-Dderivate=<derivate>	<p>Defines the target's derivate, e.g. -Dderivate=Win32x86. This is necessary when working with module configurations, which use ECU Resources. Further information on ECU Resources is available in the EB tresos Studio developer's guide, chapter Configuration models/ECU Resource Manager API. If the ECU Resources require additional parameters, such as subderivate, you need to specify any additional parameter with -D<parameter-name>=<parameter-value>. Information on how to specify these parameters is available in the EB tresos Studio developer's guide, chapter ECU Resource Manager API/Characteristics when using the legacy commandline/Defining parameter values for the DefaultEcuResourceFinder.</p>



System property	Description
-DEcuResourceModuleIds=<Module1;Module2;...>	<p>Defines a comma- or semicolon-separated list of moduleIds that are necessary for finding relevant ECU Resources, e.g.</p> <pre>-DEcuResourceModuleIds= Com_TS_T99D01M01I05R01;Rte_TS_T99D01M01I04R02</pre> <p>. Information on how to define configured modules for the Resource Finder is available in the EB tresos Studio developer's guide, chapter ECU Resource Manager API/Characteristics when using the legacy commandline/Defining configured modules for the DefaultEcuResourceFinder.</p>
-DpathMapping=auto	<p>Defines that the package structure and the SHORT-NAME paths within the input file shall automatically be changed to the required ones. This option has the same effect as the check box Automatically map paths within the AUTOSAR importer mask. See Section 6.10.2.2.1, “Path mapping in import mode” for details.</p>
-DpathMapping=<sourcePath>=<targetPath>[;...][;auto]	<p>Defines paths which shall be changed when a module configuration file is read. This option has the same effect as the path mapping table within the AUTOSAR importer mask. See Section 6.10.2.2.1, “Path mapping in import mode” for details.</p> <p>You can combine this option with the automatic path mapping option if you add ;auto.</p> <p>sourcePath and targetPath must be AUTOSAR SHORT-NAME paths pointing to the SHORT-NAME path of an ECUC-MODULE-DEF (AUTOSAR schema) or an ECUC-MODULE-CONFIGURATION-VALUES (AUTOSAR module configuration)!</p>

Additional system properties when converting system description data:

System property	Description
-DecuInstance=<path>	<p>(AUTOSAR) path to an ECU-INSTANCE entity.</p> <p>This option applies if one of the following conditions is met:</p> <ul style="list-style-type: none"> ▶ An ECU configuration is created from the system description. ▶ An ECU Extract is created. <p>If this option is set, you need to also set the option -Dsystem.</p>



System property	Description
	If this option is set, you must not set the option <code>-DtoplevelComposition</code> .
<code>-DecuSwComposition=<path></code>	<p>(AUTOSAR) path to an ECU-SW-COMPOSITION entity.</p> <p>This option is only available for system description files prior to AUTOSAR 3.2.</p> <p>This option applies when an ECU Extract is created.</p> <p>You may only use this option if one of the following conditions is met:</p> <ul style="list-style-type: none"> ▶ The options <code>-DecuInstance</code> and <code>-Dsystem</code> are set. ▶ The option <code>-DtoplevelComposition</code> is set.
<code>-Decuid=<ecuid></code>	<p>This option specifies the ID of the ECU.</p> <p>The Id is used as the name of the ECU extract.</p> <p>You must use this option if you import AUTOSAR 4.x files and if one of the following conditions is met:</p> <ul style="list-style-type: none"> ▶ The option <code>-Dsystem</code> is set. ▶ The option <code>-DtoplevelComposition</code> is set.
<code>-Dsystem=<path></code>	<p>(AUTOSAR) path to the SYSTEM entity.</p> <p>This option applies if one of the following conditions is met:</p> <ul style="list-style-type: none"> ▶ An ECU configuration is created from the system description. ▶ An ECU Extract is created. <p>If this option is set, you need to also set the option <code>-DecuInstance</code>.</p> <p>If this option is set, you must not set the option <code>-DtoplevelComposition</code>.</p>
<code>-DtoplevelComposition=<path></code>	<p>(AUTOSAR) path to a COMPOSITION-TYPE entity.</p> <p>This option applies if one of the following conditions is met:</p> <ul style="list-style-type: none"> ▶ An ECU configuration is created from the system description. ▶ An ECU Extract is created. <p>If this option is set, you must not set the options <code>-DecuInstance</code> and <code>-Dsystem</code>.</p>



System property	Description
-DatpSplitableMerge=(true false)	<p>This property specifies if the stereotype splitable (3.x) / atpSplitable (4.x) is used to merge multiple system models.</p> <p>If the property is set to <code>true</code>, the <code>atpSplitable</code> stereotype is used to merge, else the old merge algorithm is used.</p> <p>The default is <code>false</code>.</p>
-DignoreOpenReferences=(true false)	<p>This property can only be used during conversion of a AUTOSAR system description. The property controls how to deal with dangling references of the system description during export.</p> <p>If the property is set to <code>true</code> all dangling references will be ignored during conversion and the target file will be created.</p> <p>If the property is set to <code>false</code>, the conversion will be canceled if the input file contains dangling references.</p> <p>The default is <code>true</code>.</p>
-DexportUUIDs=(true false)	<p>This property can only be used during conversion of a AUTOSAR system description. The property controls how to deal with UUIDs during export of an AUTOSAR system description file.</p> <p>If the property is set to <code>true</code>, UUID attributes will be exported.</p> <p>If the property is set to <code>false</code>, UUID attributes will not be exported.</p> <p>The default is <code>false</code>.</p>
-DvalidationVariant=(strict standard)	<p>This property defines the AUTOSAR XML schema variant against which the selected files are validated if there is more than one variant available (e.g. AUTOSAR 4 XML files).</p> <p>Only applies if the system property <code>Validate</code> is set to <code>true</code> (see Section 6.12.1.3, "System properties common to all commands").</p> <p>The default is <code>strict</code>.</p>
-DpackageDBC=<name>	<p>Names the AR-PACKAGE in the project system model, into which to import system parameters from a DBC file.</p> <p>The default value for this option is <code>DBCImport</code>.</p> <p>This property can only be used during conversion of a DBC file.</p>
-DimportSignalEnumsDBC=(true false)	<p>If set to <code>false</code>, Signal Enumeration Constants are not imported.</p> <p>The default value for this option is <code>true</code>.</p>



System property	Description
	This property can only be used during conversion of a DBC file.
<p>-DselectorFieldHandlingDBC=<mode></p>	<p>This property can only be used during conversion of a DBC file.</p> <p>If <mode> is set to <code>NoComSignalForSelector</code>, multiplexer signals are not mapped to Com signals.</p> <p>If <mode> is set to <code>ComSignalForSelector</code>, multiplexer signals are mapped to Com signals.</p> <p>If <mode> is set to <code>AutosarCompliant</code>, multiplexer signals are mapped to Com signals, identical segments for all dynamic parts are created, no stuffing of segments is performed.</p> <p>The default value for this option is <code>NoComSignalForSelector</code>. For detailed information on this option, see Section 6.10.2.6.4.3, “I-Pdu Multiplexer Configuration”.</p>
<p>-DpackageFIBEX=<name></p>	<p>Names the AR-PACKAGE in the project system model, into which to import system parameters from a Fibex file.</p> <p>The default value for this option is <code>FIBEXImport</code>.</p> <p>This property can only be used during conversion of a FIBEX file.</p>
<p>-DimportSignalEnumsFbx=(true false)</p>	<p>If set to <code>false</code>, Signal Enumeration Constants are not imported.</p> <p>The default value for this option is <code>true</code>.</p> <p>This property can only be used during conversion of a FIBEX file.</p>
<p>-DimportE2EInfo=(NoE2EInfo E2EInfoFromChecksumSignals)</p>	<p>If set to <code>NoE2EInfo</code>, no EndToEndProtection-related information is imported.</p> <p>The default value for this option is <code>E2EInfoFromChecksumSignals</code>. For detailed information on this option, see Section 6.10.2.8.2.2, “EndToEndProtection related information”.</p> <p>This property can only be used during conversion of a FIBEX file.</p>
<p>-DpackageLDF=<name></p>	<p>Names the AR-PACKAGE in the project system model, into which to import system parameters from a LDF file.</p> <p>The default value for this option is <code>LDFImport</code>.</p> <p>This property can only be used during conversion of a LDF file.</p>



6.12.5.1.3. Options

The `convert` command supports following options:

Parameter	Description
<code>-c <config-path></code>	By default, the directory <code><base.path>\bin</code> is used to search the module configuration files. To search the module configuration files in the directory specified as <code><config-path></code> , use the option <code>-c <config-path></code> . <code>base.path</code> is the value of the system property <code>-Dbase.path</code> . If the system property is not set, an error is displayed. For details about system properties, see Section 6.12.1.3, “System properties common to all commands” .
<code>-n <config-name></code>	Name of the module configurations of the configuration files to load. Use this option to ignore other module configurations contained in the files specified. You may use this option multiple times.
<code>-u</code>	Use this option to generate files with Unix line endings.

6.12.5.1.4. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, the `convert` command also recognizes the following parameters:

Parameter	Description
<code><input-file>[@<type>]</code>	Use <code><input-file></code> for the name of the file to read in. The parameter may contain wildcards to match multiple files with just one argument, e.g. <code><path>/*.epc</code> . Note that no wildcard characters are supported in <code><path></code> itself. You can pass the parameter multiple times to load more than one file. To explicitly specify the content type of the file, use the optional argument <code>@<type></code> . For more information on which content types are available, see Section 3.5.3, “File content types” .
<code><output-file>[@<type>]</code>	Use <code><output-file></code> for the name of the file to write. To explicitly specify the content type of the file, use the optional argument <code>@<type></code> . For more information on which content types are available, see Section 3.5.3, “File content types” .



6.12.5.1.5. Example

The following are several examples of how to use the command `legacy convert`:

```
tresos_cmd.bat -DValidate=false -DMapOptionalAsList=false
    legacy convert in.epd out.xdm
```

The example above simply converts an EPD file to an XDM file and marks optional parameters with the optional attribute instead of creating lists around them.

```
tresos_cmd.bat -DValidate=false -DMapOptionalAsList=false
    -Duuids=true legacy convert
    in1.epd in2.epd in3.xdm out.epc@asc:4.0.3
```

This example loads the three files `in1.epd`, `in2.epd` and `in3.xdm`. The content of the files is merged and saved to the output file called `out.epc`. The content of the output file conforms to AUTOSAR 4.0.3. No validation is done during the conversion and new UUIDs are created for each node.

```
tresos_cmd.bat -Dsystem="/SYSTEM/system"
    -DecuInstance="/AUTOSAR/ECUInstance/ecuInstance_1"
    legacy convert
    in1.xml@fibex in2.arxml@sysd out.tdb
```

This example loads the files `in1.xml` (importing the file using the Fibex Importer) and `in2.arxml` (importing the file as AUTOSAR system description) and merges the content of the two files. The system `/SYSTEM/system` and the ECU `/AUTOSAR/ECUInstance/ecuInstance_1` must both be contained in the file `in2.arxml`). Both are selected for import. All software components, `EcuSwComposition`s, data mappings and service port mappings are retained in the system model. The resulting system model is saved to the output file called `out.-tdb` using the tresosDB file format.

```
tresos_cmd.bat -DValidate=false -DignoreOpenReferences=false
    legacy convert System20.arxml@sysd System314.arxml@sysd:3.1.4
```

The example converts an AUTOSAR system description in version 2.0 into an AUTOSAR system description in version 3.1.4. If the source file contains dangling references the file will not be converted (`-DignoreOpenReferences=false`).



6.12.5.2. Using XDM files in different EB tresos Studio versions

If you are working with several releases of EB tresos Studio on the same configuration data, you may want to use an XDM created with a new version of EB tresos Studio in your old installation. Newer versions of EB tresos Studio are always able to read old XDM files, but they may add new features to the XDM format that cannot be understood by older versions.

It is possible to use such a file with an older EB tresos Studio version. But you have to convert that file first into the older version. During that conversion, information that cannot be mapped into the old format may get lost!

NOTE The XDM versions numbers are not related to the AUTOSAR version numbers.



To convert a newer version of the XDM format to the older version, use the `legacy convert` command together with the EB tresos Studio installation with which you created the file.

The `legacy convert` command is explained in [Section 6.12.5.1, “Merging files and converting between different file formats”](#). All you need to do to convert between XDM file versions is to specify the correct content type for the files. The available content types are listed in [Section 3.5.3, “File content types”](#). You can use any `xdm:*` content type.

6.12.5.2.1. Example

```
tresos_cmd.bat legacy convert new.xdm old.xdm@xdm:2.0
```

This example converts an XDM from a new EB tresos Studio version to the older XDM 2.0 format.

6.12.5.3. Upgrading the AUTOSAR compliance of XDM files

The XDM files contain a representation of the AUTOSAR data structures. Sometimes it is necessary to convert that representation to a newer AUTOSAR revision because the newer revision contains incompatible changes. E.g. within AUTOSAR 2.x choice containers do not have a SHORT-NAME, but in AUTOSAR 3.x they do. Instead of editing these changes manually inside the XDM file you may use the `legacy xdmconvert` command.

6.12.5.3.1. Syntax

The syntax of the command is shown in the following example:



```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    legacy xdmconvert <mode>
    <input-file>[@<type>]... <output-file>[@<type>]
    [-variant <variant>]
```

6.12.5.3.2. System properties

This command only recognizes the system properties common to all commands. For details about common system properties, see [Section 6.12.1.3, “System properties common to all commands”](#).

6.12.5.3.3. Parameters

In addition to the common parameters `<system_property>` and `<workspace>`, this command also recognizes the following parameters:

Parameter	Description
<code><input-file>[@<type>]</code>	Use <code><input-file></code> for the name of the file to read in. The parameter can be passed multiple times to load more than one file. The optional argument <code>@<type></code> can be used to explicitly specify the content type of the file. See Section 3.5.3, “File content types” for more information on what content types are available.
<code><output-file>[@<type>]</code>	Use <code><output-file></code> for the name of the file to write. To explicitly specify the content type of the file, use the optional argument <code>@-<type></code> . For more information on which content types are available, see Section 3.5.3, “File content types” .
<code><mode></code>	Converts an XDM file from one AUTOSAR version to another. Currently, two modes are supported: <code>asc2xToAsc30</code> and <code>asc2xToAsc31</code> . They convert an XDM file (representing an AUTOSAR 2.x file) to an XDM file (representing an AUTOSAR 3.0 or 3.1 file).
<code>-variant <variant></code>	This parameter defines which SUPPORTED-CONFIG-VARIANT needs to be set at the MODULE-DEF. This parameter is optional. Possible values are: <ul style="list-style-type: none"> ▶ <code>pre</code> for a pre-compile time configuration, ▶ <code>link</code> for a link-time configuration, and ▶ <code>post</code> for a post-build configuration.



Parameter	Description
	For further information, see the EB tresos Studio developer's guide, chapter Developing modules/How to upgrade an AUTOSAR 2.x module to 3.x.

6.12.6. Executing multiple commands (batch mode)

Each command described in the previous chapters has to start EB tresos Studio once to be executed. When you execute many commands, this may be very time consuming. This section describes how any number of commands can be executed while you start EB tresos Studio just once.

To use the command `batch`, use the following syntax:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    batch [-continueOnError] <batch file>
```

The specified `batch file` may contain any number of EB tresos Studio commands. The commands are executed in the order as they appear in the file. EB tresos Studio will resolve a relative path to the `batch file` with respect to the workspace folder.

By default, if an error occurs, EB tresos Studio will stop processing the commands from a batch file. If you set the `-continueOnError` command line switch, the batch mode execution will continue, even if errors occur.

6.12.6.1. Batch file syntax

The `batch file` may contain any number of EB tresos Studio commands with the same syntax as if you execute them directly at the command line. The `tresos_cmd.bat` command may be left out.

Each of the commands may redirect its standard output and standard error streams to dedicated files. In order to redirect a stream, one or more redirection operators followed by an absolute file path or by a relative file path with respect to the workspace folder may be appended to the command. The following redirection operators are supported:

- ▶ `> <filepath>`: Redirect standard output to `<filepath>`.
- ▶ `>> <filepath>`: Redirect standard output to `<filepath>`, the stream will be appended to existing content in `<filepath>`.
- ▶ `2> <filepath>`: Same as `> <filepath>` for standard error.
- ▶ `2>> <filepath>`: Same as `>> <filepath>` for standard error.
- ▶ `&> <filepath>`: Same as `> <filepath>` for standard output and standard error.



- ▶ &>> <filepath>: Same as >> <filepath> for standard output and standard error.

The `batch` file may contain any number of empty lines.

A line comment starts with the character '#'. Everything in the same line after that character is ignored.

A line-break is marked with the characters \n at the very end of a line and is - together with the line-break itself - automatically replaced by a space.

Example of a `batch` file:

```
legacy convert -DValidate=true input.epc output.xdm # line comment
legacy convert -DValidate=false input1.epc\n
                      input2.epc\n
                      output2.xdm

# uses the Validate property set for the batch command (or its default value)
# the standard output stream is redirected to <workspace-folder>\info.txt,
# the standard error stream is redirected to c:\error.txt
legacy convert input3.epc output3.xdm >info.txt 2>c:\error.txt

# comment line preceded by an empty line
legacy generate -n Com\n
                  -g Com_TS_T16D4M2I0R0\n
                  -g Custom\n
                  -u All.epc@asc:4.2.1
```

6.12.6.2. Handling of system properties

The system properties specified with the `batch` command itself are used as the default value for all executed commands of the `batch` file. Each command may define its own system properties for the time of its execution.

After a command has been executed, all system properties which has been set for this command only will be re-set to their value prior the commands execution.

The parameter `-data <workspace>` can only be defined by the `batch` command itself. It is ignored inside the file and cannot be overwritten by a single command.

6.13. Working with variants and Post-build loadable



6.13.1. Variant handling concept by AUTOSAR

AUTOSAR defines variant handling with the following terminology:

[TPS_GST_00174] Variant Handling Terminology

- ▶ **Variation points** are locations in the model that are variable. That is, they may not exist in all variants, or may have different characteristics in different variants.
- ▶ The **binding time** is the latest possible time when a variation point may be bound.
- ▶ **Binding expressions** specify under which condition(s) a variable element exists, or determine certain variable characteristics.

AUTOSAR differentiates between PreBuild and PostBuild variation points.

- ▶ PreBuild variation points support the variability of several ECUs.
- ▶ PostBuild variation points support variants on one ECU.

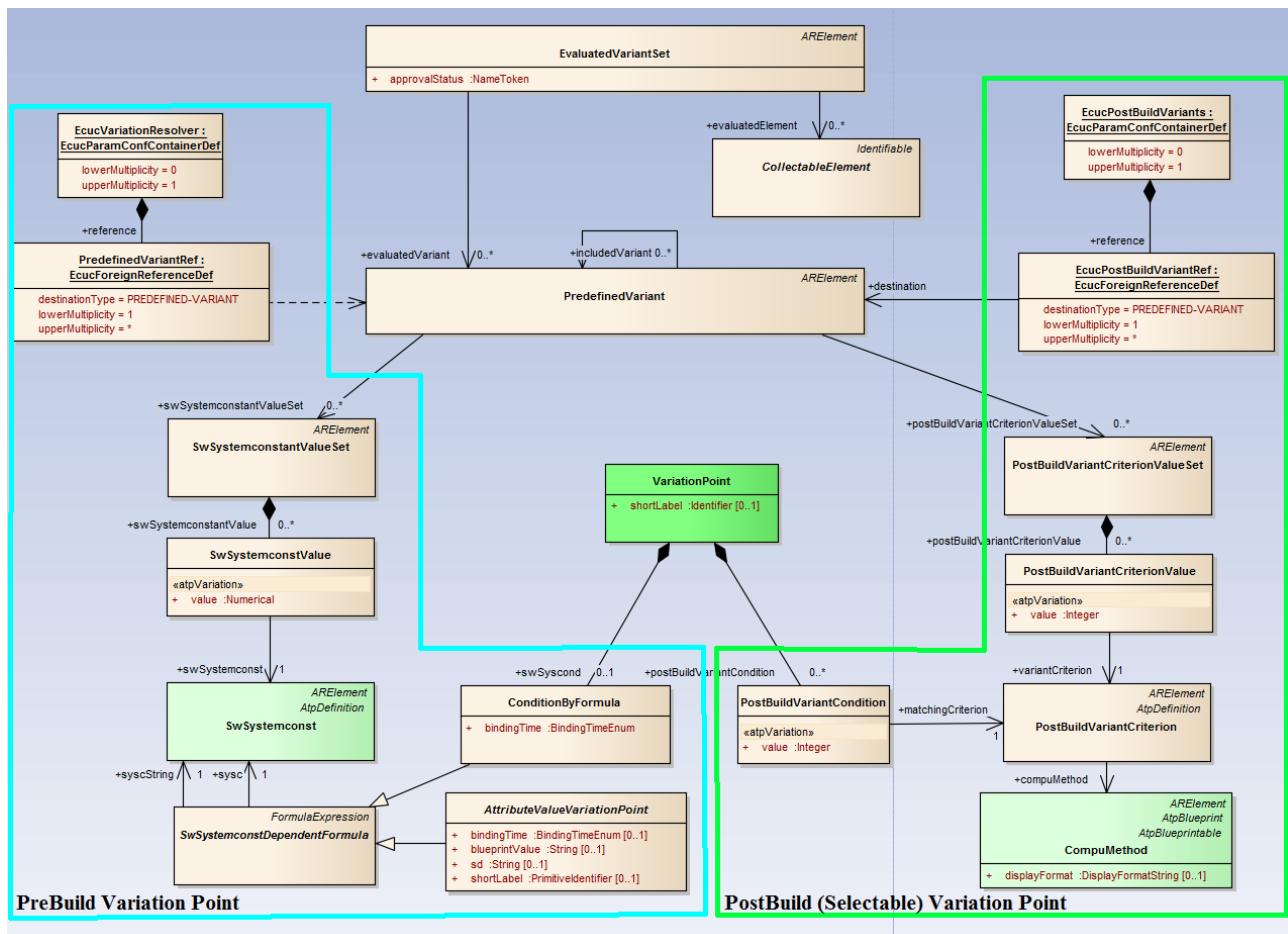


Figure 6.172. Variation point overview



This figure is based on the following AUTOSAR requirements:

[TPS_GST_00245] PreBuild variation point: If a variation point aggregates ConditionByFormula, then this variation point is a PreBuild variation point.

[TPS_GST_00246] PostBuild Variation Point: If a variation point aggregates a PostBuildVariantCondition, then this variation point is a PostBuild variation point.

[TPS_GST_00248] Combined PreBuild and PostBuild Variation Point: A variation point may also aggregate both ConditionByFormula and Post- BuildVariantCondition. In this case, it is both a PreBuild and a PostBuild variation point. . . “This is because the PreBuild and PostBuild branches of VariationPoint are not mutually exclusive”.

[TPS_GST_00266] PreBuild Disabling PostBuild support: It is possible to define a variation point as both PreBuild and PostBuild. If the PreBuild condition is false, it is not expected that the variant object (including the PostBuild condition) will be implemented. In other words, a system constant expression in a variation point may be used to disable the PostBuild variability even at PreBuild time.

[TPS_GST_00255] Definition of PreBuild Variation Point: This category contains the following binding times: systemDesignTime, codeGenerationTime, preCompileTime, and linkTime. A concrete variation point (i.e., on M1 level) is subject to PreBuild variation if it contains a ConditionByFormula element. Its binding time is specified in the bindingTime attribute of the ConditionByFormula element.

[TPS_GST_00256] Definition of PostBuild Variation Point: This category contains only a single binding time, namely PostBuild. A concrete variation point (i.e., on M1 level) is PostBuild if it contains a PostBuildVariantCondition element. Since there is only one binding time for PostBuild, no particular attribute for specifying the binding time is necessary.

6.13.1.1. PreBuild variation points

PreBuild variant points which use ConditionByFormula are bound at PreBuild time.

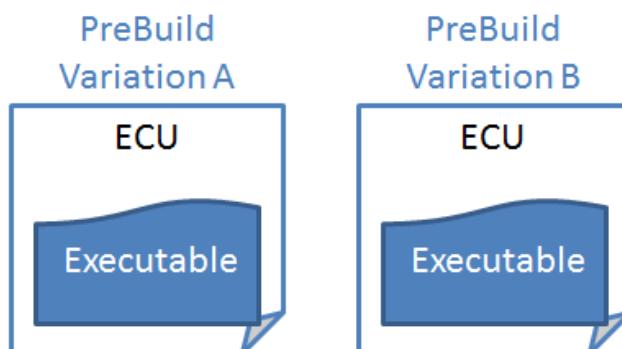


Figure 6.173. PreBuild variation point concept

NOTE**PreBuild variation points are not supported by EB tresos Studio**

Currently EB tresos Studio does not support PreBuild variation points. PreBuild and PostBuild variation points are resolved and bound during system description import. The EB tresos Studio system model is variant-free and therefore contains exactly one bound variant.

For more information on how to resolve PreBuild Variation points in system model in EB tresos Studio, see [Section 6.13.3.6, “Resolving variants”](#).

6.13.1.2. PostBuild variation points

EB tresos Studio supports the following concepts for PostBuild variation points using PostBuildVariantCondition elements.

6.13.1.2.1. PostBuild loadable concept

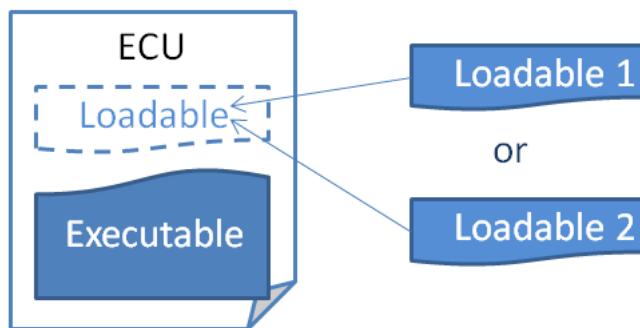


Figure 6.174. Loadables for an ECU

The picture above visualizes the use case of the post-build loadable (PBL) concept. There, an ECU executable is flashed to an ECU which is completed by any loadable configuration. For this use case, it is not necessary to re-compile or re-build the BSW module since the loadable only contains configuration data which is stored at a known memory location. Therefore, this concept can e.g. be used by the OEM to use the same ECU within different cars with just a slightly changed (loadable) configuration.

A parameter which can have loadable variants needs to specify MULTIPLICITY-CONFIG-CLASSES and/or VALUE-CONFIG-CLASSES for POST-BUILD. This is done by the module developer in the vendor-specific module definition and cannot be changed by the user.

Prior to AUTOSAR 4.2. releases this concept was handled with multiple configuration containers (MCCs). For more information, see [Section 6.14, “Post-build loadable configuration using multiple configuration containers”](#). Now this concept is represented by EB tresos Studio using PostBuild variation points similar to the PostBuild selectable concept.



6.13.1.2.2. PostBuild selectable concept (PostBuildVariants)

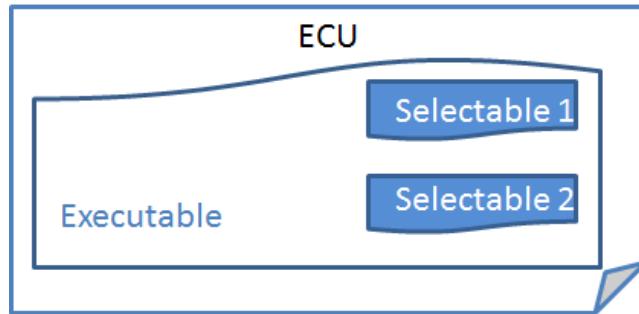


Figure 6.175. PostBuildVariant concept

The picture above visualizes the post-build selectable concept. An ECU executable contains the static code and all selectable data structures. The executable which is flashed to the ECU always contains all selectables. During runtime, most probably through the configuration of a hardware switch it is decided which selectable is used.

Only parameters marked as variant aware by the module developer can have selectable variants. You can search for these parameters using the EB tresos Studio search mask.

PostBuildVariants are designed by AUTOSAR using the PostBuild variation point.

6.13.1.2.3. Combined PostBuild loadable and selectable concept

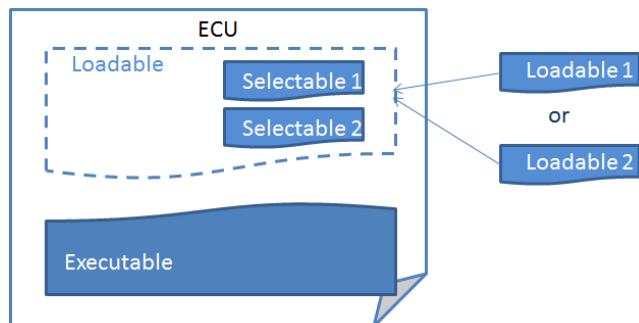


Figure 6.176. Combined loadable and selectable concept

The picture above visualizes the combination of the loadable and selectable concept. The loadable can itself contain several selectables.

EB tresos Studio covers PostBuild selectable and also the PostBuild loadable concept in a similar way. So it is easy to also handle the combined use case.



6.13.1.3. Restrictions on variant handling in EB tresos Studio

EB tresos Studio provides a first set of features for variant handling. These features will be enhanced in future releases of EB tresos Studio.

Here you find a list of restrictions for variant handling in EB tresos Studio:

- ▶ The system model in EB tresos Studio is variant-free

Currently EB tresos Studio only supports PostBuild variation points for the ECU configuration.

WARNING
In EB tresos Studio the system model is variant-free


The system model is not afflicted with variants. When you import system data via the System description importer you can select which variant of your input file you want to bind. This variant is resolved and the system data is stored in the system model.

Assuming that you have several input files for different parts of the system model (atpSplittable), each file contains system data for several PreBuild variants:

If you run several importers subsequently to import all parts of the system data with the merge option, ensure to always import the same or compatible variant.

The ECU configuration can store several variants and you can switch between these variants. If there are variant-affected parts in the system model that have an impact on the ECU configuration you must always work on the same variant in both System Model and ECU configuration.

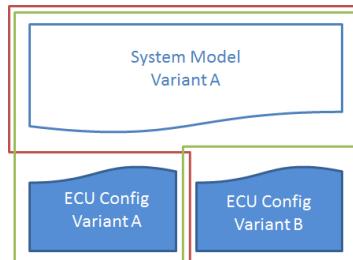


Figure 6.177. Always work on the same variant in system model AND data model

- ▶ At the moment not all parts or features of EB tresos Studio can deal with several variants. In EB tresos Studio you select the variant you want to work on. EB tresos Studio provides a view or filter for the currently selected variant. All actions work on exactly this selected variant.

**NOTE****Variant-aware code generators can handle several variants**

Currently only code generation can deal with several variants. But only code generators registered and marked as variant-aware can handle several variants.

For all registered variant-aware code generators the code generator framework loops over all existing variants and generates code for each variant.

6.13.1.4. Fast access tool bar

If one or more **EcucPostBuildVariantRef** instances are specified in the EcuC module, then a combo box appears in the tool bar of EB tresos Studio where you can switch between these selectable variants.

You can only switch between loadable variants in the EcuC module.

6.13.2. Prerequisites

6.13.2.1. Module supporting variant selection

To use variant handling in EB tresos Studio you need one module that supports the selection of variants. Elektrobit Automotive GmbH (EB) e.g. provides this functionality within the EcuC module.

The module must contribute to the `dreisoft.tresos.autosar2.api.plugin.postbuildSetup` extension point. If no such module part of your project, the variant handling support is not available for any module which is part of the project.

6.13.2.2. Variant definitions

Probably you receive the variant definition from the OEM as a `.arxml` file which you can import into your project with a system description importer.

You can also create the variant definitions in EB tresos Studio. For information on how to manage variant definitions, see [Section 7.7, “Using the Edit PostBuildVariants wizards”](#).

In this chapter an example of a valid variant definition is shown. The following parts need to be defined:

1. PostBuildVariantCriterion

At least one PostBuildVariantCriterion needs to be defined that refers a CompuMethod.



```
<POST-BUILD-VARIANT-CRITERION>
  <SHORT-NAME>EngineTypeCriterion</SHORT-NAME>
  <COMPU-METHOD-REF DEST="COMPU-METHOD">
    /EB/VariantDefinitions/EngineTypeDescription
  </COMPU-METHOD-REF>
</POST-BUILD-VARIANT-CRITERION>
```

2. CompuMethod

The CompuMethod contains the textual representation for the available PostBuildVariantCriterionValues.

```
<COMPU-METHOD>
  <SHORT-NAME>EngineTypeDescription</SHORT-NAME>
  <CATEGORY>TEXTABLE</CATEGORY>
  <COMPU-INTERNAL-TO-PHYS>
    <COMPU-SCALES>
      <COMPU-SCALE>
        <LOWER-LIMIT>0</LOWER-LIMIT>
        <UPPER-LIMIT>0</UPPER-LIMIT>
        <COMPU-CONST>
          <VT>gasoline</VT>
        </COMPU-CONST>
      </COMPU-SCALE>
      <COMPU-SCALE>
        <LOWER-LIMIT>1</LOWER-LIMIT>
        <UPPER-LIMIT>1</UPPER-LIMIT>
        <COMPU-CONST>
          <VT>diesel</VT>
        </COMPU-CONST>
      </COMPU-SCALE>
    ...
    </COMPU-SCALES>
  </COMPU-INTERNAL-TO-PHYS>
</COMPU-METHOD>
```

3. PredefinedVariant

The definition of the PredefinedVariant which refers minimum one PostBuildVariantCriterionValueSet.



```
<PREDEFINED-VARIANT>
  <SHORT-NAME>Diesel</SHORT-NAME>
  <POST-BUILD-VARIANT-CRITERION-VALUE-SET-REFS>
    <POST-BUILD-VARIANT-CRITERION-VALUE-SET-REF DEST="POST-BUILD-VARIANT-CRITERION-VALUE-SET">
      /EB/VariantDefinitions/CriterionValueSet_Diesel
    </POST-BUILD-VARIANT-CRITERION-VALUE-SET-REF>
  </POST-BUILD-VARIANT-CRITERION-VALUE-SET-REFS>
</PREDEFINED-VARIANT>
```

4. PostBuildVariantCriterionValueSet

The ValueSet itself contains a set of PostBuildVariantCriterionValues. Each PostBuildVariantCriterionValue consists of a reference to one PostBuildVariantCriterion and the used integer value.

```
<POST-BUILD-VARIANT-CRITERION-VALUE-SET>
  <SHORT-NAME>CriterionValueSet_Diesel</SHORT-NAME>
  <POST-BUILD-VARIANT-CRITERION-VALUES>
    <POST-BUILD-VARIANT-CRITERION-VALUE>
      <VARIANT-CRITERION-REF DEST="POST-BUILD-VARIANT-CRITERION">
        /EB/VariantDefinitions/EngineTypeCriterion
      </VARIANT-CRITERION-REF>
      <VALUE>1</VALUE>
    </POST-BUILD-VARIANT-CRITERION-VALUE>
  </POST-BUILD-VARIANT-CRITERION-VALUES>
</POST-BUILD-VARIANT-CRITERION-VALUE-SET>
```

This example shows the definition of a variant for all diesel vehicles.

6.13.3. Use cases

6.13.3.1. Configuring variants for a project

The AUTOSAR standard defines that the EcuC module contains the list of selectables.

The EcuC module in the following example contributes to the `dreisoft.tresos.autosar2.api.plugin.postbuildSetup` extension point and provides the possibility to configure the available variants and manage the active variant.

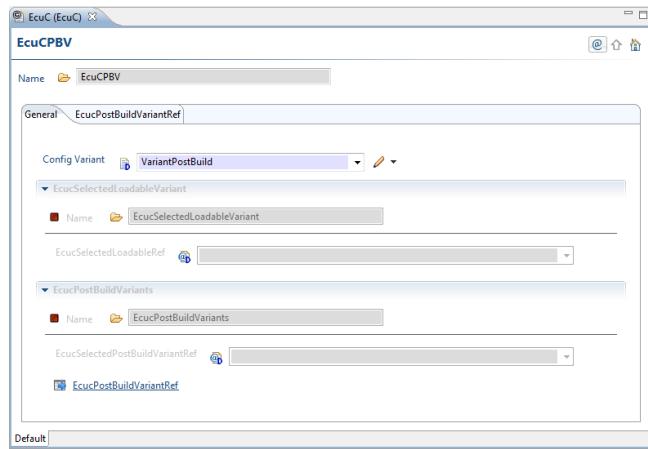


Figure 6.178. Example of a variant configuration in the EcuC module

You can either do one of the following:

- ▶ Select a loadable variant
- ▶ Select a selectable variant
- ▶ Select both loadable variant and selectable variant where the selected selectable variant is dependent on the selected Loadable variant.

To be able to specify the active selectable variant, you first need to configure the available selectables in the list of **EcucPostBuildVariantRef** elements. These entries appear in the **EcucSelectedPostBuildVariantRef** combo box.

Variant-aware code generators generate code for all available selectables.

You can change the active selectable for a project as follows:

- ▶ Use the **EcucSelectedPostBuildVariantRef** combo box in the EcuC module.
- ▶ Use the fast access tool bar in EB tresos Studio.
- ▶ Use the unattended **Switch PostBuildVariant** wizard.

6.13.3.2. Editing PostBuildVariants

For information on how to edit PostBuildVariants, see [Section 7.7, “Using the Edit PostBuildVariants wizards”](#).

6.13.3.3. Editing textual descriptions for PostBuildVariantCriterions

It is difficult to remember which integer value for PostBuildVariantCriterions has which meaning. Therefore, you can add and edit textual descriptions for these values. For more information, see [Section 7.7, “Using the Edit PostBuildVariants wizards”](#).



6.13.3.4. Editing variant conditions

For more information on how to change variant conditions for ECU configuration parameters, see [Section 6.8.5, “Editing variant conditions”](#).

6.13.3.5. Looking up variant conditions of a parameter

You can look up the variant conditions of a parameter in the following locations:

- ▶ The **PostBuildVariantConditions** dialog. For more information, see [Section 6.8.5, “Editing variant conditions”](#).
- ▶ The **Properties** view. For more information, see [Section 5.3.9, “Properties view”](#).

6.13.3.6. Resolving variants

6.13.3.6.1. Resolving variants example: commandline

TIP

You may enable variant resolving by using the system property: -DpredefinedVariant=<Name>.



You can use this property with the following commands:

- ▶ legacy convert
- ▶ legacy generate
- ▶ legacy make
- ▶ legacy verify

You can use the following example on the command line:

```
tresos_cmd.bat -DpredefinedVariant=/Variants/Basic legacy convert  
    inputFile@sysd outputFile@sysd:4.0.3
```



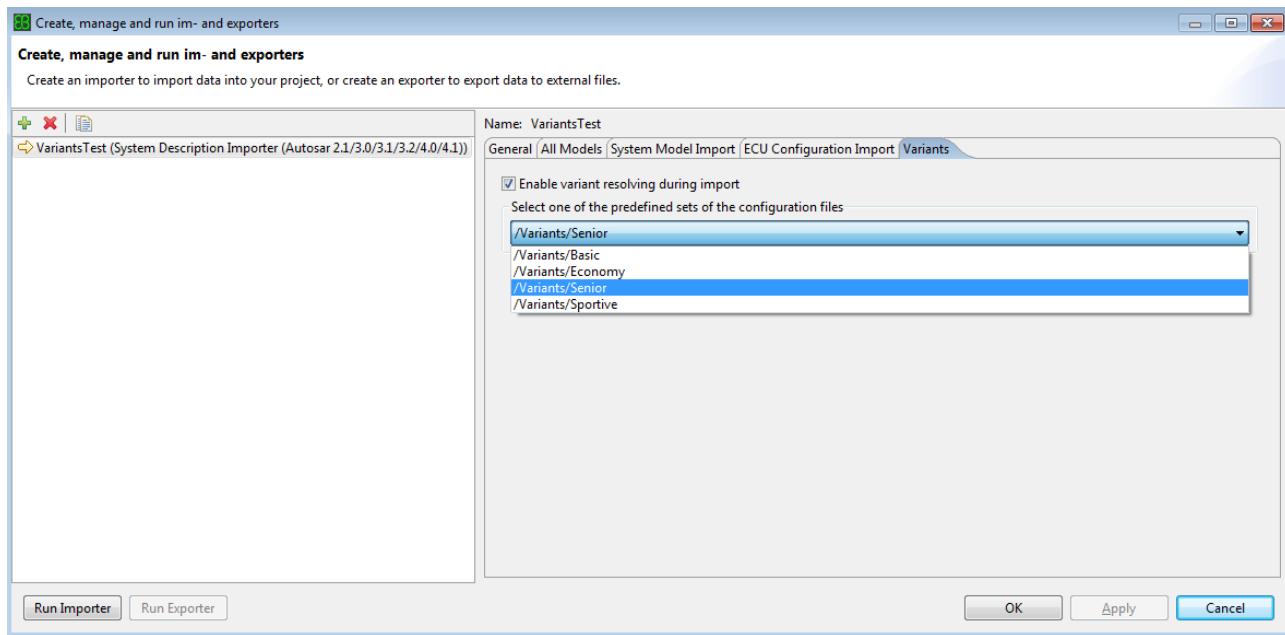
6.13.3.6.2. Resolving variants example: Im- and Export wizard

TIP



Variants resolving with the Im- and Export wizard

You can enable variants resolving with the Im- and Export wizard. You can configure the variants resolving in the **Variants** tab.



For more details about the **Variants** tab, see [Section 6.10.2.3.5, “Variants tab”](#).

6.13.3.7. Handling variants in legacy generate

To use input files that contain variants at the command prompt using `legacy generate`, you have to provide the input files in the following order:

- ▶ The tdb file or an ARXML file containing the predefined variants.
- ▶ The EcuC config file that contains the selected and active variants.
- ▶ The config files containing variants.

Example:

```
tresos_cmd.bat legacy generate systemData.arxml@sysd ecuc.arxml@asc
config1.arxml@asc config2.arxml@asc
```



6.13.4. Recommended workflow

As variant handling is a complex topic, this chapter describes the recommended workflow in EB tresos Studio.

NOTE**Automatically added variant conditions to nodes**

If a variant-affected configuration parameter is added to the data model, e.g. via import, via user interaction, e.g. by adding an element to a list, or by adding a module to the project, then this parameter always gets the variant conditions of the currently selected variant. If no variant is selected, then no variant conditions are added.

Variant-affected parameters without variant conditions are available in all variants.

6.13.4.1. Setting up a variant project

If you want to work with variants, your project needs to be prepared in an early state.

1. Creating a project with the EcuC module

First create a project for your target and derivate and only add the EcuC module. In the EcuC module you can configure which selectables are feasible for this project and you can choose the currently selected selectable. As EB tresos Studio handles the loadable and selectable concept in the same way, you can also select the current loadable variant.

2. Importing or creating variants

Either import the predefined variants, e.g. using a system description importer or just create these variants with the **Edit PostBuildVariants** wizard from the **Sidebar** view.

Post-build loadable variants can only be created in EB tresos Studio using the **Edit PostBuildVariants** wizard.

3. Selecting the current variant in the EcuC module

Open the EcuC module and select which selectables are feasible for this project and decide on which variant you want to work.

You can change the variant any time in EB tresos Studio. But if you do not choose a variant for the first time, variant-affected parameters do not get a variant condition when parameters are added. And in this case the variant-affected parameters are therefore available in all variants and you must specify the variant condition by hand for each variant-affected parameter.

4. Adding required modules

Add all further required modules to your project. Each variant-affected parameter receives the variant conditions of the current variant.



To look up variant conditions for a specific parameter, select the parameter either in the **Outline** view or in the generic configuration editor. Then the **PostBuildVariantConditions** tab in the **Properties** view shows a list of all variant conditions.

6.13.4.2. Working with post-build selectable variants

If your project involves complex variant handling and you have a dedicated system description for each variant, we strongly recommend to use the Variant Handling wizard. This wizard eases the transfer of multiple variants into the ECU configuration. For configuring the Variant Handling wizard, see [Section 7.8, “Using the Variant Handling wizard”](#).

6.13.4.2.1. Prerequisites

- ▶ The project has variants configured (see [Section 6.13.3.1, “Configuring variants for a project”](#)).
- ▶ You provided an artificial master variant.

The artificial master variant deals with the fact that the EB tresos AutoCore Generic 8 RTE does not support variant handling. It is a mandatory, EB-specific fragment to get a complete ECU extract. The artificial master variant is an additional system description. It must contain a superset of system signals and all elements related to their connection.

Contact EB if you need support creating an artificial master variant system description.

NOTE**No impact on BSW module code generation**

The artificial master variant is mainly used to generate a valid RTE and Com (interface for RTE) configuration. It has no impact on the code generation of the basic software modules.

6.13.4.2.2. Workflow

1. Create a system description importer configuration for each variant in your project. Enable each time the checkbox **Enable variant resolving during import**.
2. Create a system description importer configuration for the artificial master variant.
3. Create a configuration for the unattended wizard **Create ECU Configuration(ImportEcuConfig)**. Enable the checkbox **Overwrite project settings** and select your **System** and **EcuInstance**.
4. Configure and run the Variant Handling wizard (see [Section 7.8, “Using the Variant Handling wizard”](#)).



5. Optionally, run additional wizards that work on the system model.
6. Optionally, add service component prototypes with the Connection Editor.
7. Run the unattended wizard **Create an ECU Extract(EcuExtractCreator)** to create the ECU extract.
8. Manually adapt the ECU configuration according to your needs.
9. Remove your artificial master variant from the EcuC module configuration (**EcuC > Post Build Variants > List of Post-Build-Selectable Variants**).
10. Generate the source code.

6.13.4.3. Working with post-build loadable variants

As described in [Section 6.13.1.2.1, “PostBuild loadable concept”](#), there is always only one loadable at an ECU at a time. There might be an unknown number of loadable configurations in the future and therefore only the currently selected loadable configuration is enforced to be valid. According to AUTOSAR, there is no possibility to store multiple loadable configurations in one configuration. This would enforce you to copy an existing configuration and change it, to create a new loadable without losing the existing loadable configurations.

Nevertheless, in EB tresos Studio you are able to keep and configure all your loadable configurations in one project. But there is always only one of those loadable configurations selected. This is the one you are working on with all editors, importers, wizards etc.

1. Configure the **IMPLEMENTATION_CONFIG_VARIANT VariantPostBuild** for your module.
2. Configure the ECU configuration

You can assign a container or parameter to a certain loadable configuration if the module developer defined the container or parameter to have config class `PostBuild`. Further information is available in the EB tresos Studio developer's guide chapter **Marking elements as Post-build loadable variant aware**.

For more information on how to change variant conditions for ECU configuration parameters, see [Section 6.8.5, “Editing variant conditions”](#).

Contrary to the selectable use case, only the currently selected loadable variant must be valid. You always work on exactly one loadable variant.

3. Generate code

When you generate code - contrary to the selectable use case - only the selected loadable variant is generated. You explicitly have to switch to another loadable variant to configure and generate it.

There is no fast access toolbar to switch between the loadable variants because this is usually not done very often.

NOTE**It is currently possible to create an incorrect post-build configuration**

It is currently possible to delete or rename a post-build container at post-build time if it is not directly referenced by a precompile or link time reference. It is also possible to modify any parameter inside such a container because it is assumed that such containers are not used from precompile or link time generators.

This behavior is wrong. Instead, only containers and parameters introduced at post-build time shall be changeable. Unfortunately, this is only clarified by AUTOSAR in release 4.3 in chapter **Post-build Time Consistency** of the **Specification of ECU Configuration** as discussed in the AUTOSAR Bugzilla change request 70301.

Consequently, it is currently possible to create an incorrect post-build configuration!

6.14. Post-build loadable configuration using multiple configuration containers

NOTE**Concept using multiple configuration containers is replaced by new variant handling concept**

The concept using MCCs was removed with AUTOSAR 4.2. and is replaced by a new concept using variation points. For more information, see [Section 6.13.1.2.1, “PostBuild loadable concept”](#).

6.14.1. General post-build loadable concept

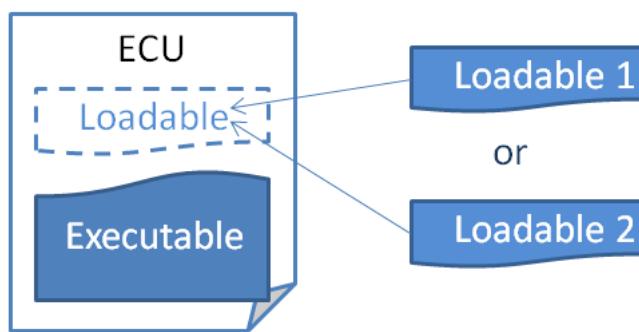


Figure 6.179. Loadables for an ECU

The picture above visualizes the use case of the post-build loadable (PBL) concept. There, an ECU executable is flashed to an ECU which will later on be completed by any loadable configuration. For this use case, it is not necessary to re-compile or re-build the BSW module since the loadable only contains configuration data which

will be stored at a known memory location. Therefore, this can e.g. be used by the OEM to use the same ECU within different cars with just a slightly changed (loadable) configuration.

6.14.2. Using Multiple Configuration Containers for post-build loadables

To configure multiple post-build loadables within one configuration project, so-called *Multiple Configuration Containers* (MCCs) may be used where each MCC represents one loadable configuration. The developer of the module (and its code generator) selectively decided which parameters of the MCC may be changed at post-build time and are part of the loadable. Therefore, after changing the configuration time of the project to *post-build only* (see chapter [Section 3.5.2.1, “Configuration time of projects”](#)), only those parameters can still be changed.

6.14.3. Defining valid sets of *Multiple Configuration Containers*

Since you may configure multiple MCCs for each module of the project, it is necessary to define which MCCs throughout all modules belong to one post-build loadable configuration. This can be configured using the generic configuration editor of EB tresos Studio of one "Post-build configuration Management" module (PbcfgM).

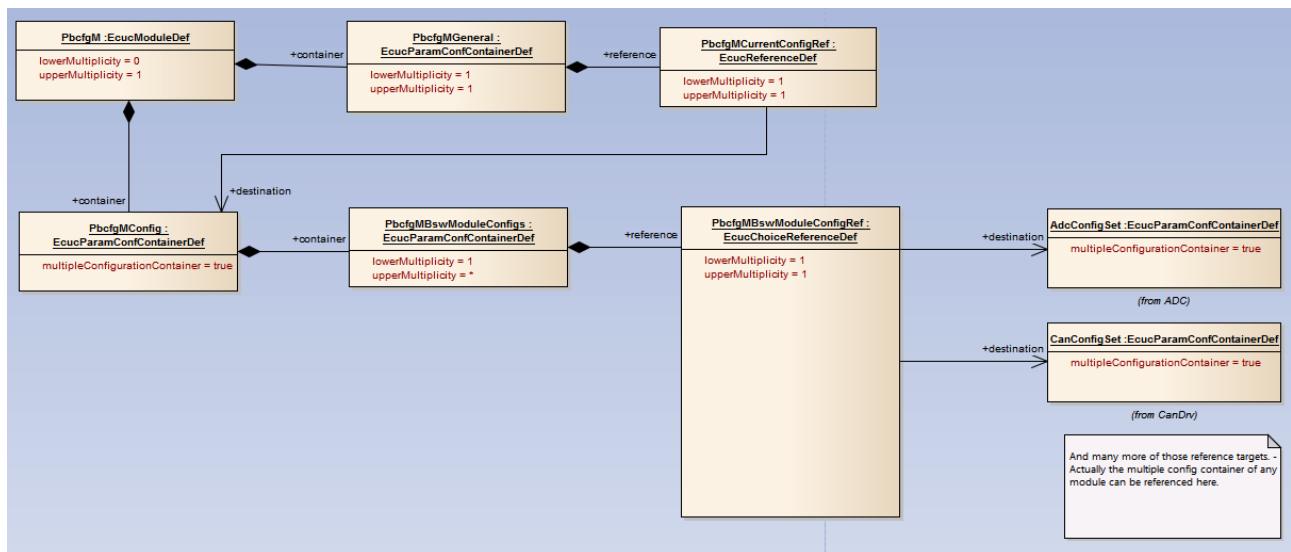


Figure 6.180. PbcfgM module

As shown in the picture above, the PbcfgM module contains a MCC **PbcfgMConfig** with a container **PbcfgMBswModuleConfigRef** which contains references to the corresponding MCC of each module. Through this mechanism, each MCC of the PbcfgM defines one set of MCCs which belong to one post-build loadable configuration. If there are e.g. two loadable configurations defined within one project, there must be two MCCs defined within the PbcfgM module accordingly.



As you can see in the picture, there also is a reference called "PbcfgMCurrentConfigRef" which selects one MCC of the PbcfgM module - and with this a complete post-build loadable configuration. This reference is required to be able to decide which is the "current set of MCCs". The following sub-chapters describe what this is used for.

TIP

For technical reasons, the "current MCC" of a module is automatically moved to the first position within its list of MCCs if the configuration of the PbcfgM changes.



6.14.3.1. Generating code

When generating code at post-build time, the code-generators will only generate code for the configured "current set of MCCs". To generate code for another post-build loadable configuration, the reference "PbcfgMCurrentConfigRef" must be configured properly.

6.14.3.2. Importing a system description

When importing a system description, the ECU configuration is automatically updated (if "ECU configuration import" is enabled within the importer preferences). Since an import logically belongs to exactly one ECU loadable only, this is always done for the "current set of MCCs" only. To import a system description into another post-build loadable configuration, the reference "PbcfgMCurrentConfigRef" must therefore be configured properly.

6.14.3.3. Guided Configuration wizards

Although a Guided configuration wizard can be implemented to take all existing MCCs of a module into account, it also may only process one specific MCC per module (e.g. the Handheld wizard). Therefore it may be required to configure the reference "PbcfgMCurrentConfigRef" properly. Consider the description of the specific wizard for details.

6.14.4. Value Constraints

6.14.4.1. Readonly

When you change the config time of a project to **Post Build** only, most of the pre-build parameters (i.e. parameters with configuration class **Pre Compile** or **Link**) become read-only. The exception of this rule is: If a pa-



parameter is contained within a post-build changeable container that is not the target of any pre-build reference, the parameter is editable at post-build time (even if it has the configuration class **Pre Compile** or **Link**). This also applies for new post-build changeable container instances. Or, to put it the other way round: if a pre-build parameter is editable at post-build time, it is contained within a post-build changeable container and has not been used by any **Pre Compile** or **Link** generator.

6.14.4.2. Dependencies between parameters

Two MCCs within one module represent two different loadable configurations but they have a huge portion of duplicated parameters which actually represent the same value. For this reason, there are some restrictions between the parameters within different MCCs:

- ▶ The value of pre-build parameters and references that exist in several MCCs must have the same value in all MCCs. Refer to AUTOSAR constraint [constr_5501] for details.
- ▶ A post-build changeable container within a list may only be removed at post-build time if neither the container itself, nor any of its subcontainers is the target of a pre-build reference. Refer to AUTOSAR constraint [constr_5504].
- ▶ EcuFunctionName parameters may not introduce new symbolic names at post-build time. Refer to AUTOSAR constraint [constr_5503] for details.

The reason for those constraints is, that the executable (which has already been flashed to the ECU) relies on the parameter values that were set at pre-build time. Thus it is not possible to change them any more since the associated code has already been generated.

6.15. Creating a customer support package

6.15.1. Overview

In this chapter you learn how to create a support package that contains relevant information about your projects and the EB tresos Studio installation using the **Create support package** dialog.

The support package can contain files of projects, licensed features, the system report, and contents from the **Error Log**. In addition you can use the **Create support package** dialog to add plug-ins for configured modules, so the support team can reproduce an error easier.



6.15.2. Using the Create support package dialog

6.15.2.1. Opening the Create support package dialog

To open the **Create support package** dialog:

- ▶ Select the **Help** menu.

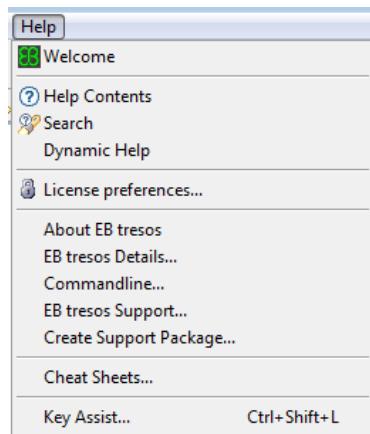


Figure 6.181. Create Support Package menu

- ▶ Select **Create Support Package** from the help menu.

The **Create support package** dialog opens.

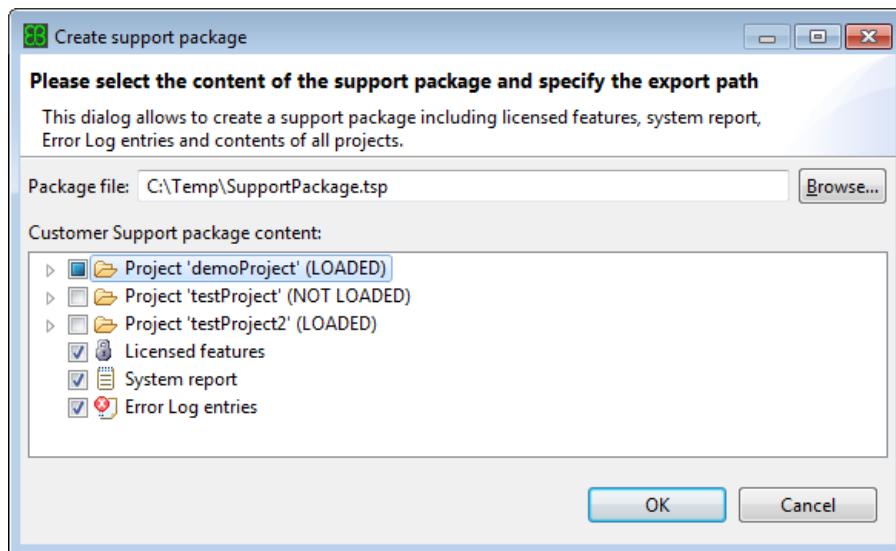


Figure 6.182. Create support package dialog

The **Create support package** dialog shows:



- ▶ One toplevel item for each project that is available in the workspace and is not closed.

This includes the information if the project is loaded or not loaded. Some information is only available if the project is loaded, e.g. **Problems View** entries, plug-ins for configured modules or linked resources.

This item contains everything that is exactly related to this project.

- ▶ One item to be able to include the list of licensed features.
- ▶ One item to add a system report of the installation (lists the product features, all installed plug-ins, etc.).
- ▶ One item to include the **Error Log** entries up to the maximum visible entries set in the EB tresos Studio preferences.

NOTE



Only saved changes are copied

If there are any unsaved changes in your project, these are not copied into the support package! Therefore it is recommended you save the project first before you create a support package.

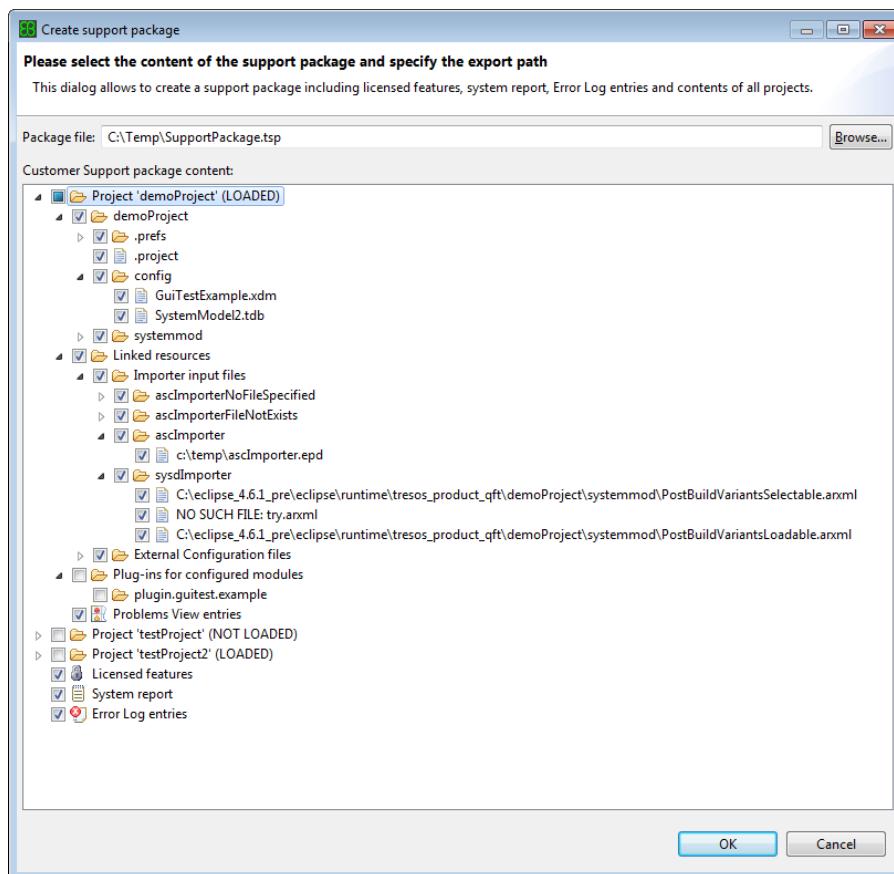


Figure 6.183. Create support package dialog (demoProject expanded)



Each project that is currently selected inside the **Project Explorer** is pre-selected when you open the **Create support package** dialog. This means that every subitem of the project toplevel item is by default selected except the item **Plug-ins for configured modules**.

This is due to the fact that adding all plug-ins for all configured modules consumes a lot of disc space. So if your problem is related to a certain plug-in then select it manually.

The **Linked resources** item contains all project-related information that is not directly stored in the project, e.g.:

▶ Importer input files

Here all importer input files are listed clustered by importer name. If a file does not exist, then an entry **NO SUCH FILE: <filename>** is shown.

▶ External configuration files

Here all module configuration files are listed which are not located inside the project.

When you add a module configuration to the project you may specify the path to the configuration file to be located outside of the project, e.g. by using variables or an absolute path.

▶ External generation files

Here all generated files are listed which are not located inside the project.

You can either change the generation path for the complete project or for each module configuration. The generation path can point to a location outside of the project, e.g. by using variables or an absolute path.

6.15.2.2. Creating the support package

When the dialog is open, proceed as follows:

- ▶ Use the check boxes to select which information shall be added to the support package. It is recommended that you include at least the following:
 - ▶ Licensed features
 - ▶ System report
 - ▶ **Error Log** entries
- ▶ Use the text box or the **Browse...** button to select the location and name for the support package. The **Save As** dialog opens if you click the **Browse...** button.
- ▶ In the **Save As** dialog, select the location and name for the support package and press the **Save** button.
- ▶ Click the **OK** button to create the support package with your selection items.

The support package is created in your selected location as a zip file with the extension ".tsp" (Tresos Support Package). You can send this support package file to the EB tresos customer support team.



7. Tool extensions

7.1. Overview

This chapter provides information on how to use tool extensions of EB tresos EB tresos Studio via the graphical user interface.

These tool extensions are separate products and may not be included in your installation.

If you have never worked with EB tresos Studio, it is recommended to start reading [Chapter 3, "Background information"](#) before you proceed to the next sections.

If you are familiar with EB tresos Studio and the concepts or you just need to look up something, proceed to the task-oriented instructions.

- ▶ [Section 7.2, "Using the Connection Editor"](#) provides instructions on how to use the Connection Editor to create connections between ports and add component prototypes to composition component types.
- ▶ [Section 7.3, "Using the System Model Viewer"](#) provides instructions on how to use the System Model Viewer for viewing the whole system model .
- ▶ [Section 7.4, "Using the Signal Mapping Editor"](#) provides instructions on how to use the Signal Mapping Editor for adding data mappings to data elements.
- ▶ [Section 7.5, "Using the Auto-connect SWC unattended wizard"](#) provides instructions on how to use the Auto-connect SWC unattended wizard for auto-connecting software components.
- ▶ [Section 7.6, "Using the Auto-map system signals unattended wizard"](#) provides instructions on how to use the Auto-map system signals unattended wizard for auto-mapping system signals.
- ▶ [Section 7.7, "Using the Edit PostBuildVariants wizards"](#) provides instructions on how to use the Edit PostBuildVariants wizard to add, edit or remove predefined variants.
- ▶ [Section 7.8, "Using the Variant Handling wizard"](#) provides instructions on how to use the Variant Handling wizard to transfer multiple variants into the ECU configuration.
- ▶ [Section 7.9, "Using the Export/Import Split Configuration unattended wizards"](#) provides instructions on how to use the **Export/Import Split Configuration** unattended wizards to export and import split ECU configuration files.
- ▶ [Section 7.10, "Using the Transform XML files using XSLT wizard"](#) provides instructions on how to use the Transform XML files using XSLT wizard to transform XML using XSLT.
- ▶ [Section 7.11, "Using the Service Needs Calculator"](#) provides instructions on how to use Service Needs Calculator wizard to automatically update the configuration.



7.2. Using the Connection Editor

7.2.1. Overview

This section describes the Connection Editor.

Note that this section is intended for readers who have good knowledge of AUTOSAR.

- ▶ [Section 7.2.2, “Background information”](#) provides a functional overview of the Connection Editor.
- ▶ [Section 7.2.3, “Prerequisites”](#) provides an overview of the prerequisites for working with the Connection Editor.
- ▶ [Section 7.2.4, “Using the EB tresos Connection Editor”](#) provides instructions on how to use the Connection Editor. This editor allows you to add component prototypes to composition component types and to add connections between ports in your AUTOSAR project.

7.2.2. Background information

The Connection Editor is used to connect ports through connectors and to add component prototypes to compositions.

The Connection Editor has the following special properties:

- ▶ The Connection Editor works on the hierarchical system model, see [Section 7.2.2.1, “The hierarchical model”](#).
- ▶ You can delegate ports to AUTOSAR services, see [Section 7.2.2.2, “Delegatable service ports”](#).
- ▶ All prototypes of a composition are affected, see [Section 7.2.2.3, “Changes affect all prototypes of a composition”](#).
- ▶ Connectors are displayed as children of the ports they connect, see [Section 7.2.2.4, “Connectors are displayed as children of the ports they connect”](#).

7.2.2.1. The hierarchical model

The Connection Editor uses the hierarchical system model, which is displayed in the tree table to display the component structure that starts with the `TopLevelComposition`. In former versions of EB tresos Studio the following elements were only added to the ECU extract:



- ▶ Component prototypes of service component types
- ▶ Service port mappings to connect service ports

Because the ECU extract was potentially generated each time a system description importer was executed again due to a reimport, all service port mappings were lost. Now the Connection Editor executes these actions on the hierarchical model. However, these actions are also reflected in the ECU extract, when it is created after the Connection Editor is used. All changes made with the Connection Editor are also saved in an *.arxml file, which can then be reimported. When you import the file you should not validate against a strict XML schema because the *.arxml file contains only a partial model. For information about creating the ECU extract, see [Section 7.2.4.7, “Proceeding after editing connections”](#).

7.2.2.2. Delegatable service ports

The AUTOSAR Software Component Template [constr_1174] specifies that `PortInterface` elements used in the context of `CompositionSwComponentType` elements cannot refer to AUTOSAR services. However, you can connect ports with the attribute `isService=true` across composition boundaries. Thus, appropriate delegation ports are generated. If you generate an ECU extract, this constraint is not violated there, because delegation connectors are resolved.

7.2.2.3. Changes affect all prototypes of a composition

If you edit a composition in the Connection Editor, all prototypes of that composition type are affected simultaneously. For example if you add a component prototype to the composition, the added component prototype is immediately shown in all prototypes of the composition type. This behavior follows the AUTOSAR specification.

7.2.2.4. Connectors are displayed as children of the ports they connect

The Connection Editor displays connectors as children of the ports they connect, although they are aggregated by the composition type in the system model. This means that there are always two entries that represent the same connector of the system model. For example if you add a connector, a new connector entry appears at both connected ports and also possibly at other instances of these ports if the affected composition type is instantiated more than once.



▼ Edit Compositions and Connections

type filter text			
Entity	Target	Interface	
RootSwCompositionPrototype			
iii		SenderReceiverInterface	
II			
IV			
SwcE_Port_e_ID11			
SwcE_Port_e_ID11_SwcE_Port_e_ID11	E/SwcE_Port_e_ID11	SenderReceiverInterface	
iv		SenderReceiverInterface	
D			
E			
SwcE_Port_e_ID11		SenderReceiverInterface	
SwcE_Port_e_ID11_SwcE_Port_e_ID11	IV/SwcE_Port_e_ID11		

Figure 7.1. Each connector appears twice in the tree table

The Connection Editor also displays connectors across ECU boundaries. These connectors have a different icon which resembles a *cut connection*. The path of the port from the other ECU is displayed in the corresponding target column. The interface names are displayed in the interface column.

▼ Edit Compositions and Connections			
type filter text			
Entity	Target	Interface	
swArc			
PPortPrototype		if1	
RPortPrototype		if2	
RDelegation_acrossECU	PrototypeBB/OuterPortForB		
PrototypeAA			

Figure 7.2. These connectors appear only once in the tree table

7.2.3. Prerequisites

To work with the Connection Editor, make sure that the following conditions are fulfilled:

- ▶ *The project contains a system model (AUTOSAR 3.2 or newer).* Import a system model with the relevant elements into your project before you open the Connection Editor.



- ▶ A system and an ECU instance is selected. In the EB tresos Studio project properties, select the system and the ECU instance for which you want to edit the compositions and connections.

You can create a new Ecu instance in the system selection settings if of the project, if there is no Ecu instance in the imported system description data. This Ecu instance is exported into the ConnectionEditor.arxml file.

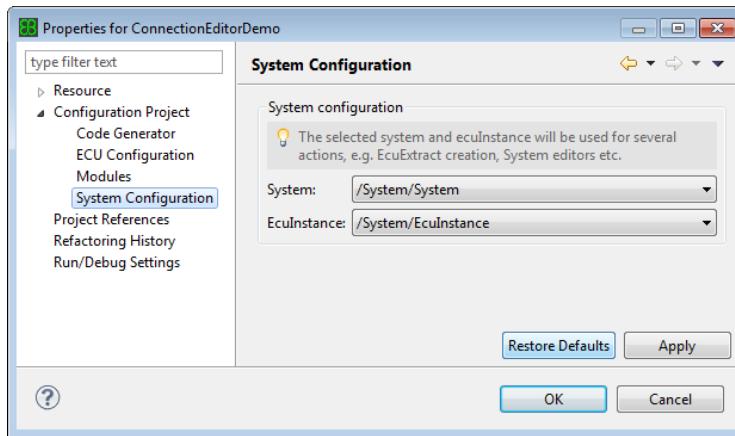


Figure 7.3. Setting the system and ECU instance

Otherwise, you will be asked to select a system and an ECU instance when you open the Connection Editor for the first time.

- ▶ The selected system contains a valid root software composition prototype. The Connection Editor displays the software component hierarchy beginning at the root software composition prototype of the selected system.
- ▶ The project system model contains all software component types that shall be instantiated. The Connection Editor provides the option to add prototypes to software composition types. You can only add prototypes of existing component types. This also applies to service component types. The *Update Service Component and BSWM Descriptions* adds service component types for modules where usage of the RTE is enabled. To add service component prototypes in your project, execute the *Update Service Component and BSWM Descriptions* before you use the Connection Editor.

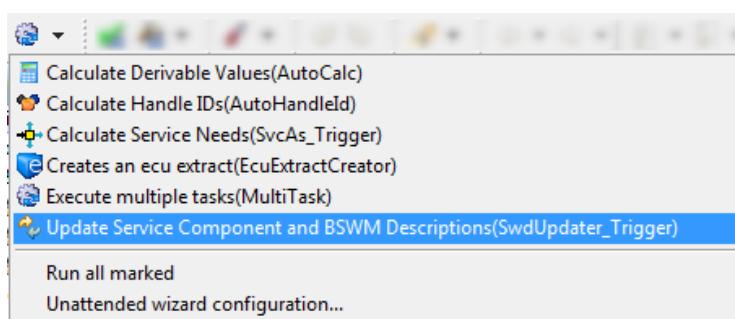


Figure 7.4. Triggering the *Update Service Component and BSWM Descriptions* unattended wizard



7.2.4. Using the EB tresos Connection Editor

- ▶ [Section 7.2.4.1, “Getting started with the Connection Editor”](#) provides instructions on how to open the Connection Editor, and describes the functionality of its three main sections.
- ▶ [Section 7.2.4.2, “Connecting ports manually”](#) provides instructions on how to connect two ports.
- ▶ [Section 7.2.4.3, “Creating a delegation connector”](#) provides instructions on how to create a delegation connector for a port.
- ▶ [Section 7.2.4.4, “Adding a component prototype”](#) provides instructions on how to add a component prototype to a composition component type.
- ▶ [Section 7.2.4.5, “Removing elements”](#) provides instructions on how to remove elements such as connectors or software component prototypes from the model.
- ▶ [Section 7.2.4.6, “Connecting ports automatically”](#) provides instructions on how to automatically connect ports by matching their short names to regular expressions.

7.2.4.1. Getting started with the Connection Editor

7.2.4.1.1. Opening the Connection Editor

To open the Connection Editor, proceed in the following way:

- ▶ Select the **System** section in the **Sidebar**.
- ▶ Double-click the **Edit Compositions and Connections** entry.

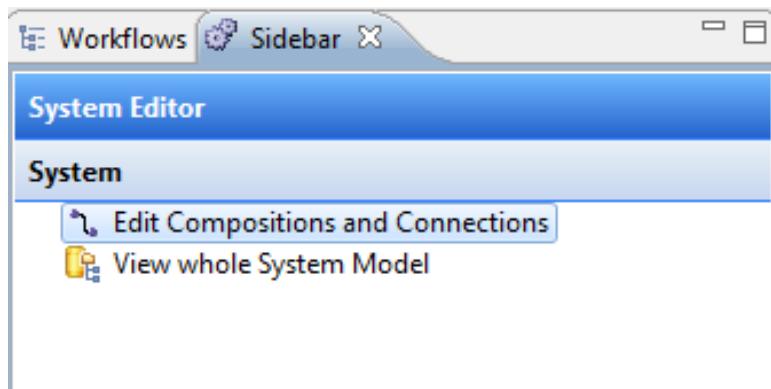


Figure 7.5. Opening the Connection Editor from the Sidebar



7.2.4.1.2. The sections of the Connection Editor

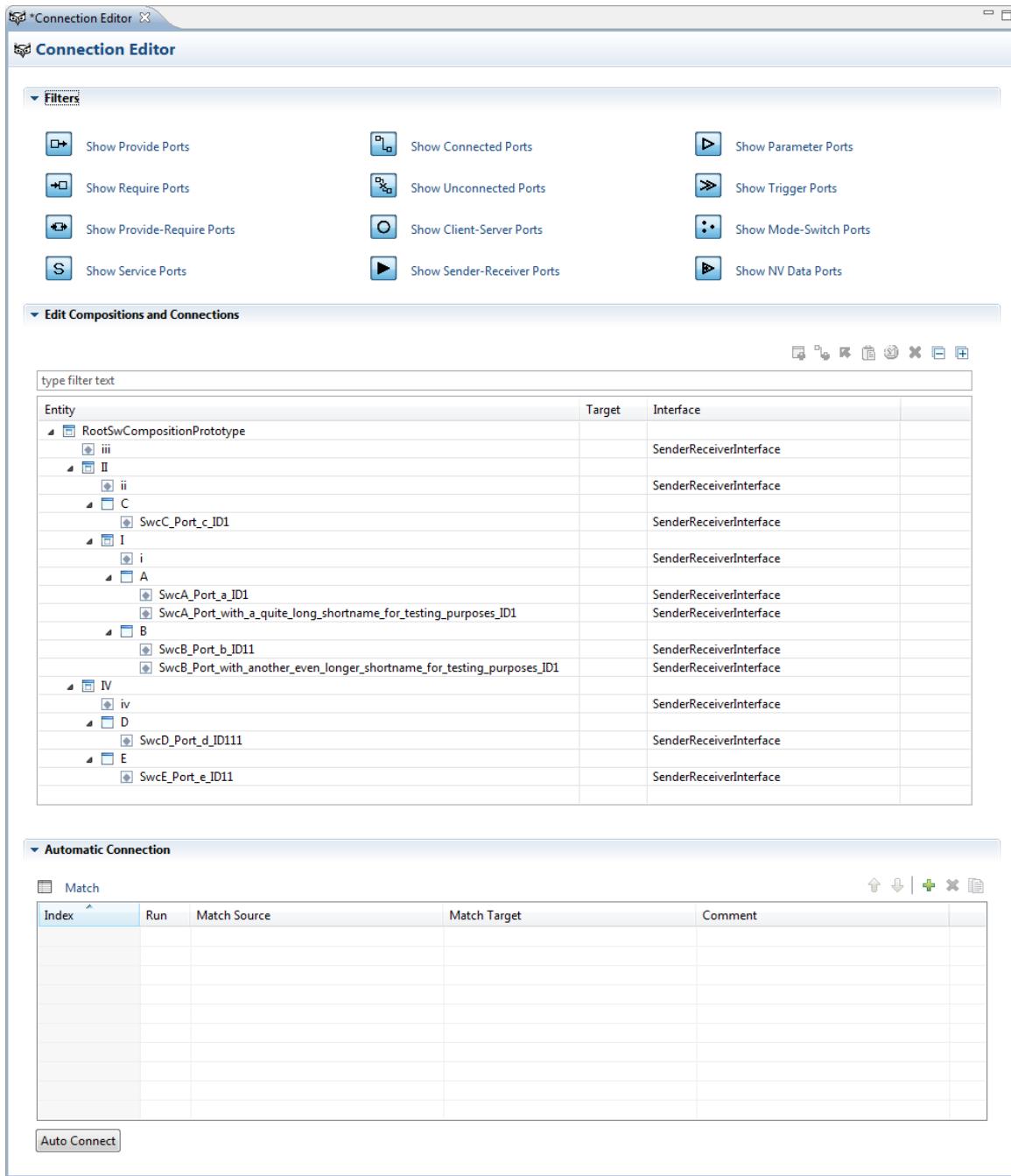


Figure 7.6. The Connection Editor

Filters

In the **Filters** section you control which ports to display in the tree table of the **Edit Compositions and Connections** section.

When you open the Connection Editor for the first time, all elements are displayed by default. This means, all filters are enabled.



If you apply any filter and close the Connection Editor, the filter remains active. If you re-open the Connection Editor, the previous session is restored.

Some filters make defined items invisible in the tree. For example, if the system model contains a NV Data Provider Port, you can hide this port, either by unselecting the **Show Provider Ports** filter, or by unselecting the **Show NV Data Ports** filter.

The filters in this section affect only the displayed elements in the tree table. The filters do not change the system model itself.

Edit Compositions and Connections

In the **Edit Compositions and Connections** section, you view the relevant elements of the project's system model in a tree table.

The Connection Editor displays all components of the currently selected system that are mapped directly or indirectly to the currently selected ECU and their ports. Compositions that are not mapped to the current ECU but contain components that are mapped to the current ECU are considered implicitly mapped to the current ECU. Components mapped to another ECU instance are not shown.

You modify the system model elements through the actions available in the context menu or the tool bar.

You can limit the displayed elements if you enter a search string in the text box **type filter text**.

You can navigate to the connected port if you select a connector in the tree table and click the **Jump to connected port** button in the tool bar.

Automatic Connection

In the **Automatic Connection** section, you connect ports automatically.

You specify match rules that consist of regular expression pairs.

You configure the match rules through the buttons in the toolbar, e.g. you change the order of the match rules.

You control which match rules to apply during automatic connection.

7.2.4.2. Connecting ports manually

The Connection Editor supports two modes of port connection: connecting ports manually, and connecting ports automatically. In this chapter, you learn how to connect two compatible ports manually. For information on automatic port connection, see [Section 7.2.4.6, “Connecting ports automatically”](#).

To connect two compatible ports manually, proceed in the following way:

- ▶ Select a port you wish to connect in the tree table that is displayed in the **Edit Compositions and Connections** section.
- ▶ Right-click on the port to open the context menu.



- Select the **Add Connector** menu item.

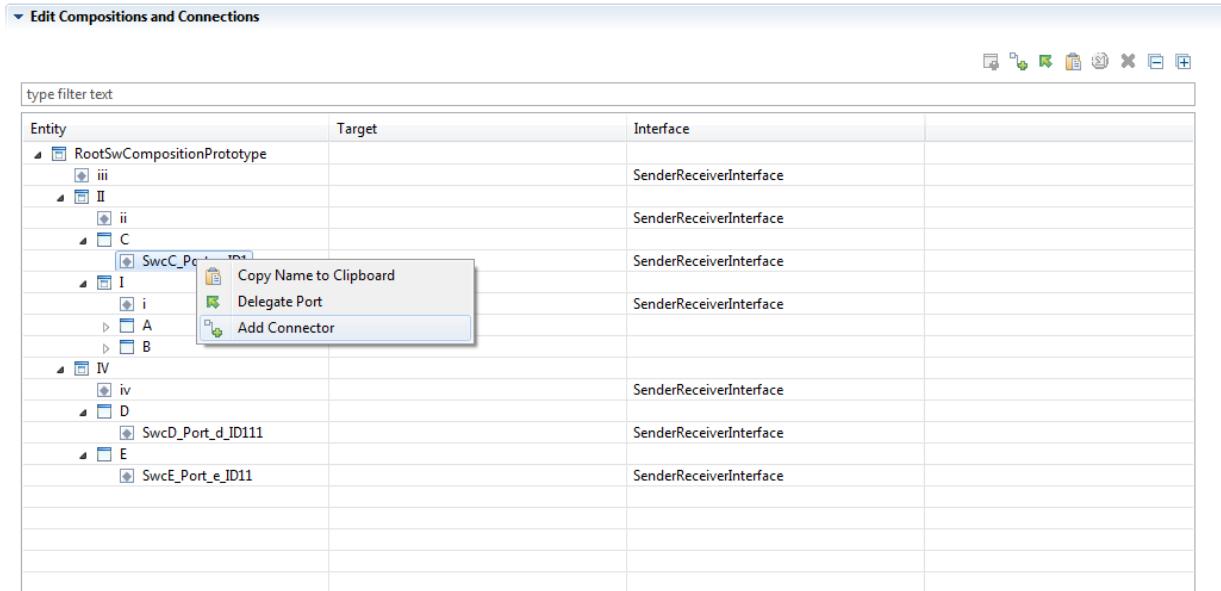


Figure 7.7. The **Add Connector** menu item

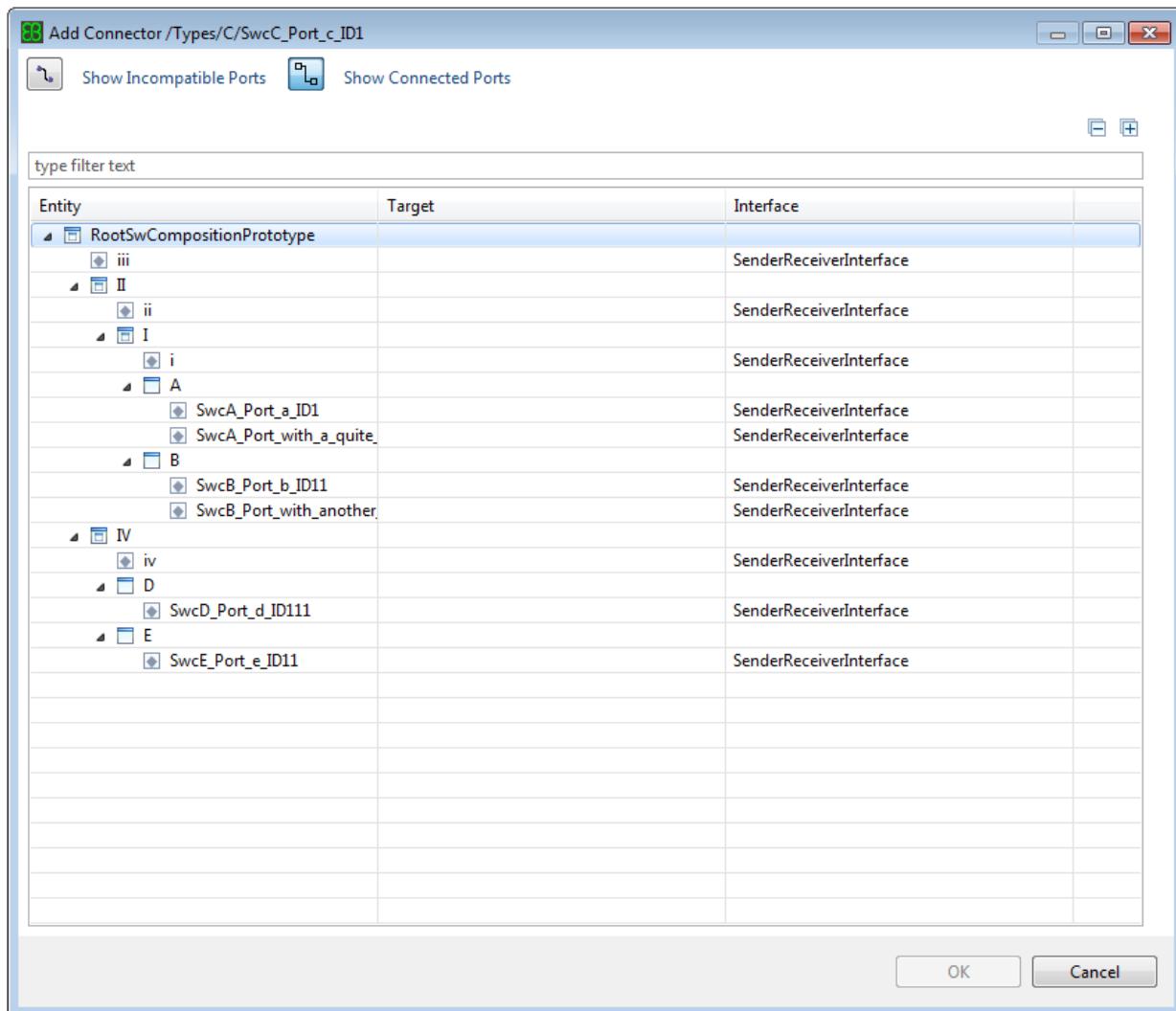
NOTE



Add Connector is available if applicable

If the **Add Connector** menu item does not appear, the selected port cannot be connected, for example, because it is a client/server require port that is already connected.

- The **Add Connector** dialog opens.

Figure 7.8. The **Add Connector** dialog

► In the **Add Connector** dialog, proceed in the following way:

► Optionally, filter the displayed elements in the tree table.

To filter the displayed elements in the tree table, use one of the following options:

- The **Show Incompatible Ports** button
- The **Show Connected Ports** button
- The **type filter text** field
- Select a compatible port in the tree table.
- Click the **OK** button.

**NOTE****OK is enabled if a compatible port is selected**

If the **OK** button is not enabled, you selected a port that is not compatible. Select a different port.

The selected ports are connected, even when they are not accessible in the same composition. In this case, delegation connectors are created. The delegation connectors delegate the selected ports to the first common composition. In the first common composition, the selected ports are connected by an assembly connector.

7.2.4.2.1. Connecting incompatible ports

In Connection Editor it is allowed to connect incompatibles ports. As a result, the user can connect ports that refer incompatible interfaces. The functionality should be used only on the earliest stages of the project. In case the incompatibility connections persists, the generated code of Runtime Environment(RTE) will not be compilable.

In order to connect incompatible ports, proceed in the following way:

- ▶ Select the button **Allow incompatible ports connection**



Figure 7.9. The **Allow incompatible ports connection** button from the section **Edit Compositions and Connections**

- ▶ Select the incompatible ports that will be connected by assembly connection
- ▶ Check in the table view the new connection. The new connection will be marked with an warning sign

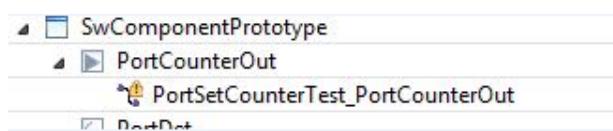


Figure 7.10. Incompatibility connection in the table view

- ▶ Close the editor

Next time when Connection Editor will be open the user will be notified with a pop-up windows that the configuration contains incompatible connections. User has to chose to load or not the incompatible connections. In case the user select yes, the existing incompatible connections will persist in the configuration, if not the connections will be deleted

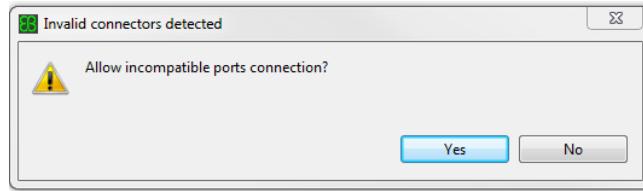
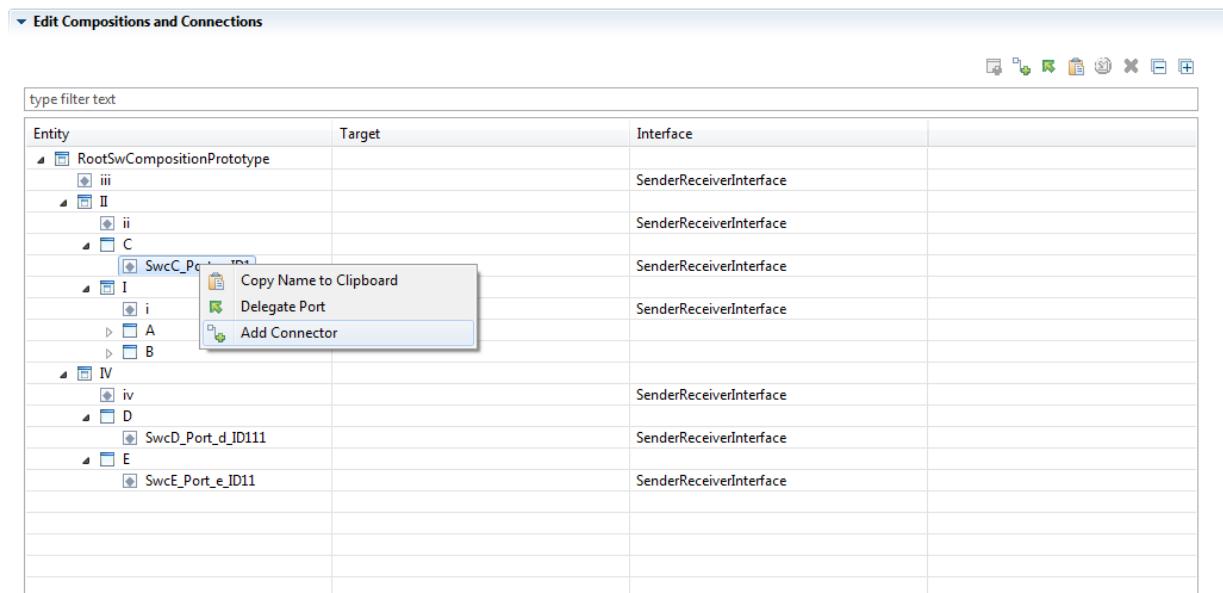


Figure 7.11. Pop up Invalid connectors detected

7.2.4.3. Creating a delegation connector

To create a delegation connector, proceed in the following way:

- ▶ Select a port for which you need a delegation connector in the tree table .
- ▶ Right-click on the port to open the context menu.
- ▶ Select the **Delegate Port** menu item.

Figure 7.12. The **Delegate Port** menu item

To create multiple delegations connectors, proceed in the following way:

- ▶ Select a port for which you need a delegation connector in the tree table.
- ▶ Press and hold **Ctrl** or **Shift**.
- ▶ Select the other ports for which you need delegations connectors in the tree table.
- ▶ Release **Ctrl** or **Shift**.
- ▶ Right-click the port to open the context menu.



- ▶ Select the **Delegate Port** menu item.

NOTE**Multiple selection delegation warning**

If an element cannot be delegated, a warning is logged.



A new port with the same name is created in the surrounding composition type with a delegation connector to the selected port. As ports are part of the type and not of the prototype, the new port appears in the tree table for every prototype of the composition type.

7.2.4.4. Adding a component prototype

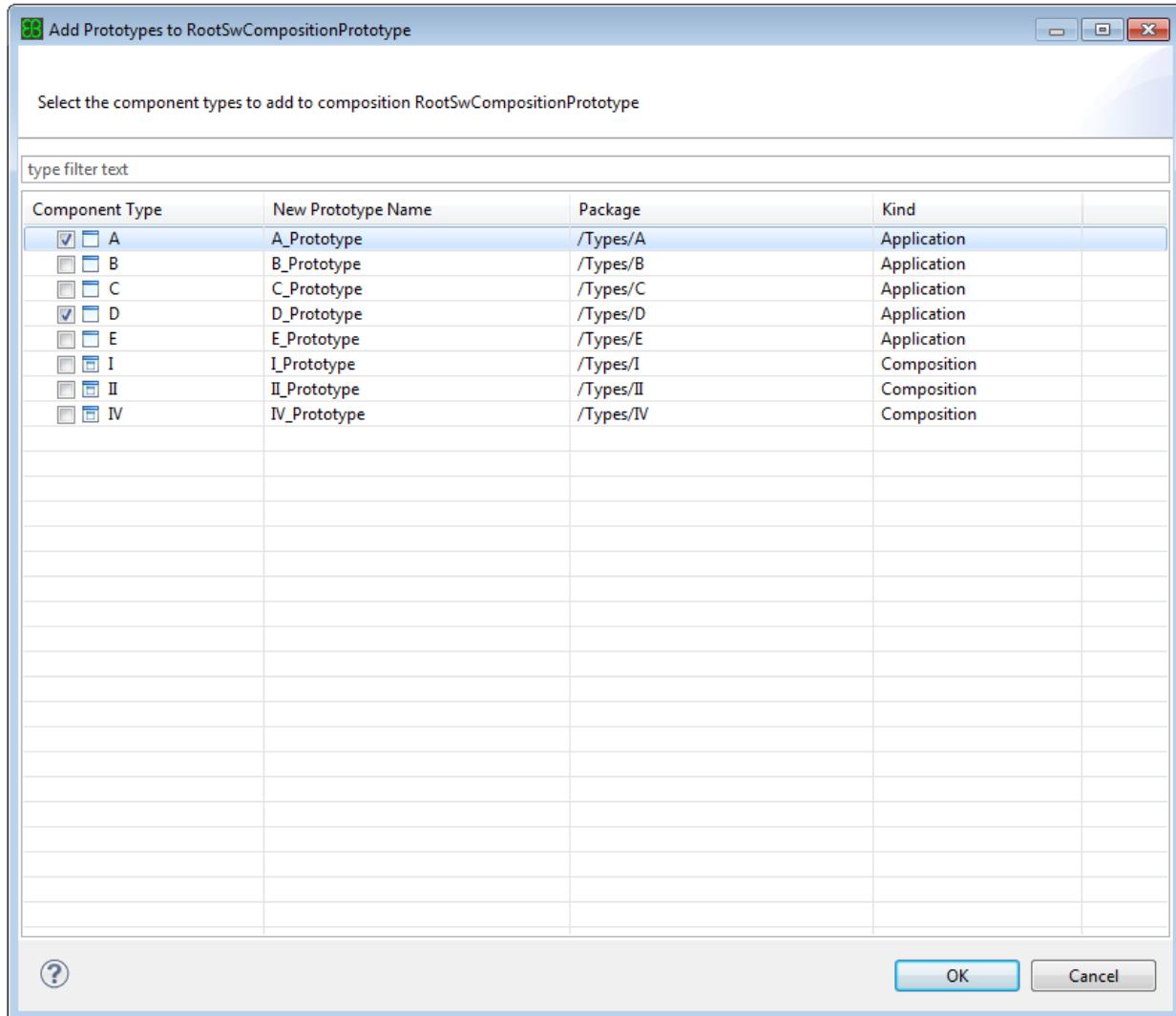
To add a component prototype to a composition component type, proceed in the following way:

- ▶ Select a prototype of the composition type (for example the root composition shown as the tree root) where you want to add a component prototype.
- ▶ Right-click on the composition to open the context menu.
- ▶ Select the **Add Prototypes** menu item.

Edit Compositions and Connections			
<input type="text" value="type filter text"/> New Open Save Save All Close Minimize Maximize			
Entity	Target	Interface	
RootSwCompositionProt			
iii	Expand the Selected Element		
II	Collapse the Selected Element		
ii	Copy Name to Clipboard		
C	Add Prototypes		
SwcC_Port_c_1uz1			
I			
i			
A			
SwcA_Port_a_ID1		SenderReceiverInterface	
SwcA_Port_with_a Quite_l		SenderReceiverInterface	
B			
SwcB_Port_b_ID11		SenderReceiverInterface	
SwcB_Port_with_another_c		SenderReceiverInterface	
IV			
iv			
D			
SwcD_Port_d_ID111		SenderReceiverInterface	
E			
SwcE_Port_e_ID1		SenderReceiverInterface	

Figure 7.13. The **Add Prototypes** menu item

- ▶ The **Add Prototypes** dialog opens.

Figure 7.14. The **Add Prototypes** dialog**NOTE****Available component types depend on the selected composition type**

The component types that are displayed in the table depend on the selected composition type. Service component types can only be added to the top level composition. Composition types that would lead to a cycle in the composition hierarchy also cannot be added.

- ▶ In the **Add Prototypes** dialog, proceed as follows:

- ▶ Optionally, filter the displayed component types in the table by using the **type filter text** field.
- ▶ Check all component types from which you want to add a prototype to the selected composition.
- ▶ Optionally, double-click into the **New Prototype Name** table column of a selected component type to specify the new prototype name.



NOTE**Only valid prototype names are considered**

Make sure you enter a valid name for a new prototype. If you enter an invalid name, the suggested default name is used instead of the name you specified. A name is invalid if it is, for example, a no valid AUTOSAR short name, or a prototype of the same name that already exists in the selected composition.

- ▶ Click the **OK** button.

A prototype of every checked component type is added to the composition type.

NOTE**Changes affect all prototypes of the composition type**

If you add a component prototype to the composition type, the added component prototype is reflected in every prototype of this composition type.

NOTE**Service component prototypes can only be added to the TLC**

Although you can delegate service port prototypes, you can add service component types only to the top level composition (TLC) of your system. In conformance with AUTOSAR you can only instantiate service component types once.

NOTE**SWC to ECU mapping**

If you add a component prototype in the Connection Editor, it is mapped to the ECU you selected for the current project.

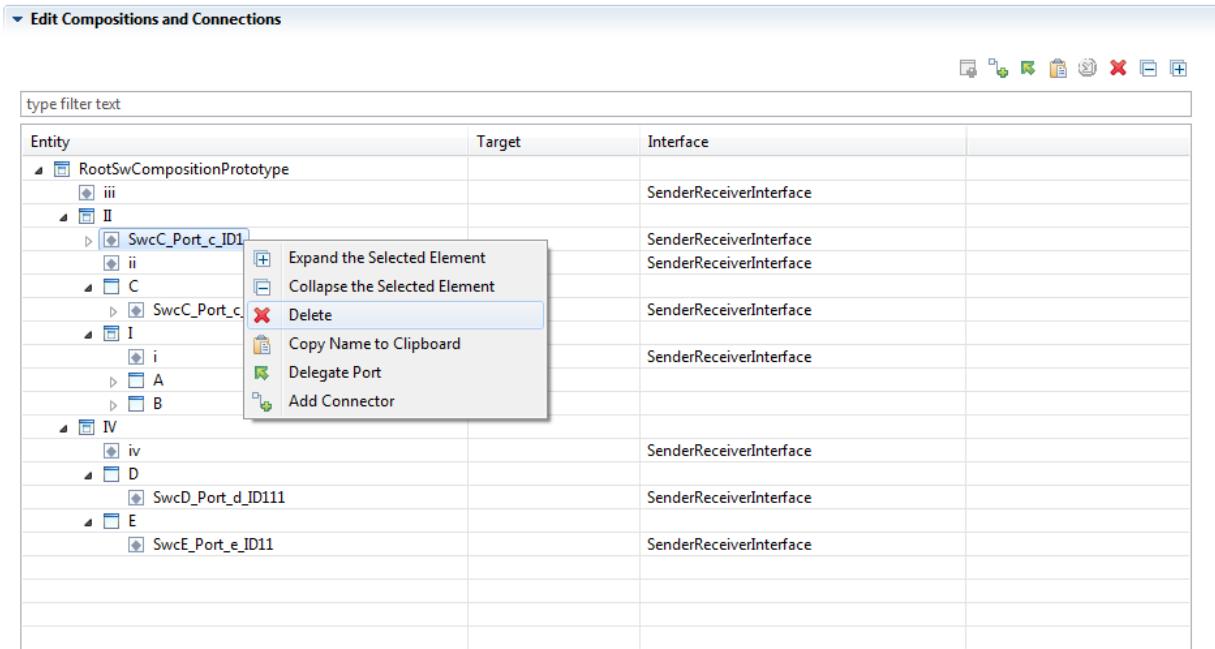
7.2.4.5. Removing elements

NOTE**Removable elements are restricted**

You can only remove elements that have been added through the Connection Editor, either in the current session or in previous sessions.

To remove a component prototype, a connector, or a delegation port, proceed in the following way:

- ▶ Select the element you want to remove.
- ▶ Right-click on the element to open the context menu.
- ▶ Select the **Delete** menu item.

Figure 7.15. The **Delete** menu item

The selected element is removed.

To remove multiple component prototypes, connectors, or delegation ports, proceed in the following way:

- ▶ Select one of the elements you want to remove.
- ▶ Press and hold **Ctrl** or **Shift**.
- ▶ Select the other elements you want to remove.
- ▶ Release **Ctrl** or **Shift**.
- ▶ Right-click the element to open the context menu.
- ▶ Select the **Delete** menu item.

The selected elements are removed.

**NOTE****Changes affect all prototypes of the composition type**

The element is removed from all prototypes of the affected composition type.

NOTE**Multiple selection delete warning**

If an element cannot be deleted, a warning is logged.

7.2.4.6. Connecting ports automatically

The Connection Editor also supports an automatic connection feature which matches and connects ports automatically. The ports' match is based on their short name. The connection is based on match rules that consist of regular expression pairs.

Automatic Connection				
Match				
Index	Run	Match Source	Match Target	Comment

Auto Connect

Figure 7.16. The Automatic Connection section in the Connection Editor

To match ports and connect ports automatically, proceed in the following way:

- ▶ Add a match rule. To add a match rule click on the **Add new element with default values** button
- ▶ Specify the match rule:
 - ▶ Specify the match source regular expression in the **Match Source** column.
 - ▶ Specify the match target regular expression in the **Match Target** column.

**TIP****Use the short name of an element to make a regular expression**

Copy the short name of an element in the **Edit Compositions and Connections** tree table and make a match source regular expression from it.

To make a regular expression from an element's short name:

- ▶ Select an element in the **Edit Compositions and Connections** tree table.
- ▶ Click on the **Copy Name to Clipboard** button
- ▶ Insert the content of the clipboard into the **Match Source** table.

TIP**Use capture groups in match rules**

Use capture groups in the match source expression, for example `([a-zA-Z0-9]+_ -)`. Reference the capture groups in the match target expression. To reference the capture groups use `\#ofGroup`, for example `\1` to reference the first capture group of the source expression. Non-capture groups denoted by `(?:)` are not counted in the group reference number. Group references in the match target expression are substituted by the current value of the group before the regular expression is used to find a match port.

-
- ▶ If you need further match rules, repeat the above steps.
 - ▶ Optionally, change the order of match rules.

To change the order of match rules, move match rules up or down:

- ▶ To move up a match rule, select the match rule you want to apply, and click on the **Move up selected elements** button
- ▶ To move down a match rule, select the match rule you want to apply, and click on the **Move down selected elements** button
- ▶ Optionally, duplicate match rules.

To duplicate a match rule, select the match rule you want to apply, and click on the **Duplicate selected elements** button

The selected match rule is copied and added to the end of the list. The state of the check box in the **Run** column of the match rule is also copied.

- ▶ Optionally, remove match rules.

To remove a match rule, select the match rule you want to apply, and click on the **Remove selected elements** button

The selected match rule is deleted. The index for the remaining match rules is adapted to be zero-based consecutive.

- ▶ Check the check box in the **Run** column of each match rule you want to apply. Only checked match rules are considered during automatic connection.



NOTE**Match rules are performed in index order, regardless of table appearance**

You are able to change the table appearance through sorting the columns by criteria other than the index. However, automatic connection performs the match rules in the zero-based consecutive order of the index.

- ▶ Click the **Auto Connect** button.

The source regular expression is evaluated for all ports of the system. If the short name of a port matches this expression, a match target is searched. For every match of the respective source expression, the target expression is evaluated for all ports of the system. If the short name of a target port matches this expression and it is compatible to the source port, the two ports are connected. The automatic connection creates delegation connectors and delegation ports as described in [Section 7.2.4.2, “Connecting ports manually”](#).

WARNING**Match expressions must match unique port names**

To avoid unwanted behavior, take care when you design the regular expressions for automatic connection.

Automatic connection may connect ports more than once if such connections are valid. This may, for example, happen if several expressions match the same port.

The match does only consider the short name of a port, not the path. This means that if multiple instances of the same port prototype exist, or if different ports have the same short name, you may get unexpected results.

When ports are delegated using the Connection Editor, the delegation ports receive the same short name as the delegated port. Delegation ports that are created by automatic connection also receive the same short name as the delegated port. This may result in unexpected connections when you use multiple match expressions.

The following example illustrates the instructions above:



The screenshot shows the Connection Editor window in EB tresos® Studio. The main area displays a tree view of entities and their ports, with a table below showing connections. The table has columns for Entity, Target, and Interface.

Entity	Target	Interface
RootSwCompositionPrototype		
iii		SenderReceiverInterface
II		SenderReceiverInterface
ii		SenderReceiverInterface
C		SenderReceiverInterface
SwcC_Port_c_ID1		SenderReceiverInterface
I		SenderReceiverInterface
i		SenderReceiverInterface
A		SenderReceiverInterface
SwcA_Port_a_ID1		SenderReceiverInterface
SwcA_Port_with_a Quite_long_shortname_for_testing_purposes_ID1		SenderReceiverInterface
B		SenderReceiverInterface
SwcB_Port_b_ID11		SenderReceiverInterface
SwcB_Port_with_another_even_longer_shortname_for_testing_purposes_ID1		SenderReceiverInterface
IV		SenderReceiverInterface
iv		SenderReceiverInterface
D		SenderReceiverInterface
SwcD_Port_d_ID111		SenderReceiverInterface
E		SenderReceiverInterface
SwcE_Port_e_ID11		SenderReceiverInterface

The 'Automatic Connection' section contains a table for matching source and target ports:

Index	Run	Match Source	Match Target	Comment
0	<input checked="" type="checkbox"/>	[?:[a-zA-Z0-9]+_]{3}ID(\d+)	[?:[a-zA-Z0-9]+_]{3}ID\1	SwcC_Port_c_ID1 -> SwcA_Port_a_ID1
1	<input type="checkbox"/>	[?:[a-zA-Z0-9]+_]{3}ID(\d+)	[?:[a-zA-Z0-9]+_]{3}ID\1	currently not used

Auto Connect button is located at the bottom left of the 'Automatic Connection' section.

Figure 7.17. Example of connecting ports automatically

The match rule with the match source expression `(?:[a-zA-Z0-9]+_){3}ID(\d+)` and the match target expression `(?:[a-zA-Z0-9]+_){3}ID\1` connects the following ports:

- ▶ `SwcC_Port_c_ID1` to `SwcA_Port_a_ID1`
- ▶ `SwcB_Port_b_ID11` to `SwcE_Port_e_ID11`

7.2.4.7. Proceeding after editing connections

After you have edited the connections of your system, proceed in the following way:



1. Save the changes to the system model.

If you close the Connection Editor without having saved your changes, a dialog opens and you are asked if you want to save the changes. To save the changes, click **Yes**.

Your changes are saved to the system model and written to the file `ConnectionEditor.arxml` that is located in the folder `systemmod` inside your project.

The `ConnectionEditor.arxml` only contains elements that are added by the **Edit Compositions and Connections** editor and the **System Signal Mapping**.

The `ConnectionEditor.arxml` does not contain the full System model as it exists in tresos Studio project. You must configure a System Description Exporter to have a complete System model in the `.arxml` file. This also includes the data in the `ConnectionEditor.arxml` file.

2. Update the ECU extract after the changes have been saved to the system model.

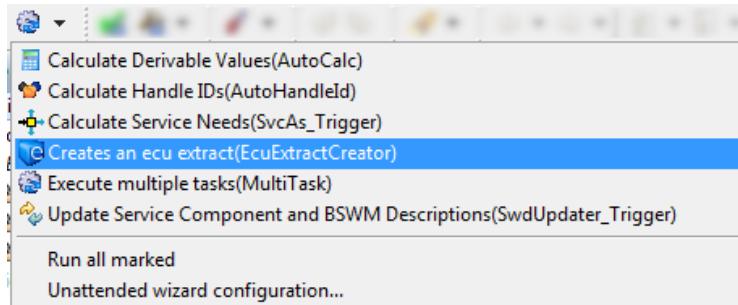


Figure 7.18. Triggering the *Creates an ecu extract* unattended wizard

NOTE



Update the ECU extract to make changes available to the Rte

If you do not update the ECU extract, your changes will not be available for the `Rte` module configuration.

7.3. Using the System Model Viewer

7.3.1. Overview

This section describes the System Model Viewer.

Note that this user's guide is intended for readers who have good knowledge of AUTOSAR.

- ▶ [Section 7.3.3, “Opening the System Model Viewer”](#) provides instructions on how to open the System Model Viewer.



- ▶ [Section 7.3.4, “Displaying entities in the Entity tree table”](#) provides instructions on how to display entities in the **Entity** tree table.
- ▶ [Section 7.3.5, “Filtering the displayed entities”](#) provides instructions on how to filter the amount of displayed entries in the **Packages** tree table.

7.3.2. Background information

The System Model Viewer allows you to view the whole system model. The System Model Viewer consists of two tree tables. The **Packages** tree table displays all existing packages and their contained entities of your system model. The **Entity** tree table is empty until you select an entity in the **Packages** tree table.

The screenshot shows the "System Model Viewer" window. On the left, there is a "Tree" view titled "Packages". It lists several packages under the "Tree" column and their types under the "Type" column. Some packages include XAUTOSAR, X0IKluxVikpz, X0Ikib3MpMXMVMKQZ, X2WTViV00LTZK, X5Yfijju, XCUM2b000TiVOQ0, XERKMwWP, XKYVJLz02VaHtz, XRM_EIMG, XRVhZGJOTUO, XTMOzXnRGV0U, XXW4WQI, XVEJiXP, XZGIZAGKV8Y, X_jl7PwQmoiskiX, XfxytHys, Xhn_wjXj, XjSi5Gr, XlipmVS7ZXU, X0aPZQ62PZQaOQa0Q820U8a, X2WTViV00LTZK, XF2JYFLk6, XFSLNXXNQ, XYXZQHGXW, XQEVIWI, XUP2IJUH0ZZL, XVXGS, X8YVMOYRTGY, Xmrywx, and XlipmVS7ZXU. On the right, there is a table titled "Entity" with columns "Property", "Value", and "Type". The table contains rows for properties of the selected package XlipmVS7ZXU, such as category (String), ecuExtractVersion (ByteO...), pncVectorLength (long), pncVectorOffset (long), shortName (Xlipm...), systemVersion (String), and uuid (String). There are also sections for mappings and other entity types like communicationClusters, couplingElements, ecuInstances, frames, gateways, globalTimeDomains, iSignalGroups, iSignals, nmConfigs, pdurIPduGroups, pdus, soAdRoutingGroups, and tpConfigs.

Figure 7.19. Viewing the whole system model with the **System Model Viewer**

7.3.2.1. Entity tree table icons



The elements displayed in the **Entity** tree table are displayed with icons to quickly differentiate their type:

- ▶ Attribute - an attribute is a simple value of an entity.
- ▶ Custom attribute - a custom attribute can be attached to any entity by various EB products.
- ▶ Attribute group - an attribute group is an array of attribute values. E.g. 8 byte values form one frame.
- ▶ Entity - an entity is a class in the AUTOSAR model. An entity can contain attributes, references and child entities.
- ▶ Entity group - a list of entities which are usually empty or have an infinite number.
- ▶ Reference - a link to an identifiable entity.
- ▶ Reference group - a list of references that point to identifiable entities of the same type.

7.3.3. Opening the System Model Viewer

To open the System Model Viewer, double-click **View whole System Model** in the **System** section of the **Sidebar** view. As a result, the System Model Viewer opens up in the editor area.

7.3.4. Displaying entities in the Entity tree table

To display entities in the **Entity** tree table, select an entity in the **Packages** tree table.

To select a child entity of the currently displayed entity as root element, click the **Switch to entity** button . You can also select a referenced entity this way.

To navigate to previously displayed entities, click the **Back** button . If you moved back, you can click the **Forward** button to move forward again

7.3.5. Filtering the displayed entities

You can filter the amount of displayed entries in the **Package** tree table either by type or by name.

7.3.5.1. Filtering entities by type

To filter the amount of displayed entries by type, use the **Filter by type** drop-down list box. You can either select an entity type from the list or enter an entity type. A content assistant shows valid entity types that are contained in the system model.

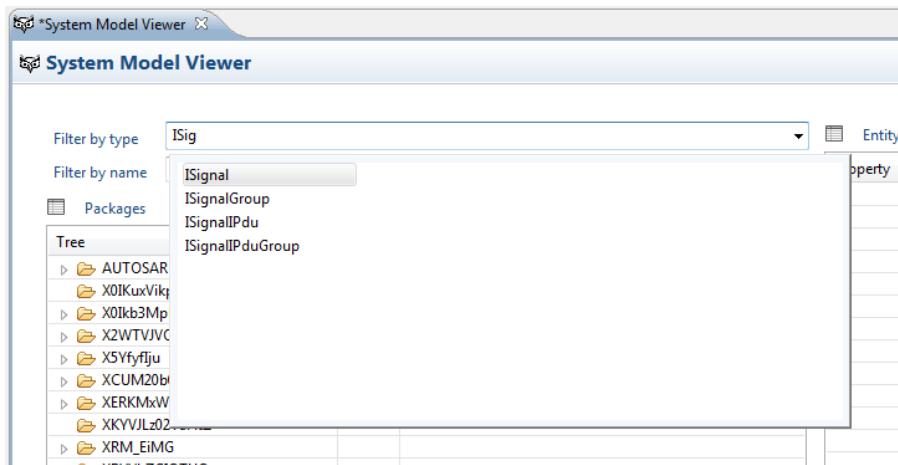


Figure 7.20. Filtering the displayed entity types in the **Package table** Editor

If you enter the desired entity type, the **Package table** only shows packages that contain entities of the specified type.

7.3.5.2. Filtering entities by name

To filter the amount of displayed entries by name, enter the SHORT-NAME of the desired entity into the **Filter by name** text box.

7.4. Using the Signal Mapping Editor

7.4.1. Overview

This section describes the EB tresos Signal Mapping Editor.

Note that this section is intended for readers who have good knowledge of AUTOSAR.

- ▶ [Section 7.4.2, “Background information”](#) provides a functional overview of the Signal Mapping Editor.
- ▶ [Section 7.4.3, “Prerequisites”](#) provides an overview of the prerequisites for working with the Signal Mapping Editor.
- ▶ [Section 7.4.4, “Using the EB tresos Signal Mapping Editor”](#) provides instructions on how to use the Signal Mapping Editor. This editor allows you to add data mapping to the data elements of the sender-receiver port and to the operations of the client-server port in your AUTOSAR project.



7.4.2. Background information

The Signal Mapping Editor is used to add data mapping to the data elements of the sender-receiver port and to the operations of the client-server port.

The Signal Mapping Editor has the following special properties:

- ▶ The Signal Mapping Editor works on the hierarchical system model, see [Section 7.4.2.1, “The hierarchical model”](#).
- ▶ The Signal Mapping Editor displays data mappings as children of the ports, see [Section 7.4.2.2, “Displayed data mappings”](#).
- ▶ The Signal Mapping Editor supports three types of data mapping, see [Section 7.4.2.3, “Supported types of data mapping”](#).

7.4.2.1. The hierarchical model

The Signal Mapping Editor uses the hierarchical system model, which is displayed in the tree table to display the component structure that starts with the `TopLevelComposition`. In former versions of EB tresos Studio, the data mappings were only added to the ECU extract. Because the ECU extract was potentially generated each time a system description importer was executed, for example after a re-import, all data mappings were lost. Now the Signal Mapping Editor executes these actions on the hierarchical model. The actions are also reflected in the ECU extract, when it is created after the Signal Mapping Editor is used. All changes made with the Signal Mapping Editor are also saved in an `*.arxml` file, which can then be re-imported. When you import the file, you should not validate against a strict XML schema because the `*.arxml` file contains only a partial model. For information about creating the ECU extract, see [Section 7.4.4.8, “Proceeding after data mappings were created”](#).

7.4.2.2. Displayed data mappings

The Signal Mapping Editor displays data mappings as children of the ports, although they are aggregated by the system mapping in the system model.

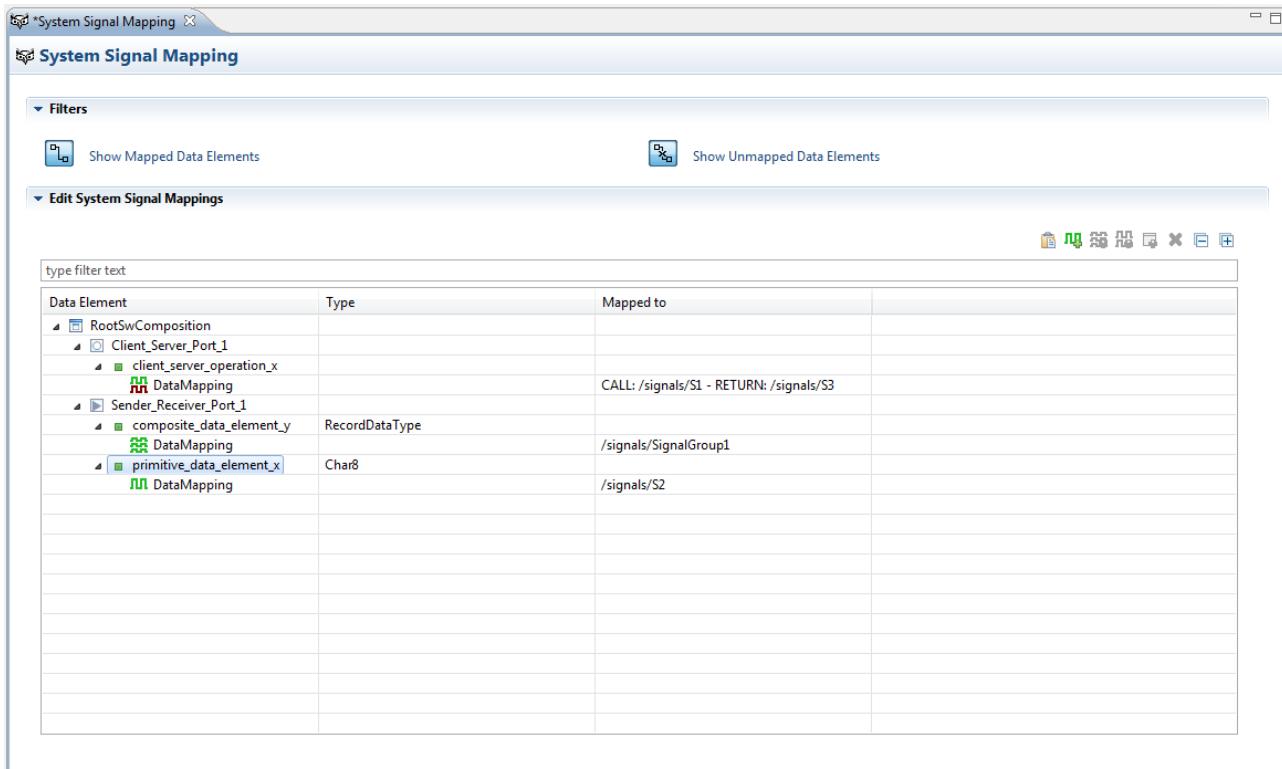


Figure 7.21. Data mappings are displayed as children of the ports

7.4.2.3. Supported types of data mapping

The Signal Mapping Editor supports the following types of data mapping:

- ▶ **SenderReceiverToSignalMapping**
Sender-receiver to signal mapping can be created for primitive data elements of a `PortPrototype` typed by a `SenderReceiverInterface`.
Sender-receiver to signal mapping can also be created for composite data elements of a `PortPrototype` typed by a `SenderReceiverInterface` if data transformation is used.
- ▶ **SenderReceiverToSignalGroupMapping**
Sender-receiver to signal group mapping can be created for data elements (typed by composite data types) of a `PortPrototype` typed by a `SenderReceiverInterface`.
- ▶ **ClientServerToSignalMapping**
Client-server to signal mapping can be created for `ClientServerOperation` instances of a `PortPrototype` typed by a `ClientServerInterface`.

**NOTE****Only sender-receiver and client-server ports are shown**

Ports with interfaces where no data mapping can be added by the Signal Mapping Editor are not shown.

7.4.3. Prerequisites

To work with the Signal Mapping Editor, make sure that the following conditions are fulfilled:

- ▶ *The project contains a system model (AUTOSAR 3.2 or newer).* Import a system model with the relevant elements into your project before you open the Signal Mapping Editor.
- ▶ *A system and an ECU instance is selected.* In the EB tresos Studio project properties, select the system and the ECU instance for which you want to define data mappings.

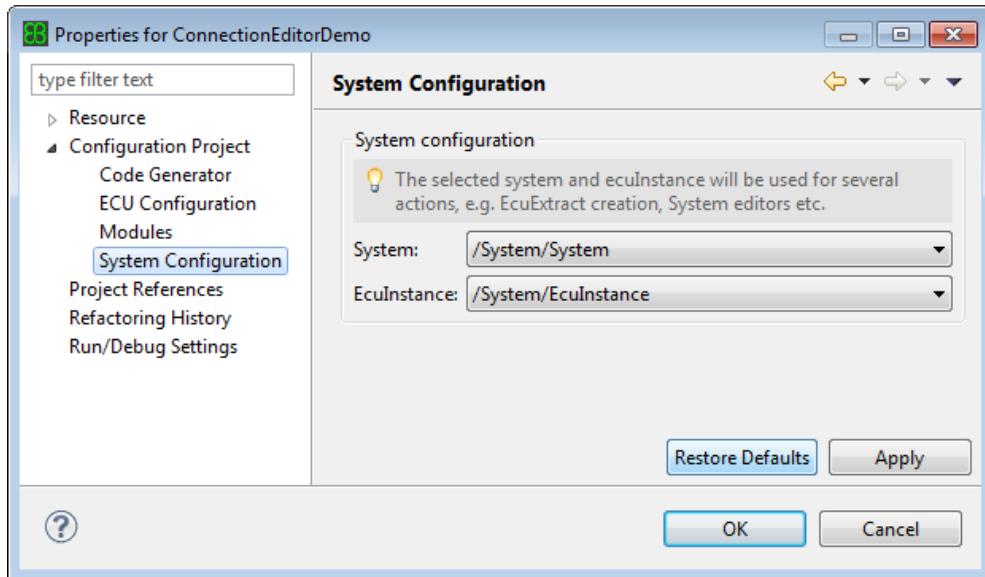


Figure 7.22. Setting the system and ECU instance

Otherwise, you will be asked to select a system and an ECU instance when you open the Signal Mapping Editor for the first time.

- ▶ *The selected system contains a valid root software composition prototype.* The Signal Mapping Editor displays the software components which begin at the root software composition prototype of the selected system. The ports are displayed as children of the software components. The data elements are displayed as children of the sender-receiver ports. The operations are displayed as the children of client-server ports. If data mappings are defined for a data element or operation, the defined data mappings are displayed as the children of the data element or operation.



7.4.4. Using the EB tresos Signal Mapping Editor

- ▶ [Section 7.4.4.1, “Getting started with the Signal Mapping Editor”](#) provides instructions on how to open the Signal Mapping Editor, and describes the functionality of its three main sections.
- ▶ [Section 7.4.4.2, “Creating sender-receiver to signal mapping”](#) provides instructions on how to create a sender-receiver to signal mapping.
- ▶ [Section 7.4.4.3, “Creating sender-receiver to signal group mapping”](#) provides instructions on how to create a sender-receiver to signal group mapping.
- ▶ [Section 7.4.4.4, “Creating client-server to signal mapping”](#) provides instructions on how to create a client-server to signal mapping.
- ▶ [Section 7.4.4.5, “Editing sender receiver to signal group mapping”](#) provides instructions on how to edit a sender-receiver to signal group mapping.
- ▶ [Section 7.4.4.6, “Removing data mappings”](#) provides instructions on how to remove data mappings.
- ▶ [Section 7.4.4.7, “Creating data mappings automatically”](#) provides instructions on how to automatically create data mappings.

7.4.4.1. Getting started with the Signal Mapping Editor

7.4.4.1.1. Opening the Signal Mapping Editor

To open the Signal Mapping Editor, proceed in the following way:

- ▶ Select the **System** section in the **Sidebar**.
- ▶ Double-click the **Edit System Signal Mappings** entry.

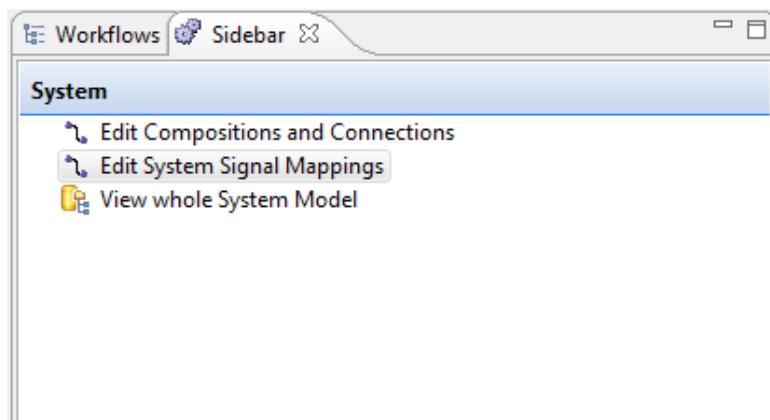


Figure 7.23. Opening the Signal Mapping Editor from the sidebar



7.4.4.1.2. The sections of the Signal Mapping Editor

The screenshot shows the "System Signal Mapping" editor window. At the top, there are two buttons: "Show Mapped Data Elements" and "Show Unmapped Data Elements". Below these are sections for "Edit System Signal Mappings" and "Automatic Mapping".

Edit System Signal Mappings:

Data Element	Type	Mapped to
RootSwComposition		
Client_Server_Port_1		
client_server_operation_x	DataMapping	CALL: /signals/S1 - RETURN: /signals/S3
Sender_Receiver_Port_1		
composite_data_element_y	RecordDataType	/signals/SignalGroup1
DataMapping		
primitive_data_element_x	Char8	/signals/S2
DataMapping		

Automatic Mapping:

Index	Run	Match Data Element	Match System Signal	Match System Signal Group	Comment

Buttons:

- Show Mapped Data Elements
- Show Unmapped Data Elements
- Save, Undo, Redo, Cut, Copy, Paste, Delete, New, Filter
- Match, Auto Map

Figure 7.24. The Signal Mapping Editor



Filters

In the **Filters** section you control which data elements to display in the tree table of the **Edit System Signal Mappings** section.

When you open the Signal Mapping Editor for the first time, all elements are displayed by default. This means, all filters are enabled.

If you apply any filter and close the Signal Mapping Editor, the filter remains active. If you re-open the Signal Mapping Editor, the previous session is restored.

Filters make defined items invisible in the tree. For example, if the system model contains a data element with `DataMapping`, you can hide this data element, by disabling the **Show Mapped Data Elements** filter. If the system model contains a data element without a `DataMapping`, you can hide this data element, by disabling the **Show Unmapped Data Elements**.

The filters in this section affect only the displayed elements in the tree table. The filters do not change the system model itself.

Edit System Signal Mappings

In the **Edit System Signal Mappings** section, you view the relevant elements of the project's system model in a tree table.

The Signal Mapping Editor displays the software components that are mapped directly or indirectly to the currently selected ECU and their ports. Compositions that are not mapped to the current ECU but contain components that are mapped to the current ECU are considered implicitly mapped to the current ECU. The Signal Mapping Editor displays only ports typed by a `SenderReceiverInterface` or a `ClientServerInterface`.

A port typed by a `SenderReceiverInterface` displays all the `VariableDataPrototype` instances as child data elements in the tree table. The column **Type** displays the type of this data element.

NOTE

ImplementationDataType not configured



If a data element has no `ImplementationDataType` configured, the warning message `SWCAS_22` is displayed. This element is not available in the Signal Mapping Editor.

A port typed by a `ClientServerInterface` displays all the `ClientServerOperation` instances as child data elements in the tree table.

If data mappings reference a given data element in the system model, the data element displays all these data mappings as child elements in the tree table.

You can modify the system model elements through the actions available in the context menu or the tool bar.

You can limit the displayed elements if you enter a search string in the text box **filter text**.

Automatic Mapping

In the **Automatic Mapping** section, you create data mappings automatically.



You specify match rules that consist of regular expression pairs.

You configure the match rules through the buttons in the toolbar, e.g. you change the order of the match rules.

You control which match rules to apply during automatic mapping.

7.4.4.2. Creating sender-receiver to signal mapping

In this chapter, you learn how to create `SenderReceiverToSignalMapping`.

NOTE Sender-receiver to signal mapping is restricted to specific data elements



You can create a `SenderReceiverToSignalMapping` for primitive data elements of a `PortPrototype` typed by a `SenderReceiverInterface` or to a `Byte-Array`.

To create a `SenderReceiverToSignalMapping` manually, proceed in the following way:

- ▶ Select a data element for which you wish to create a data mapping in the tree table that is displayed in the **Edit System Signal Mappings** section.
- ▶ Right-click on the data element to open the context menu.
- ▶ Select the **Add Sender-Receiver to Signal Mapping** menu item.

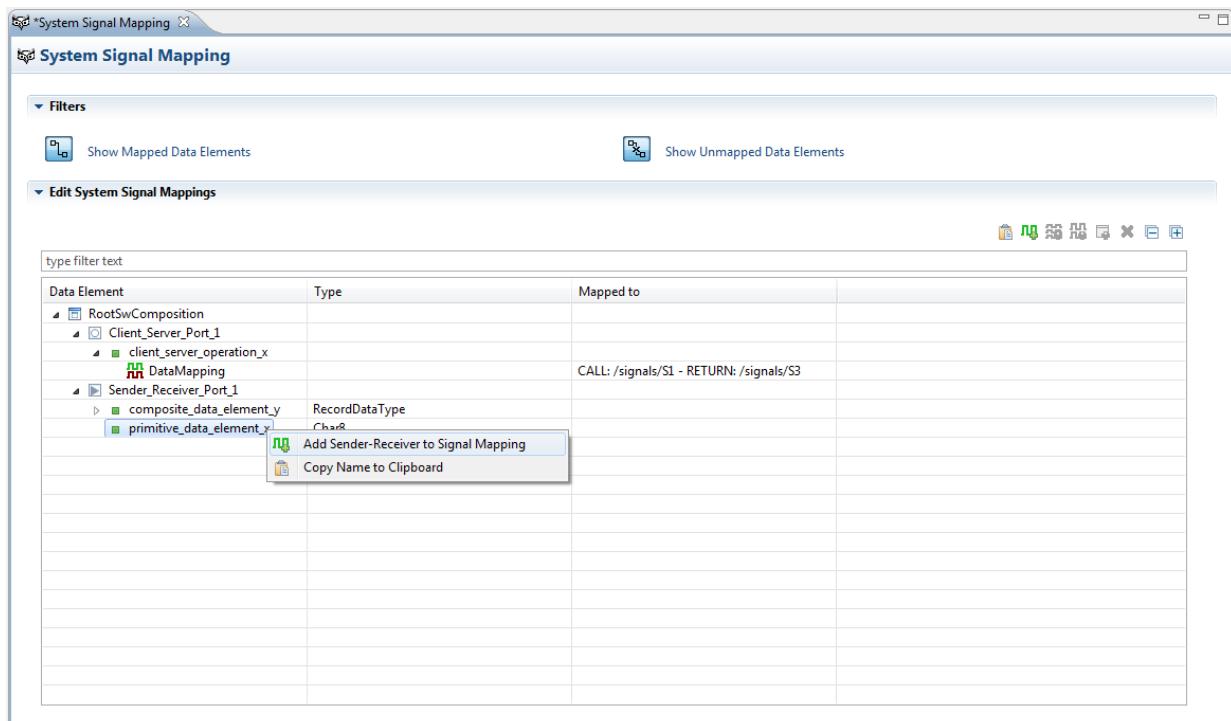


Figure 7.25. The **Add Sender-Receiver to Signal Mapping** menu item



- ▶ The **Add Sender-Receiver to Signal Mapping** dialog opens.

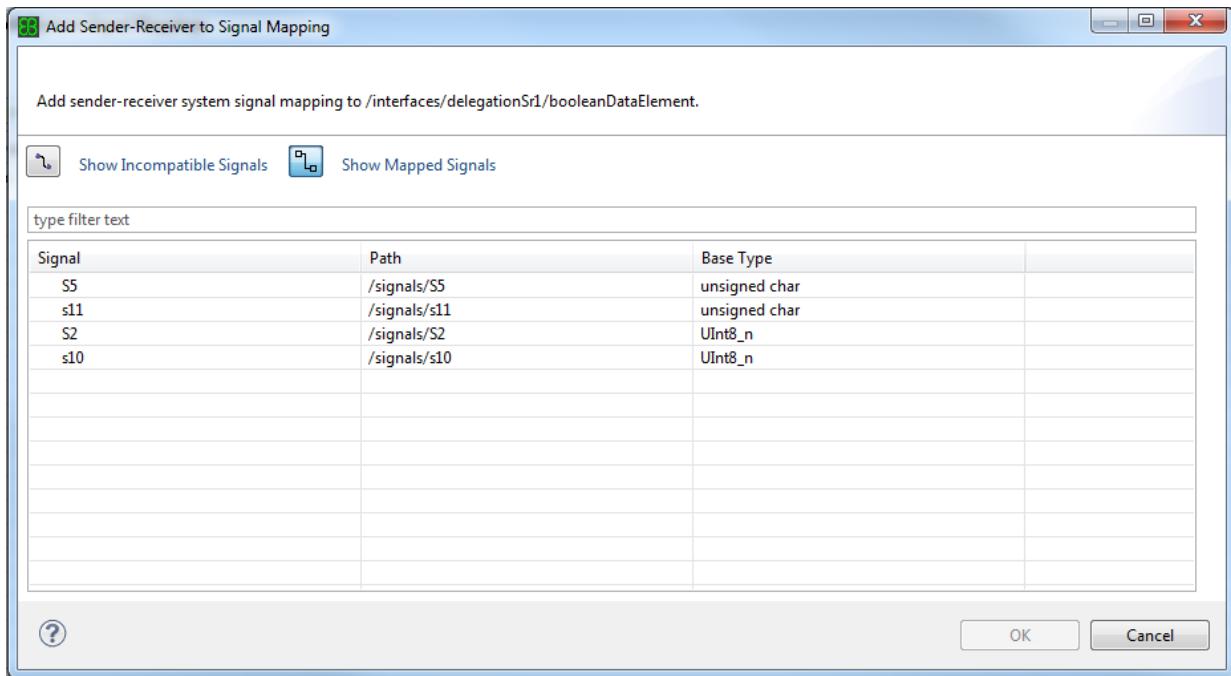


Figure 7.26. The **Add Sender-Receiver to Signal Mapping** dialog

- ▶ In the **Add Sender-Receiver to Signal Mapping** dialog, proceed in the following way:

- ▶ Optionally, filter the displayed elements in the tree table.

To filter the displayed elements in the tree table, use one of the following options:

- ▶ The **Show Incompatible Signals** button
- ▶ The **Show Mapped Signals** button

You can limit the displayed system signals if you enter a search string in the text box **type filter text**.

The displayed system signals are first sorted by the base type. If they have the same base type, then they are also sorted by the name. The system signals at the beginning of the list have the same base type as the DataElement for which the mapping is created. The system signals with base type UInt8_-n are displayed at the end of the list.

- ▶ Select a system signal in the tree table.
- ▶ Click the **OK** button.



NOTE**OK is enabled only if a system signal is selected**

You can click **OK** only after you have selected a system signal in the tree table.

A new SenderReceiverToSignalMapping is created.

7.4.4.3. Creating sender-receiver to signal group mapping

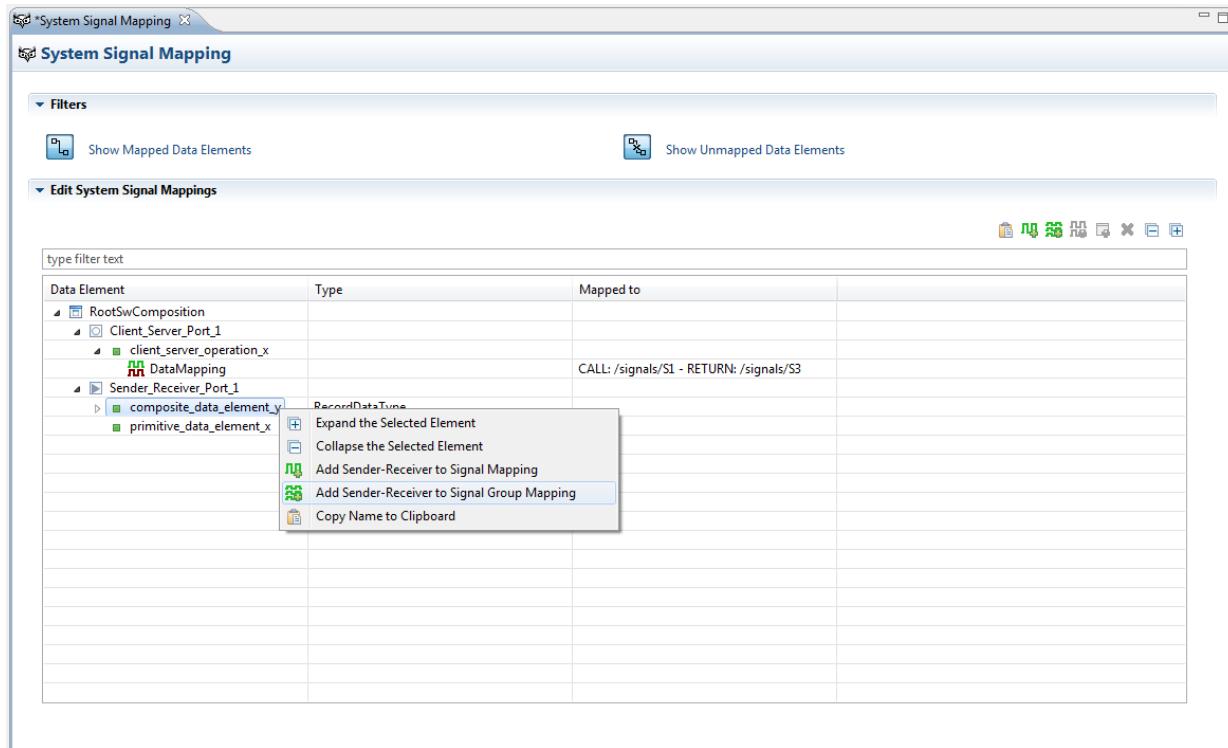
In this chapter, you learn how to create SenderReceiverToSignalGroupMapping.

NOTE**Sender-receiver to signal group mapping is restricted to specific data elements**

You can create a SenderReceiverToSignalGroupMapping only for data elements (typed by composite data types) of a PortPrototype typed by a SenderReceiverInterface.

To create a SenderReceiverToSignalGroupMapping manually, proceed in the following way:

- ▶ Select a data element for which you wish to create a data mapping in the tree table that is displayed in the **Edit System Signal Mappings** section.
- ▶ Right-click on the data element to open the context menu.
- ▶ Select the **Add Sender-Receiver to Signal Group Mapping** menu item.

Figure 7.27. The **Add Sender-Receiver to Signal Group Mapping** menu item

- The **Add Sender-Receiver to Signal Group Mapping** dialog opens.

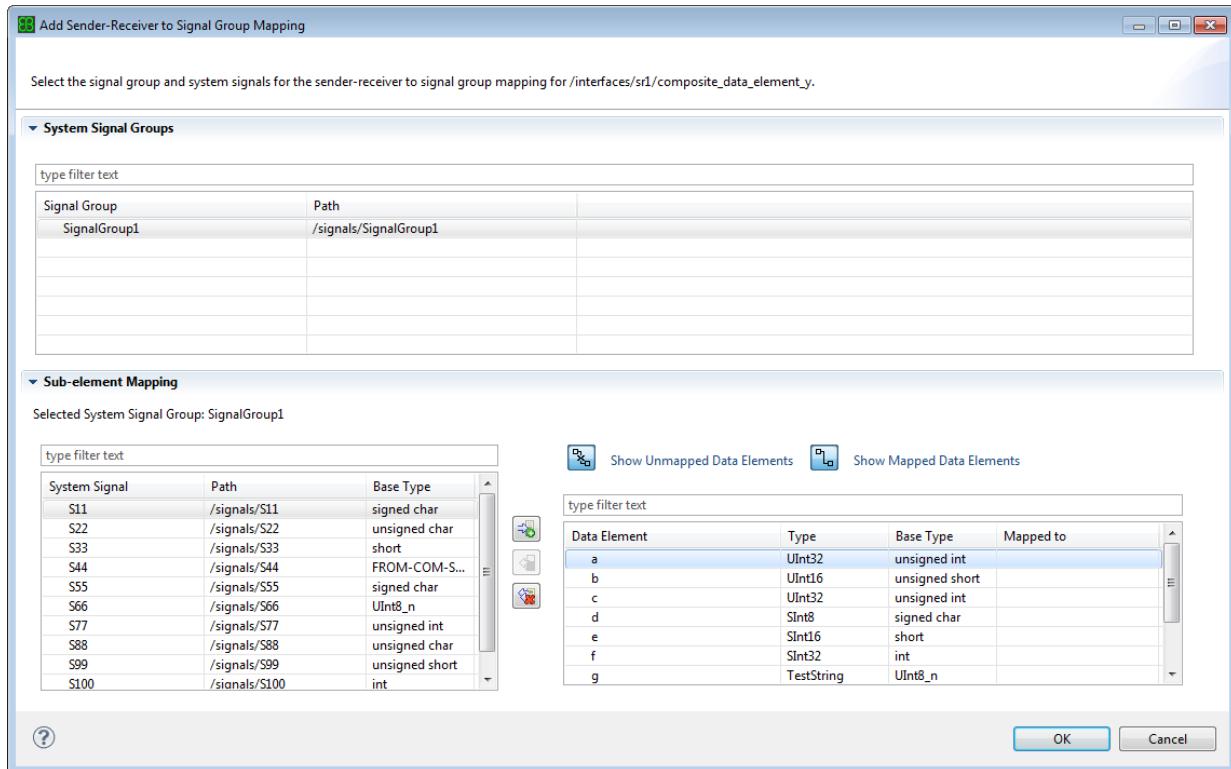


Figure 7.28. The Add Sender-Receiver to Signal Group Mapping dialog

- ▶ The table in the section **System Signal Groups** displays all the System Signal Group elements in the system model.

You can limit the displayed system signal groups if you enter a search string in the text box **type filter text**.

- ▶ The first table in the section **Sub-element Mapping** displays all the System Signal elements that are part of the selected system signal group.

You can limit the displayed system signals if you enter a search string in the text box **type filter text**.

- ▶ The second table in the section **Sub-element Mapping** displays all the sub-elements of the selected complex data element in a hierarchical way.

You can limit the displayed sub-elements if you enter a search string in the text box **type filter text**.

- ▶ In the **Add Sender-Receiver to Signal Group Mapping** dialog, proceed in the following way:

- ▶ Select a system signal group in the **System Signal Groups** section.
- ▶ To set a system signal for a sub-element, proceed in the following way:
 - ▶ Select a system signal and a sub-element in the **Sub-element Mapping** section.



- ▶ Click the button in the **Sub-element Mapping** section which is located between the **System Signal Table** and **Sub-element Table** .
- ▶ To clear a system signal from a sub-element, proceed in the following way:
 - ▶ Select a sub-element in the **Sub-element Mapping** section.
 - ▶ Click the button in the **Sub-element Mapping** section which is located between the **System Signal Table** and **Sub-element Table** .
 - ▶ To clear all system signals from the sub-elements, click the button in the **Sub- element Mapping** section which is located between the **System Signal Table** and **Sub- element Table** .
 - ▶ Click the **OK** button.

NOTE**OK is enabled only if a system signal group is selected**You can click **OK** only after you have selected a system signal group.

A new `SenderReceiverToSignalGroupMapping` is created.

NOTE**Only the first 100 subelements are displayed**

The **Add Sender-Receiver to Signal Group Mapping** dialog only displays the first 100 subelements of any data element of type array or nested composite data elements. This means you can assign signals only to the first 100 subelements

7.4.4.4. Creating client-server to signal mapping

In this chapter, you learn how to create `ClientServerToSignalMapping`.

NOTE**Client-server to signal mapping is restricted to specific operations**

You can create a `ClientServerToSignalMapping` only for `ClientServerOperation` instances of a `PortPrototype` typed by a `ClientServerInterface`.

To create `ClientServerToSignalMapping` manually, proceed in the following way:

- ▶ Select a client-server operation for which you wish to create a data mapping in the tree table that is displayed in the **Edit System Signal Mappings** section.
- ▶ Right-click on the client-server operation to open the context menu.
- ▶ Select the **Add Client-Server to Signal Mapping** menu item.

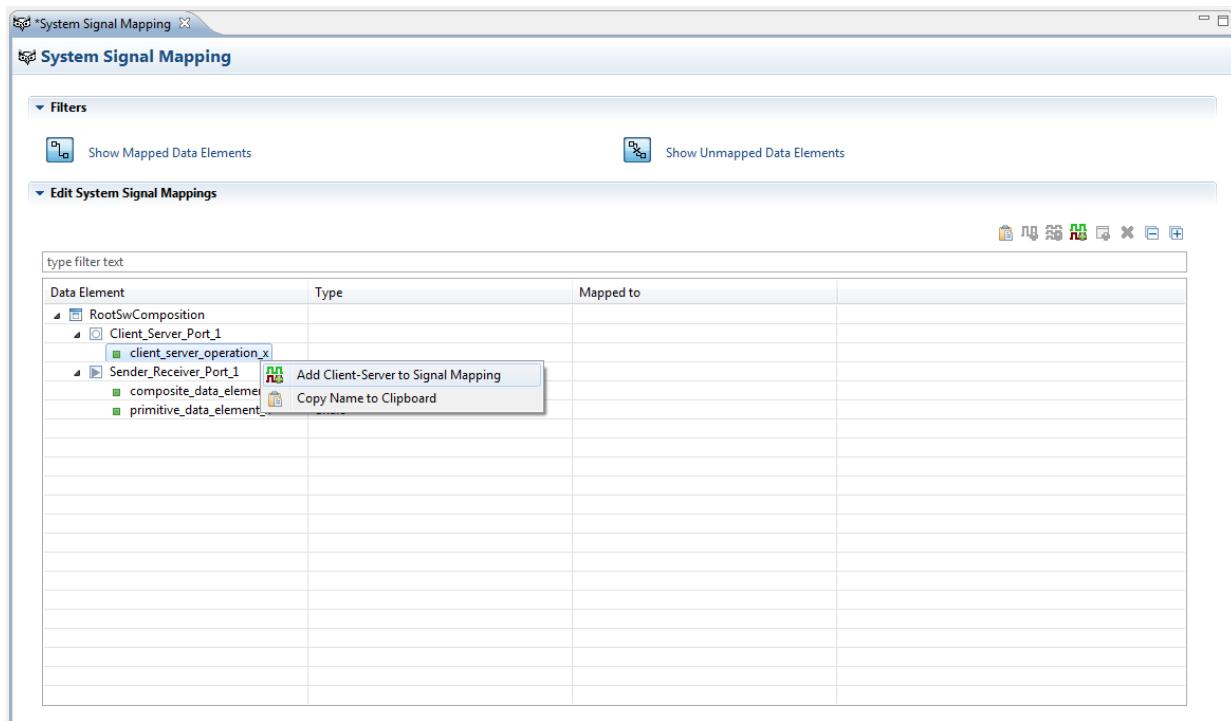
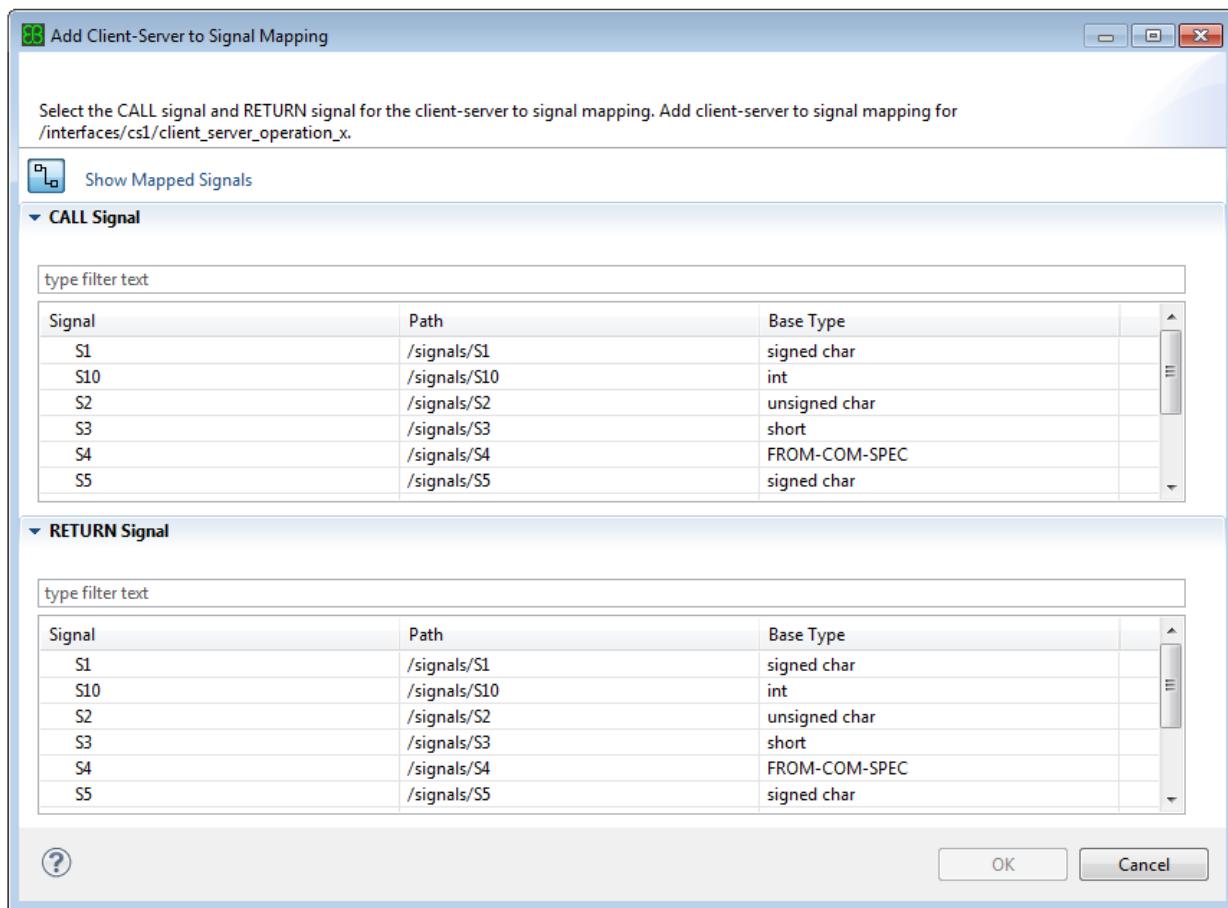


Figure 7.29. The **Add Client-Server to Signal Mapping** menu item

- The **Add Client-Server to Signal Mapping** dialog opens.

Figure 7.30. The **Add Client-Server to Signal Mapping** dialog

► In the **Add Client-Server to Signal Mapping** dialog, proceed in the following way:

► Optionally, filter the displayed elements in the tree table.

To filter the displayed elements in the tree table, use the **Show Mapped Signals** button.

You can limit the displayed elements if you enter a search string in the text box **type filter text**.

► Select a call signal and a return signal in the tree table.

► Click the **OK** button.

NOTE

OK is enabled only if a return signal is selected

You can click **OK** only after you have selected a return signal.



A new ClientServerToSignalMapping is created.



7.4.4.5. Editing sender receiver to signal group mapping

In this chapter, you learn how to edit sender receiver to signal group mappings.

To edit a sender-receiver to signal group mappings manually, proceed in the following way:

- ▶ Select the data mapping that you want to edit in the tree table that is displayed in the **Edit Sender-Receiver to Signal Group Mapping** section.
- ▶ Right-click on the data mapping to open the context menu.
- ▶ Select the **Edit Sender-Receiver to Signal Group Mapping** menu item.

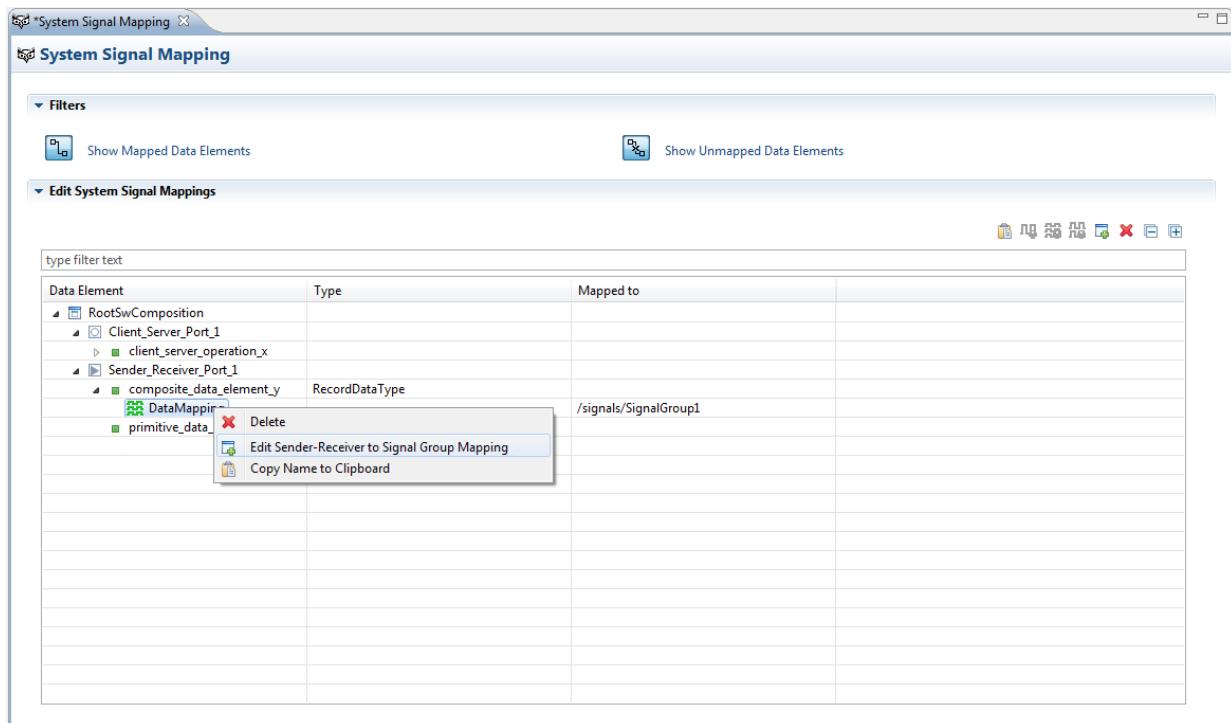
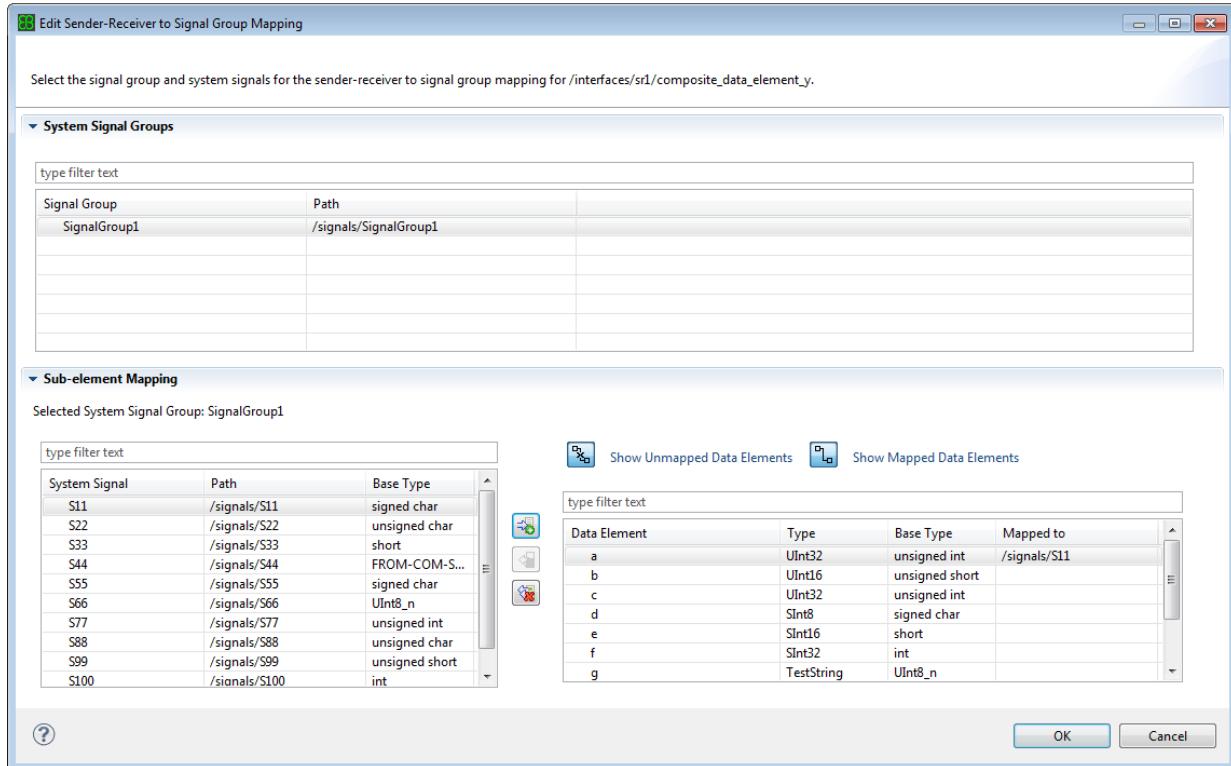


Figure 7.31. The **Edit Sender-Receiver to Signal Group Mapping** menu item

- ▶ The **Edit Sender-Receiver to Signal Group Mapping** dialog opens.

Figure 7.32. The **Edit Sender-Receiver to Signal Group Mapping** dialog

- ▶ In the **Edit Sender-Receiver to Signal Group Mapping** dialog, proceed in the following way:
 - ▶ To change a system signal group for the selected data mapping, proceed in the following way:
 - ▶ Select the desired signal group in the **System Signal Groups** section.
 - A confirmation dialog opens.
 - ▶ In the confirmation dialog, click the **OK** button if you want to change the system signal group.
 - All the system signals that are assigned to the sub-elements are cleared.
 - ▶ Click the **Cancel** button if you do not want to change the system signal group.
 - ▶ To set/change a system signal for a sub-element, proceed in the following way:
 - ▶ Select a system signal and a sub-element in the **Sub-element Mapping** section.
 - ▶ Click the button in the **Sub-element Mapping** section which is located between the **System Signal Table** and **Sub-element Table**.
 - ▶ To clear a system signal from a sub-element, proceed in the following way:
 - ▶ Select a sub-element in the **Sub-element Mapping** section.
 - ▶ Click the button in the **Sub-element Mapping** section which is located between the **System Signal Table** and **Sub-element Table**.



- ▶ To clear all the system signals from the sub-elements, click the button in the **Sub- element Mapping** section which is located between the **System Signal Table** and **Sub- element Table** .
- ▶ Click the **OK** button.

NOTE

OK is enabled only if a system signal group is selected

You can click **OK** only after you have selected a system signal group.



The `SenderReceiverToSignalGroupMapping` is edited.

NOTE

Only sender-receiver to signal group mappings can be edited



You can only edit sender-receiver to signal group mappings. You cannot edit other data mappings.

NOTE

Only mappings created with the Signal Mapping Editor can be edited



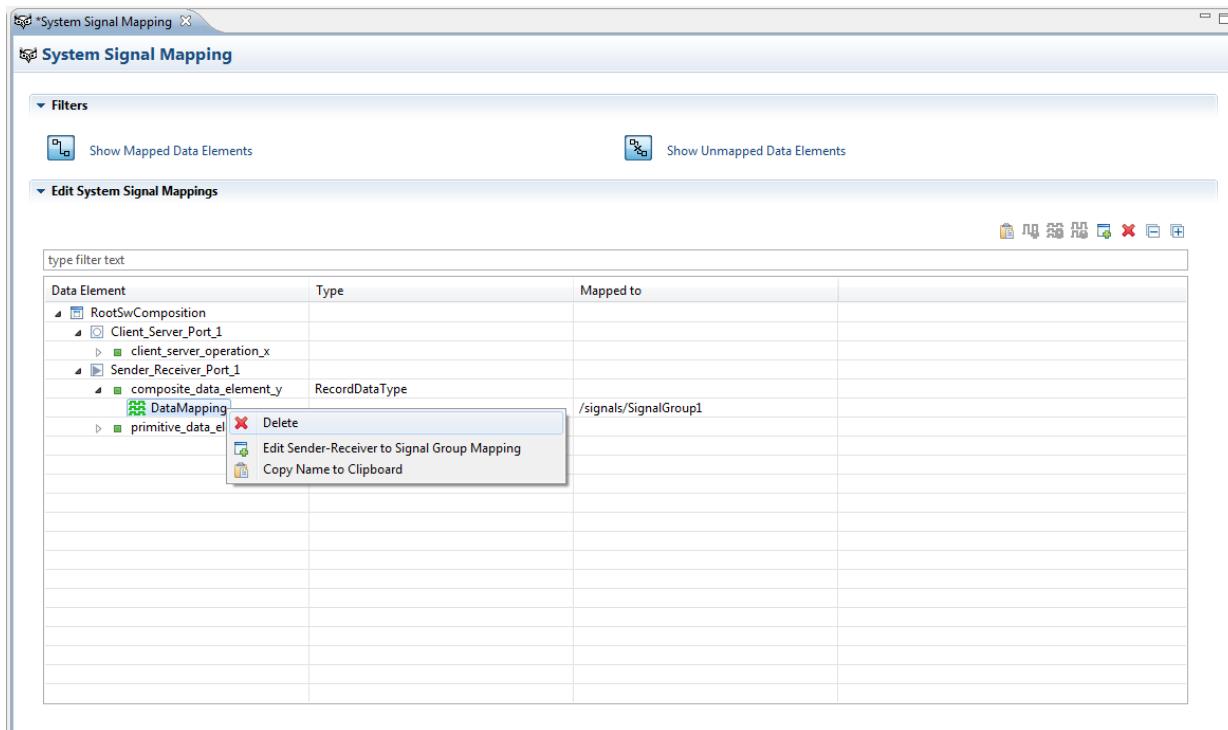
You can edit a sender-receiver to signal group mapping only if it was previously created using the Signal Mapping Editor.

7.4.4.6. Removing data mappings

In this chapter, you learn how to remove data mappings

To remove data mappings, proceed in the following way:

- ▶ Right-click on the data mapping to open the context menu.
- ▶ Select the **Delete** menu item.

Figure 7.33. The **Delete data mapping** menu item

- ▶ The DataMapping is removed.



NOTE **Only data mappings created with the Signal Mapping Editor can be removed**

You can remove a data mapping only if it was previously created using the Signal Mapping Editor.

To remove multiple data mappings, proceed in the following way:

- ▶ Select one of the elements you want to remove.
- ▶ Press and hold **Ctrl** or **Shift**.
- ▶ Select the other elements you want to remove.
- ▶ Release **Ctrl** or **Shift**.
- ▶ Right-click the element to open the context menu.
- ▶ Select the **Delete** menu item.

The selected elements are removed.

NOTE

Multiple selection delete warning

If an element cannot be deleted, a warning is logged.



7.4.4.7. Creating data mappings automatically

The Signal Mapping Editor supports an automatic mapping feature which creates data mapping for data elements automatically. The data element and the system signal or system signal group match based on their short name. The data mapping is based on match rules that consist of regular expression pairs. In this chapter, you learn how to create data mappings automatically.

Figure 7.34. The **Automatic Mapping** section in the Signal Mapping Editor

To create data mappings automatically, proceed in the following way:

- ▶ Add a match rule. To add a match rule, click the **Add new element with default values** button .
 - ▶ Specify the match rule:
 - ▶ Specify the match source regular expression in the **Match Data Element** column.
 - ▶ Specify the match target regular expression in the **Match System Signal** or **Match System Signal Group** column.

**TIP****Use the short name of an element to make a regular expression**

Copy the short name of an element in the **Edit System Signal Mappings** tree table and make a match source regular expression from it.

TIP**Both columns can be filled with data**

If you enter a system signal name and a system signal group name in the same row, so, both columns **Match System Signal** and **Match System Signal Group** are filled with information. Then, if the primitive data element will match the **Match System Signal** column this one will have a higher priority. Otherwise if the composite data element will match the **Match System Signal Group** column this one will have a higher priority.

To make a regular expression from an element's short name:

- ▶ Select an element in the **Edit System Signal Mappings** tree table.
- ▶ Click on the **Copy Name to Clipboard** button
- ▶ Insert the content of the clipboard into the **Match Data Element** table.

TIP**Use capture groups in match rules**

Use capture groups in the match data element expression, for example ([a-zA-Z0-9]+_+). Reference the capture groups in the match system signal expression. To reference the capture groups use \#ofGroup, for example \1 to reference the first capture group of the data element expression. Non-capture groups denoted by (?:) are not counted in the group reference number. Group references in the match system signal expression are substituted by the current value of the group before the regular expression is used to find a match system signal.

- ▶ If you need further match rules, repeat the above steps.
- ▶ Optionally, change the order of match rules.

To change the order of match rules, move match rules up or down:

- ▶ To move up a match rule, select the match rule you want to apply, and click the **Move up selected elements** button
- ▶ To move down a match rule, select the match rule you want to apply, and click the **Move down selected elements** button
- ▶ Optionally, duplicate match rules.

To duplicate a match rule, select the match rule you want to apply, and click on the **Duplicate selected elements** button . The selected match rule is copied and added to the end of the list. The state of the check box in the **Run** column of the match rule is also copied.



- ▶ Optionally, remove match rules.

To remove a match rule, select the match rule you want to apply, and click on the **Remove selected elements** button . The selected match rule is deleted. The index for the remaining match rules is adapted to be zero-based consecutive.

- ▶ Check the check box in the **Run** column of each match rule you want to apply. Only checked match rules are considered during automatic data mapping creation.

NOTE**Match rules are performed in index order, regardless of table appearance**

You are able to change the table appearance through sorting the columns by criteria other than the index. However, automatic connection performs the match rules in the zero-based consecutive order of the index.

-
- ▶ Click the **Auto Map** button.

The source regular expression is evaluated for all data elements. If the short name of a data element matches this expression, a match system signal is searched. For every match of the respective source expression, the target expression is evaluated for all system signals of the system. If the short name of a target system signal matches this expression and it is compatible to the source data element, a sender-receiver to signal mapping is created. The automatic mapping creates sender-receiver to signal mapping as described in [Section 7.4.4.2, “Creating sender-receiver to signal mapping”](#). If the short name of a target system signal group matches this expression, a sender-receiver to signal group mapping is created. The automatic mapping creates sender-receiver to signal group mapping as described in [Section 7.4.4.3, “Creating sender-receiver to signal group mapping”](#).



WARNING**Match expressions must match a unique data element short name and a system signal or system signal group short name**

To avoid unwanted behavior, take care when you design the regular expressions for automatic mapping.

Automatic mapping may create data mappings more than once if such mappings are valid. This may, for example, happen if several expressions match the same system signal or system signal group.

The match does only consider the short name of a data element and the short name of the system signal or system signal group. This means that if multiple data elements or multiple system signals (or system signal groups) with same short name exist, you may get unexpected results.

NOTE**Match system signal group column availability**

Match System Signal Group column, where you can enter the signal group name to perform automapping is available only if the composite data elements of a `PortPrototype` typed by a `SenderReceiverInterface` are available in the tree table of data elements. Otherwise the column is blocked.

7.4.4.8. Proceeding after data mappings were created

After you created data mappings, proceed in the following way:

1. Save the changes to the system model.

If you close the Signal Mapping Editor without having saved your changes, a dialog opens and you are asked if you want to save the changes. To save the changes, click **Yes**.

Your changes are saved to the system model and written to the file `ConnectionEditor.arxml` that is located in the folder `systemmod` inside your project.

2. Update the ECU extract after the changes have been saved to the system model.

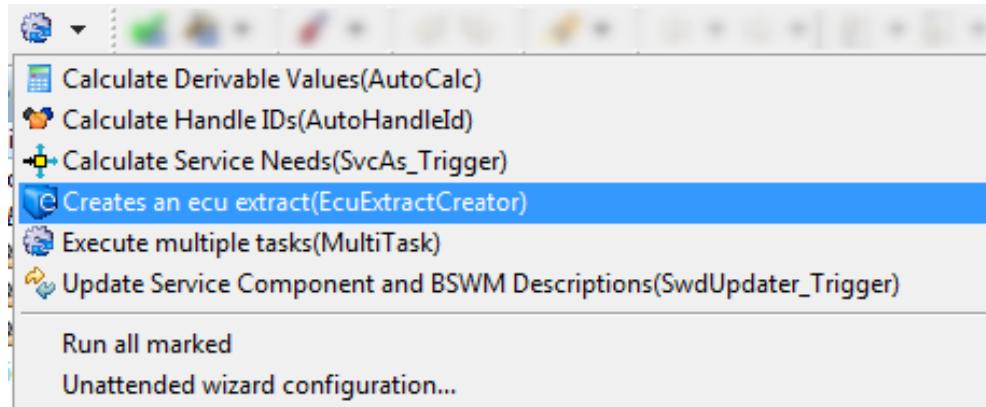


Figure 7.35. Triggering the *Creates an ecu extract* unattended wizard

NOTE

Update the ECU extract to make changes available to the Rte



If you do not update the ECU extract, your changes will not be available for the Rte module configuration.

7.5. Using the Auto-connect SWC unattended wizard

Use the Auto-connect SWC unattended wizard to auto-connect software components for an existing EB tresos Studio project stored in an EB tresos Studio workspace. You can execute the wizard in EB tresos Studio or via the command line. In both cases, you must first configure the wizard.

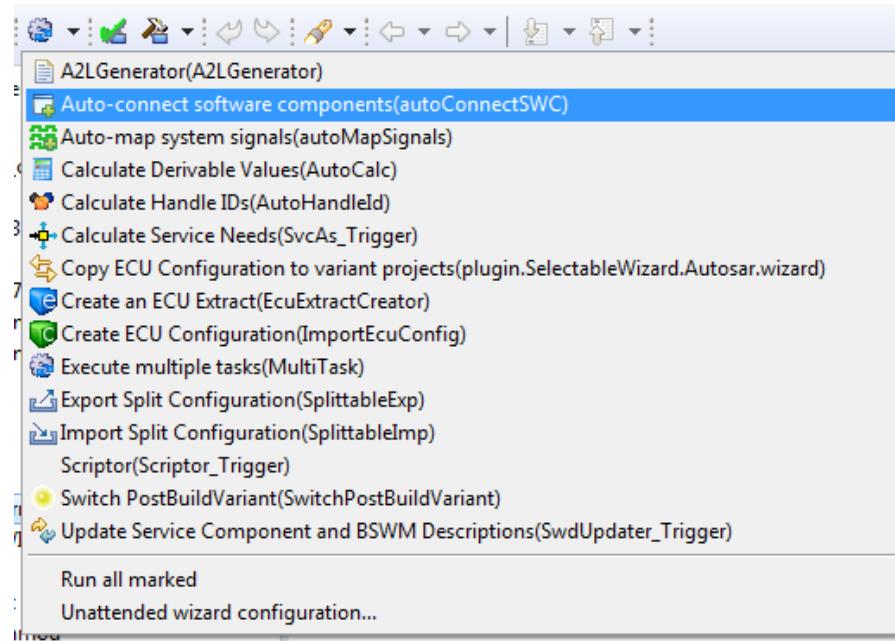


Figure 7.36. Location of the Auto-connect SWC unattended wizard

7.5.1. Configuring the Auto-connect SWC unattended wizard

Before you can run the Auto-connect SWC unattended wizard, you must configure it in EB tresos Studio. Open **Unattended wizard configuration** and select **Auto-connect software components**. You can either fill in the Match table manually, or you can select to import a rule file. Any changes you make in the Match table are not included in the rule file. The content of the Match table is taken into account when you execute the unattended wizard. For more information about the rule file, see [Section 7.5.1.2, “Rule file”](#).

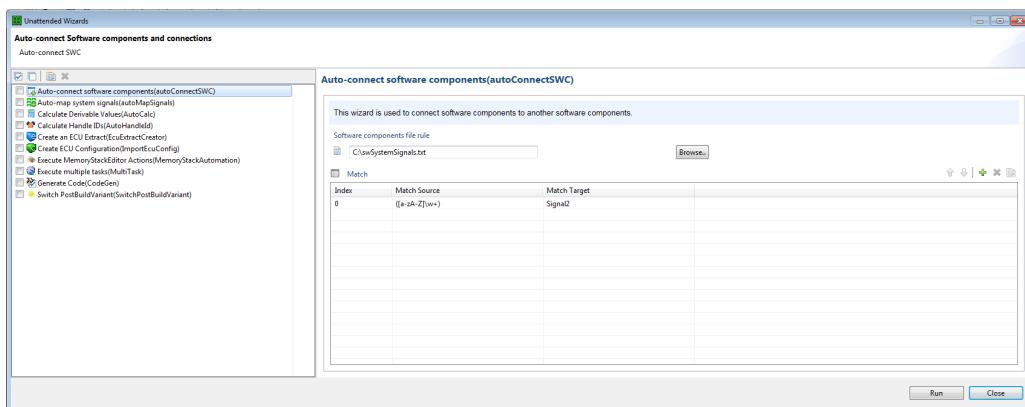


Figure 7.37. Configuring the Auto-connect SWC unattended wizard

7.5.1.1. Match table

For the configuration of the Auto-connect SWC unattended wizard, you can choose to use a rule file or fill in the Match table manually. The Match table contains one or several rules for mapping the sources to targets. Note that your changes to the Match table are not included in the rule file. The information in the Match table is used for the execution of the unattended wizard.

The following picture shows an example of a rule in the Match table.

Index	Match Source	Match Target
0	{[a-zA-Z]+w+}	Signal2

Figure 7.38. Example of a rule in the Match table

7.5.1.2. Rule file

For the configuration of the Auto-connect SWC unattended wizard, you can choose to use a rule file. This file contains one or several rules for matching the software component ports. The information in the rule file is used as input to the Match table.

The file must be a text file in .txt format. In this text file, you provide the regular expressions for the source ports and their matching target ports. Insert the key word `regex` between the regular expressions. See an example of a rule in the following picture.

`(?:[a-zA-Z0-9]+){3}ID(\d+) regex (?:[a-zA-Z0-9]+){3}ID\1`

Figure 7.39. Example of a rule in the rule file

7.5.2. Validation mechanism

There is a validation mechanism for the Auto-connect SWC unattended wizard that is based on the input to the wizard.

The following table shows the validation messages and the corresponding errors.

Message	Error
(Code: SWCAS_40, Severity: ERROR) There are no rules inside the template rule file! Please check and	There is a problem in the rule file.



Message	Error
correct the template and make sure the rules are delimited by 'regex'.	
(Code: SWCAS_43, Severity: ERROR) There are no rules inside the auto-connect table. Please insert source and target match in order to run this wizard!	The Match table is empty.
(Code: SWCAS_44, Severity: ERROR) For the row index {0}, please insert the source match in order to run this wizard!	The source match on row index {0} is empty.
(Code: SWCAS_45, Severity: ERROR) For the row index {0}, please insert the target match in order to run this wizard!	The target match on row index {0} is empty.

7.5.3. Executing the Auto-connect SWC unattended wizard

After you configured the wizard in EB tresos Studio, you can execute it directly in EB tresos Studio or on the command line.

7.5.3.1. Executing the Auto-connect SWC unattended wizard on the command line

To execute an unattended wizard, the following command syntax applies:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    autoconfigure <project> [<wizard>]...
```

To execute the Auto-connect SWC unattended wizard as configured in EB tresos Studio on a project called *myproject*, use:

```
tresos_cmd.bat autoconfigure myproject autoConnectSWC
```

For more information on executing unattended wizards, see [Section 6.12.2.10, “Executing unattended wizards on projects”](#).

7.5.4. Logging mechanism

After executing the Auto-connect SWC unattended wizard on the command line, you receive feedback in the command line window. For more information, consult the `.log` file in your current workspace.



If you executed the Auto-connect SWC unattended wizard in EB tresos Studio, you see the results in the **Results** view as shown in [Figure 7.40, “Example of the Results view”](#). The displayed table contains the following information:

Composition/Prototype/Source Port Names	Composition/Prototype/Target Port Names
Displays all source ports that were connected to the target ports displayed in the second column.	Displays all target ports that were connected to the source ports.
The ports are displayed with their prototypes and compositions. So you can see where these ports are in the system composition.	The ports are displayed with their prototypes and compositions. So you can see where these ports are in the system composition.

The screenshot shows the EB tresos Studio interface with the 'Results' tab selected. The title bar includes 'Search', 'Problems View', 'Error Log', and 'Results'. Below the title bar, it says 'Overview Auto-connect software components(autoConnectSWC)' and the date '2018-10-03 10:24:36'. A link 'Back to Overview tab' is present. The main area contains two tables and a message log.

Composition/Prototype/Source Port Names	Composition/Prototype/Target Port Names
I/A/SwcA_Port_a_ID1	II/I/SwcA_Port_a_ID1
II/I/SwcA_Port_a_ID1	I/A/SwcA_Port_a_ID1
II/I/SwcA_Port_a_ID1	II/C/SwcC_Port_c_ID1
II/I/SwcA_Port_a_ID1	II/C/SwcC_Port_c_ID1
I/B/SwcB_Port_b_ID11	II/I/SwcB_Port_b_ID11
II/I/SwcB_Port_b_ID11	I/B/SwcB_Port_b_ID11
II/I/SwcB_Port_b_ID11	III/I/SwcB_Port_b_ID11
III/I/SwcB_Port_b_ID11	II/I/SwcB_Port_b_ID11
I/E/SwcE_Port_e_ID11	III/I/SwcE_Port_e_ID11
III/I/SwcE_Port_e_ID11	IV/E/SwcE_Port_e_ID11
III/I/SwcB_Port_b_ID11	III/I/SwcE_Port_e_ID11
III/I/SwcB_Port_b_ID11	III/I/SwcE_Port_e_ID11

Message
:(SWCAS_35) Auto-connect finished. There were added 2 connections - (delegation/assembly).
:(SWCAS_34) The following connection has been added: source port I/A/SwcA_Port_a_ID1 connected to the target port II/I/SwcA_Port_a_ID1.
:(SWCAS_34) The following connection has been added: source port II/I/SwcA_Port_a_ID1 connected to the target port I/A/SwcA_Port_a_ID1.
:(SWCAS_34) The following connection has been added: source port II/I/SwcA_Port_a_ID1 connected to the target port II/C/SwcC_Port_c_ID1.
:(SWCAS_34) The following connection has been added: source port II/I/SwcA_Port_a_ID1 connected to the target port II/C/SwcC_Port_c_ID1.
:(SWCAS_34) The following connection has been added: source port I/B/SwcB_Port_b_ID11 connected to the target port II/I/SwcB_Port_b_ID11.
:(SWCAS_34) The following connection has been added: source port II/I/SwcB_Port_b_ID11 connected to the target port I/B/SwcB_Port_b_ID11.
:(SWCAS_34) The following connection has been added: source port II/I/SwcB_Port_b_ID11 connected to the target port III/I/SwcB_Port_b_ID11.
:(SWCAS_34) The following connection has been added: source port III/I/SwcB_Port_b_ID11 connected to the target port II/I/SwcB_Port_b_ID11.
:(SWCAS_34) The following connection has been added: source port IV/E/SwcE_Port_e_ID11 connected to the target port III/I/SwcE_Port_e_ID11.
:(SWCAS_34) The following connection has been added: source port III/I/SwcE_Port_e_ID11 connected to the target port IV/E/SwcE_Port_e_ID11.
:(SWCAS_34) The following connection has been added: source port III/I/SwcB_Port_b_ID11 connected to the target port III/I/SwcE_Port_e_ID11.
:(SWCAS_34) The following connection has been added: source port III/I/SwcB_Port_b_ID11 connected to the target port III/I/SwcE_Port_e_ID11.
:(Info

Figure 7.40. Example of the Results view

7.5.4.1. ConnectionEditor.arxml file

After executing the Auto-connect SWC unattended wizard, the file `ConnectionEditor.arxml` is automatically created in the `systemmod` folder. This folder is located inside your project in the current workspace.

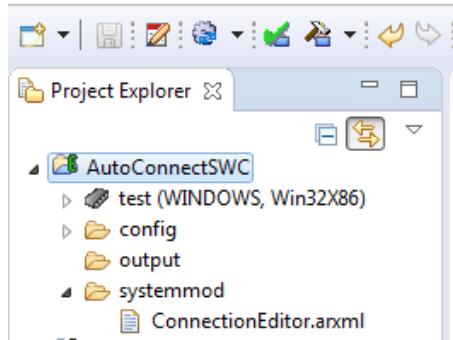


Figure 7.41. Location of the ConnectionEditor.arxml file

7.6. Using the Auto-map system signals unattended wizard

Use the Auto-map system signals unattended wizard to auto-map system signals for an existing EB tresos Studio project stored in an EB tresos Studio workspace. You can execute the wizard in EB tresos Studio or via the command line. In both cases, you must first configure the wizard.

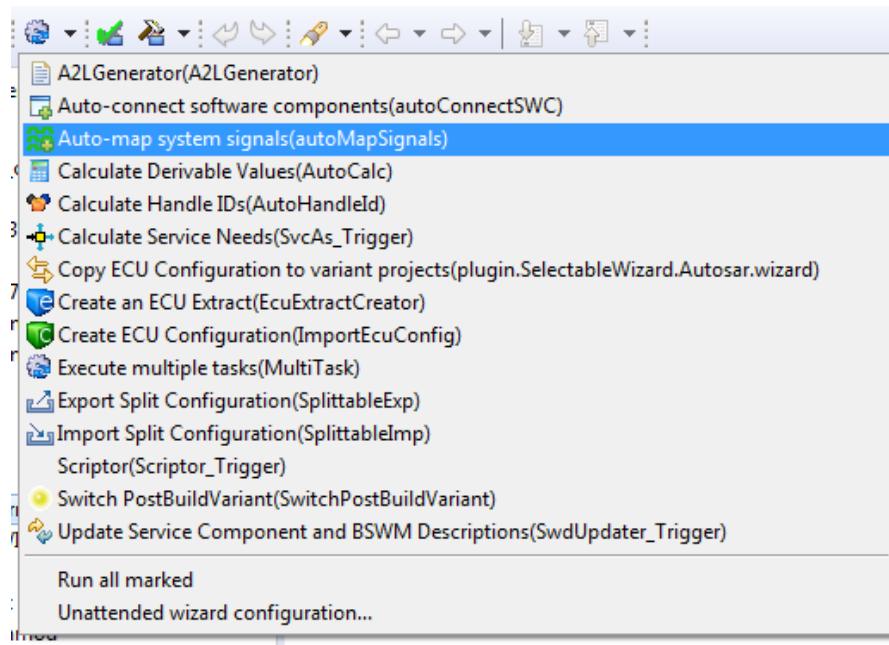


Figure 7.42. Location of the Auto-map system signals unattended wizard



7.6.1. Configuring the Auto-map system signal unattended wizard

Before you can run the Auto-map system signal unattended wizard, you must configure it in EB tresos Studio. Open **Unattended wizard configuration** and select **Auto-map system signals**. You can either fill in the Match table manually, or you can select to import a rule file. Any changes you make in the Match table are not included in the rule file. The content of the Match table is taken into account when you execute the unattended wizard. For more information about the rule file, see [Section 7.6.1.2, “Rule file”](#).

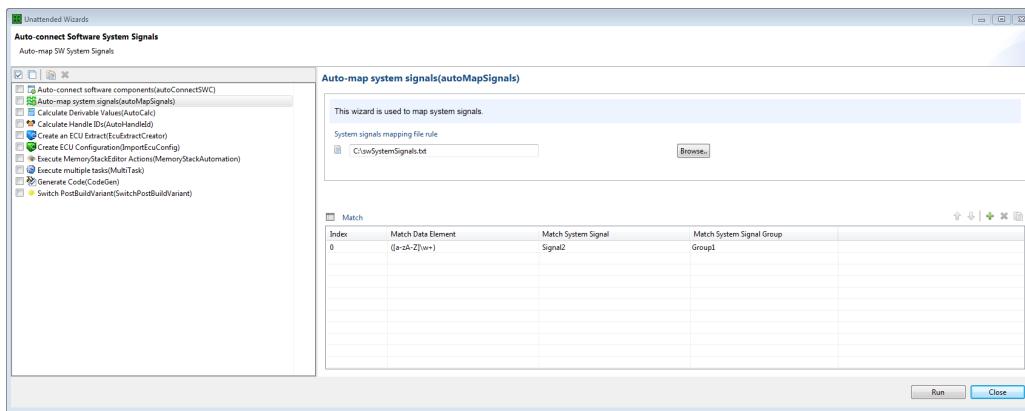


Figure 7.43. Configuring the Auto-map system signal unattended wizard

7.6.1.1. Match table

For the configuration of the Auto-map system signal unattended wizard, you can choose to use a rule file or fill in the Match table manually. The Match table contains one or several rules for mapping the data elements to system signals. Note that your changes to the Match table are not included in the rule file. The information in the Match table is used for the execution of the unattended wizard.

The following picture shows an example of a rule in the Match table.

Index	Match Data Element	Match System Signal	Match System Signal Group
0	([a-zA-Z]\w*)	Signal2	Group1

Figure 7.44. Example of a rule in the Match table



7.6.1.2. Rule file

For the configuration of the Auto-map system signal unattended wizard, you can choose to use a rule file. The rule file contains one or several rules for mapping the data elements to system signals. The information in the rule file is used as input to the Match table.

The file must be a text file in `.txt` format. In this text file, you provide the regular expressions for the data elements and the mapped system signals. Insert the key word `regex` between the regular expressions. See an example of a rule in the following picture.

Figure 7.45. Example of a rule in the rule file

7.6.2. Validation mechanism

There is a validation mechanism for the Auto-map system signal unattended wizard that is based on the input to the wizard.

The following table shows the validation messages and corresponding errors.

Message	Error
(Code: SWCAS_40, Severity: ERROR) There are no rules inside the template rule file! Please check and correct the template and make sure the rules are delimited by 'regex'.	There is a problem in the rule file.
(Code: SWCAS_46, Severity: ERROR) There are no rules inside the auto-map table. Please insert match data element and match system signal or match system signal group in order to run this wizard!	The Match table is empty.
(Code: SWCAS_47, Severity: ERROR) For the row index {0}, please insert the match data element in order to run this wizard!	The match data element on row index {0} is empty.
(Code: SWCAS_48, Severity: ERROR) For the row index {0}, please insert the match system signal or match system signal group in order to run this wizard!	The match system signal and match system signal group on row index {0} is empty.



7.6.3. Executing the Auto-map system signal unattended wizard

After you configured the wizard in EB tresos Studio, you can execute it directly in EB tresos Studio or on the command line.

7.6.3.1. Executing the Auto-map system signal unattended wizard on the command line

To execute an unattended wizard, the following command syntax applies:

```
tresos_cmd.bat [<system_property>...] [-data <workspace>]
    autoconfigure <project> [<wizard>]...
```

To execute the Auto-map system signal unattended wizard as configured in EB tresos Studio on a project called *myproject*, use:

```
tresos_cmd.bat autoconfigure myproject autoMapSignals
```

For more information on executing unattended wizards, see [Section 6.12.2.10, “Executing unattended wizards on projects”](#).

7.6.4. Logging mechanism

After executing the Auto-map system signal unattended wizard on the command line, you receive feedback in the command line window. For more information, consult the `.log` file in your current workspace.

If you executed the Auto-map system signal unattended wizard in EB tresos Studio, you see the results in the **Results** view as shown in [Figure 7.46, “Example of the Results view”](#). The displayed table contains the following information:

Data Element Kind/Data Element	System Signal
Displays all data elements as source that were mapped to the target system signals displayed in the second column.	Displays all target system signals that were mapped to data elements inside the system. The system signals are displayed from their hierarchical composition in the imported system.



Data Element Kind/Data Element	System Signal
The data elements are displayed from their hierarchical composition. The element kind is also displayed. For arrays, the element kind can be simple or complex.	

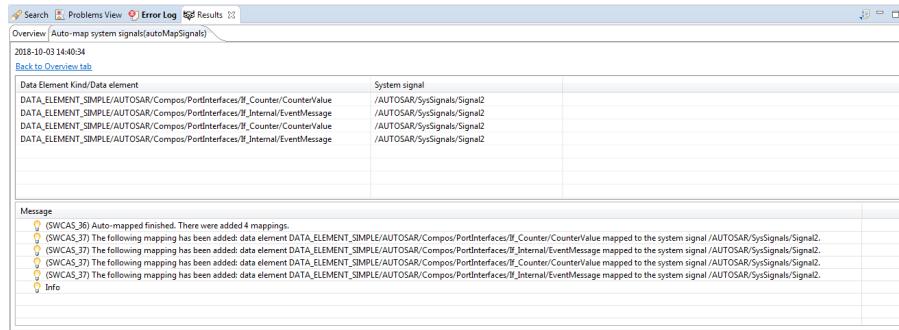


Figure 7.46. Example of the Results view

7.6.4.1. ConnectionEditor.arxml file

After executing the Auto-map system signal unattended wizard, the file `ConnectionEditor.arxml` is automatically created in the `systemmod` folder. This folder is located inside your project in the current workspace.

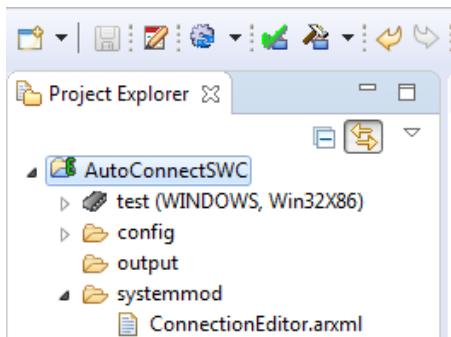


Figure 7.47. Location of the ConnectionEditor.arxml file

7.7. Using the Edit PostBuildVariants wizards

The **Edit PostBuildVariants** wizards are available in the **System** category of the **Sidebar** view. Two types of wizards exist for the different PostBuild concepts described in [Section 6.13.1.2, “PostBuild variation points”](#):

Edit Loadable PostBuildVariants

Use this wizard to create and configure loadable variants and criterions.



Edit Selectable PostBuildVariants

Use this wizard to create and configure selectable variants and criterions.

NOTE
Creation of new elements using these wizards


All elements (variants, criterions, etc.) that are created using these wizards are stored in a separate package named *EB*.

WARNING

Editability of elements

You can only change or delete elements (variant, criterions, etc.) that were created using this wizard and are therefore located in the *EB* package.

All other elements are read-only to prevent changing of elements delivered by OEM which may get re-imported during project lifecycle.

You can do the following with the **Edit PostBuildVariants** wizards:

- ▶ Create, rename, and delete variants
- ▶ Create, rename, and delete criterions
- ▶ Configure textual representations
- ▶ Configure references between criterions and variants

NOTE
Undo/redo support


All actions provide undo/redo functionality.

Both wizards provide the same functionality. The only difference is the data each wizard is working on. Therefore the following description is applicable for both wizards.

The wizard consists of two pages:

- ▶ [Section 7.7.1, “The Predefined Variants page”](#)
- ▶ [Section 7.7.2, “The Variant Criterions page”](#)

When you close the wizard, a confirmation dialog is displayed which asks you whether you want to save the changes to the system model or not.

Every time you close the wizard the part of the EB package which belongs to the type of the wizard (loadable or selectable) is exported as .arxml file to the location <projectName>/systemmod/PostBuildVariants<type>.arxml.

The previous EB tresos Studio version had only one wizard that exported the data to the <projectName>/systemmod/PostBuildVariants.arxml file. If a file with such a name exists, it is renamed to prevent you from re-importing outdated data.



When you re-import your system data, e.g. you received a new version from an OEM, you can add the files <projectName>/systemmod/PostBuildVariantsLoadable.arxml and <projectName>/systemmod/PostBuildVariantsSelectable.arxml to the file list. Therefore you do not lose the information of the EB package which was edited with **Edit PostBuildVariants** wizards.

7.7.1. The Predefined Variants page

The wizard shows all variants (loadable or selectable) that are available in the system model. The first column displays the path to the PredefinedVariant in the model. All other columns represent one criterion.

NOTE

Variant configuration matrix

You can see all variant configurations at a glance in this matrix.



PostBuildVariants				
Variants path	Headlight	Sunroof	Turbocharger	
/EB/PostBuildSelectable/Basic	2:Eco	Not referred	11:No	
/EB/PostBuildSelectable/Economy	3:EcoPlus	5:Manual	11:No	
/Variants/Sportive	Not referred	Not referred	12:Yes	

Figure 7.48. The **Predefined Variants** page

You can perform the following tasks on this page:

- ▶ Create a new variant

You can create a variant . You can also choose the name for the variant. The name must be a valid short name and must be unique. Its length must be below 255 and it must match the format ^[a-zA-Z][a-zA-Z_0-9]*.

- ▶ Delete variant(s)

You can delete one or more variants which were created using this wizard.

- ▶ Rename a variant



You can rename a variant which was created using this wizard.

► Copy variant info to clipboard

This action copies the data of the currently selected variant to the clipboard. It contains the path of the variant and all referred criterions including the selected value.

► Refer criterions to variants

You can either select a value from the drop-down list box or insert an integer value directly.

The drop-down list box always contains the item **Not referred** to provide the possibility to unselect a criterion. The drop-down list box also contains all values from the CompuMethod associated to the criterion. If a value has a textual representations this is also shown in addition prefixed by a `:`. This makes it easier for the integrator to select the correct value.

You can configure the textual representations on the **Variant Criterions** page.

7.7.2. The Variant Criterions page

On the **Variant Criterions** page on the left side you see the list of available criterions, which show up in the **Predefined Variants** page as columns. For each criterion you can configure the textual representations in the table on the right side of the wizard.

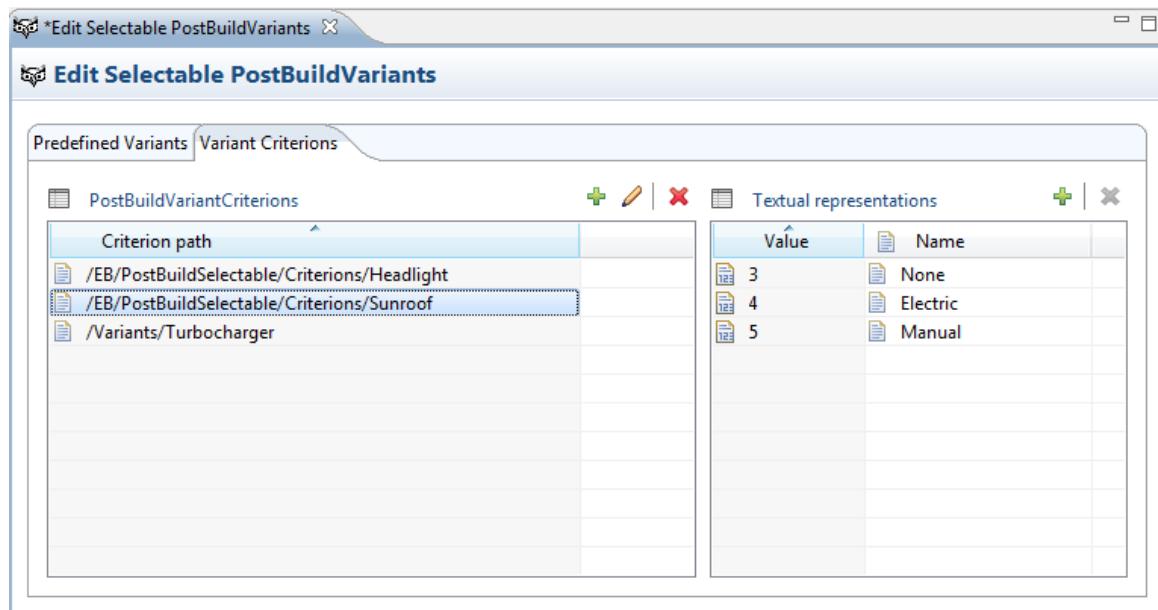


Figure 7.49. The **Variant Criterions** page

You can perform the following tasks on this page:



► Create a criterion

You can create a criterion . You can also choose the name for the criterion. The name must be a valid short name and unique. Its length must be below 255 and it must match the format ^ [a-zA-Z] [a-zA-Z_0-9]*.

► Delete criterion(s)

You can delete one or more criterions which were created with this wizard. If the selected criterion that shall be deleted is referred by a variant, then a confirmation dialog is displayed that asks you if you really want to delete the criterion as this causes that also its references are deleted.

► Rename a criterion

You can rename a criterion which was created using this wizard.

► Create textual representation

You can add a new row to the textual representation table. It is also possible to just insert integer values to the table without specifying a name. This may help to define a valid range for a criterion.

The values and names are shown when you select the value of a criterion for a variant on the **Predefined Variants** page.

► Delete textual representation(s)

You can delete one or more textual representations which were created with this wizard.

7.8. Using the Variant Handling wizard

The Variant Handling wizard deals with the limitation that the system model of EB tresos Studio and EB tresos AutoCore Generic 8 RTE contains only one variant (contrary to the ECU configuration) and eases the transfer of multiple variants into the ECU configuration.

Before you can run the Variant Handling wizard, you must configure it.



Configuring the Variant Handling wizard

Prerequisite:

- The project has variants configured.

**Step 1**

Open Project > Unattended Wizards > Unattended wizard configuration.

Step 2

Click on **Variant Handling wizard**.

Step 3

For each configured variant, the wizard displays a drop-down menu. Select a unique system description importer configuration for each variant.

Step 4

In **Artificial Master Variant**, select the corresponding system description importer configuration. For information on the artificial master variant, see [Section 6.13.4.2, “Working with post-build selectable variants”](#).

Step 5

In **ImportEcuConfig**, select your ECU configuration importer.

7.9. Using the Export/Import Split Configuration unattended wizards

7.9.1. Overview

This section describes how to use the **Export/Import Split Configuration** unattended wizards.

These wizards allow you to export and import *.arxml and *.ecuconfig files from and to an EB tresos Studio project in the AUTOSAR XML format.

- ▶ [Section 7.9.2, “Background information”](#) explains the concepts of the **Export/Import Split Configuration** unattended wizards.
- ▶ [Section 7.9.3, “Configuring the Export Split Configuration wizard”](#) provides step-by-step instructions on how to export a project configuration into split configuration files.
- ▶ [Section 7.9.4, “Configuring the Import Split Configuration wizard”](#) provides step-by-step instructions on how to import split configuration files into your project.

For more information about unattended wizards in general, see [Section 6.8.10, “Autoconfiguring routine jobs”](#).

7.9.2. Background information

This section explains the concept behind the **Export/Import Split Configuration** unattended wizards.



According to **AUTOSAR**, model data can be split into different ARXML files. The **Export/Import Split Configuration unattended wizards** enable you to export and import configuration data to and from such split ARML files.

7.9.2.1. Splittags

This section describes the SPLITTAG custom attribute.

The custom attribute SPLITTAG marks subtrees (i.e. containers) or single nodes (i.e. parameters) of the ECU configuration data to be part of a specific split part of one module configuration.

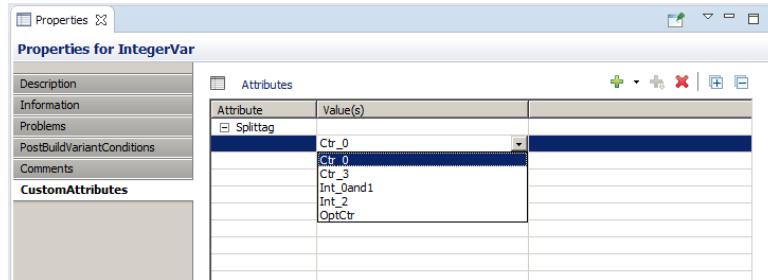


Figure 7.50. Splittag drop-down list box

EB tresos Studio allows you to select the SPLITTAG from a drop-down list box that lists all splittags used in the current project.

**NOTE****SPLITTAG invalid characters**

Do not use any of the following special characters in the SPLITTAG name:

/ \ | " < > ? : *

An information dialog is shown each time if you use any of these special characters.

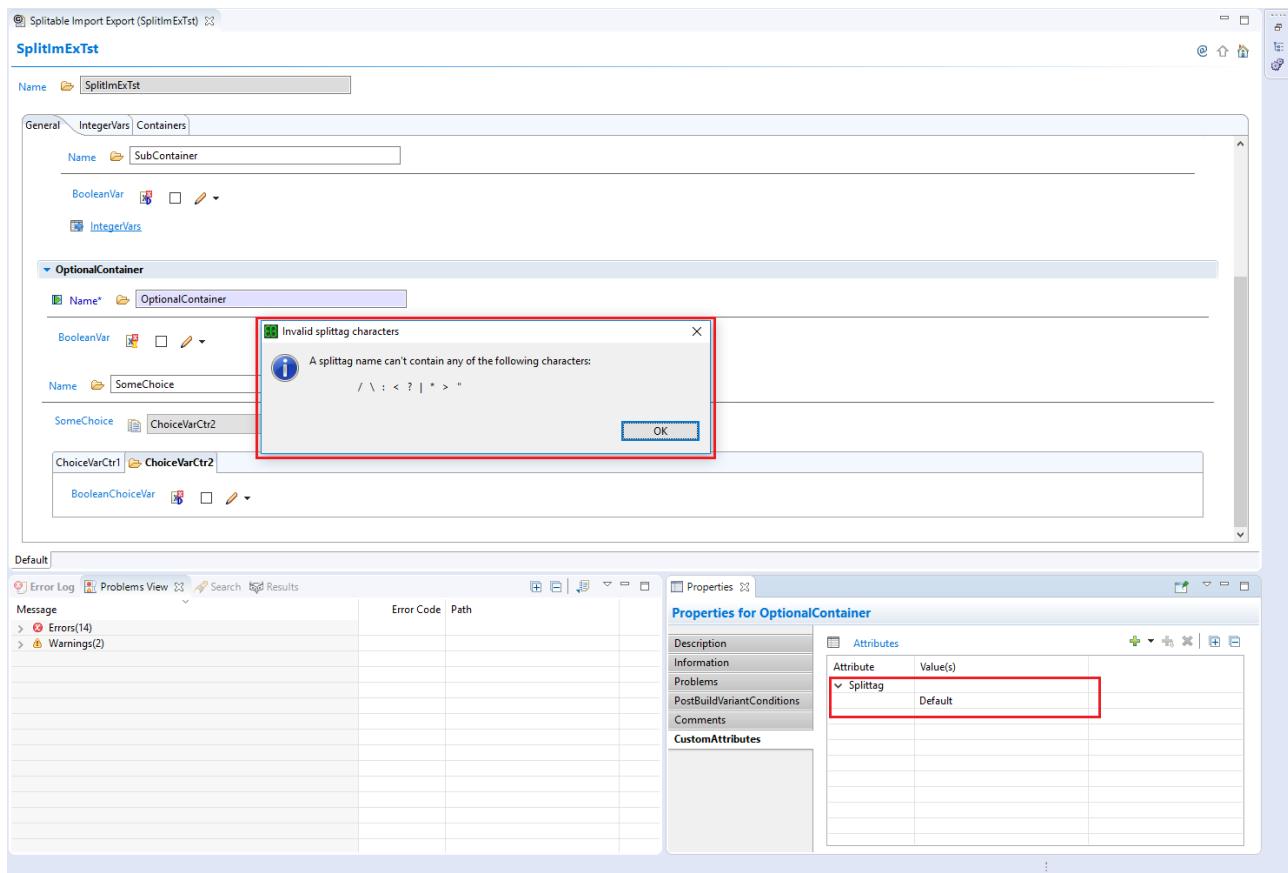


Figure 7.51. SPLITTAG invalid characters dialog

7.9.2.2. Export split arxml files

The **Export Split Configuration** wizard is used to export the configuration from the currently selected EB tresos Studio project to the corresponding split *.arxml files. It works as follows:

- ▶ Based on the **Splittag** custom attribute value of each node, the wizard exports the nodes to the corresponding file using the naming pattern <Module Configuration Name>.<SPLITTAG>.arxml. See [Section 5.3.9.6, “CustomAttributes tab”](#) for how to edit the custom attributes of a node.
- ▶ Nodes that have no **Splittag** custom attribute set are exported to a file named <Module Configuration Name>.Rest.arxml.



- ▶ Nodes with calculated values are put into a separate file named <Module Configuration Name>_STUDIO_CALCULATED.arxml, unless they have a **Splittag** custom attribute explicitly set.
- ▶ In the export, the provided AUTOSAR version is considered as the AUTOSAR version of the export configuration.

This way, the exporter wizard writes *at least* one file per module.

7.9.2.3. Import split arxml files

The **Import Split Configuration** wizard can import *.arxml or *.ecuconfig split files into the currently selected project in EB tresos Studio. It works as follows:

- ▶ The **Import Split Configuration** wizard merges or replaces the existing project configuration with the related configuration from the input files.
- ▶ You must ensure that the input files match the pattern <Module Configuration Name>.<SPLITTAG>.<extension> so that the nodes get the **Splittag** custom attribute set to <SPLIT-TAG>.
- ▶ You can either **INCLUDE** or **EXCLUDE** the input files that match the pattern.
- ▶ To be able to afterwards export the configuration into files with the same names, the importer sets a **Splittag** custom attribute in each node. This custom attribute defines from which file the node comes.
- ▶ During import, automatic path mapping is applied. This adjusts e.g. the package name of the input module configuration to match the convention that is used in EB tresos Studio reference paths to the module definition. For details, see [Section 6.10.2.2.1, “Path mapping in import mode”](#).

7.9.3. Configuring the Export Split Configuration wizard

This chapter describes how to configure the **Export Split Configuration** wizard to export the configuration from the currently selected EB tresos Studio project to the corresponding split *.arxml files.

7.9.3.1. Opening the Export Split Configuration wizard

1. In the **Project Explorer** view, select a loaded ECU configuration project.
2. On the tool bar, click **Unattended wizard configuration**.

The **Unattended Wizard** opens up.

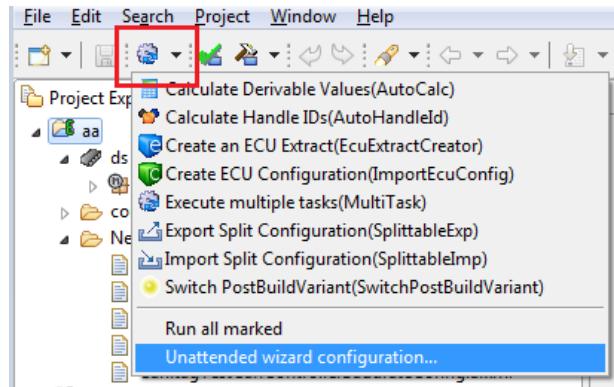


Figure 7.52. Opening the Unattended wizard configuration

3. Select **Export Split Configuration** wizard.

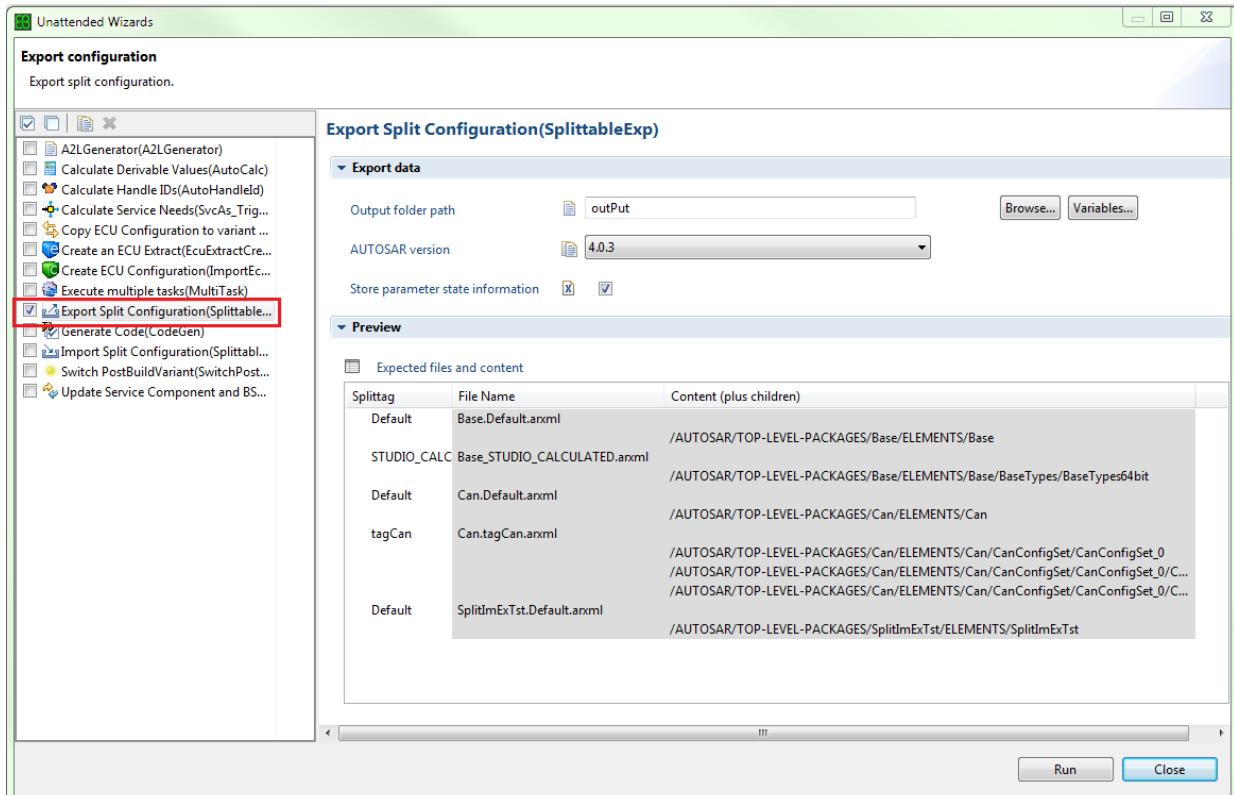


Figure 7.53. Opening the Export Split Configuration wizard

7.9.3.2. Output folder path

- In the **Output folder path** text field, specify a valid output folder path. See [Figure 7.53, “Opening the Export Split Configuration wizard” Output folder path](#).



2. You can specify either an absolute path or a relative project path.

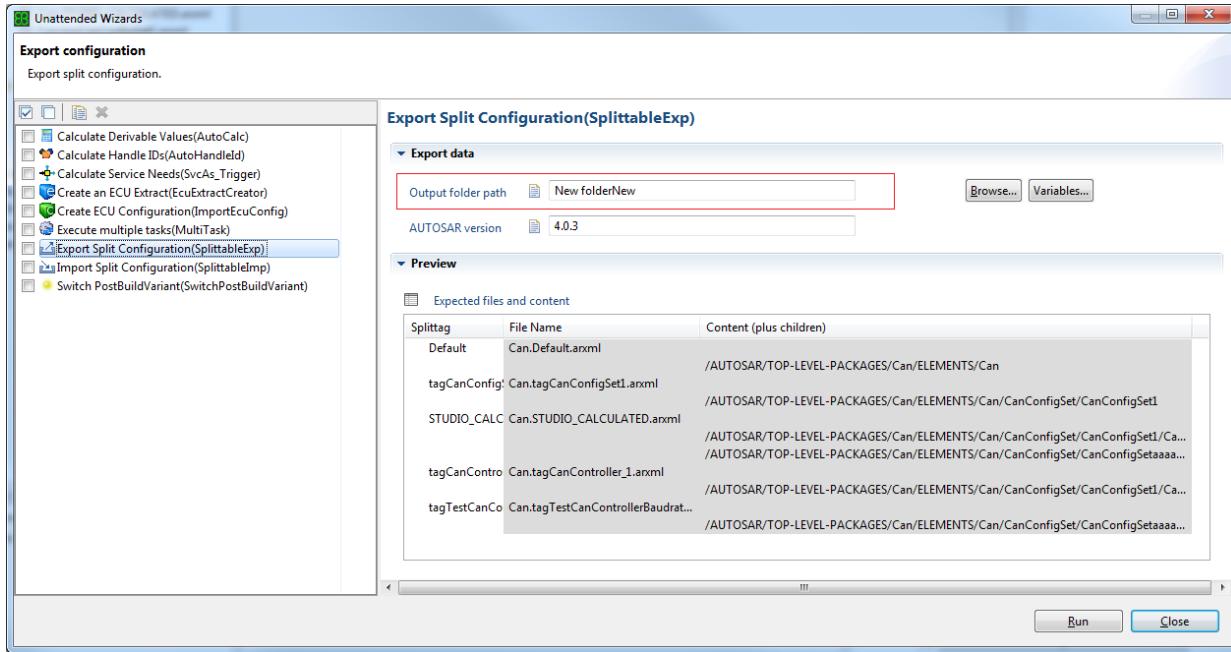


Figure 7.54. Output folder path

7.9.3.3. AUTOSAR version

You can select the AUTOSAR release version from the **AUTOSAR version** drop-down list box. For the supported output formats, see [Section 3.5.3.2, “Content types for AUTOSAR module configuration files”](#).

The AUTOSAR version selected defines the XML schema version to be used for the output files in the AUTOSAR XML format. Valid inputs follow the pattern <Major>. <Minor>. <Patch>, e.g. 4 . 0 . 3.

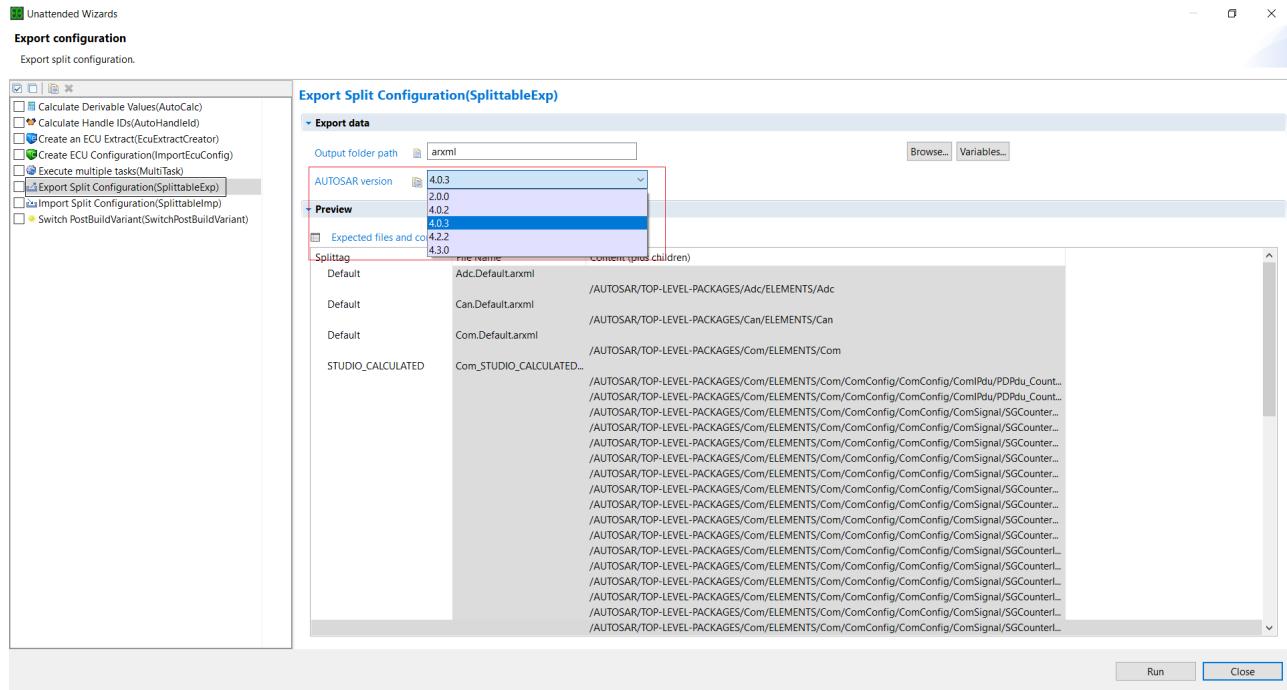


Figure 7.55. AUTOSAR version

7.9.3.4. Store parameter state information

You can use the checkbox **Store parameter state information** to decide if you want to store the parameter state information. By default, the checkbox is Checked.

- ▶ Checked: the exporter stores the parameter state information into the ADMIN-DATA tag of the generated output files.
 - ▶ Unchecked: the exporter ignores the parameter state information and does not store it into the ADMIN-DATA tag of the generated output files.

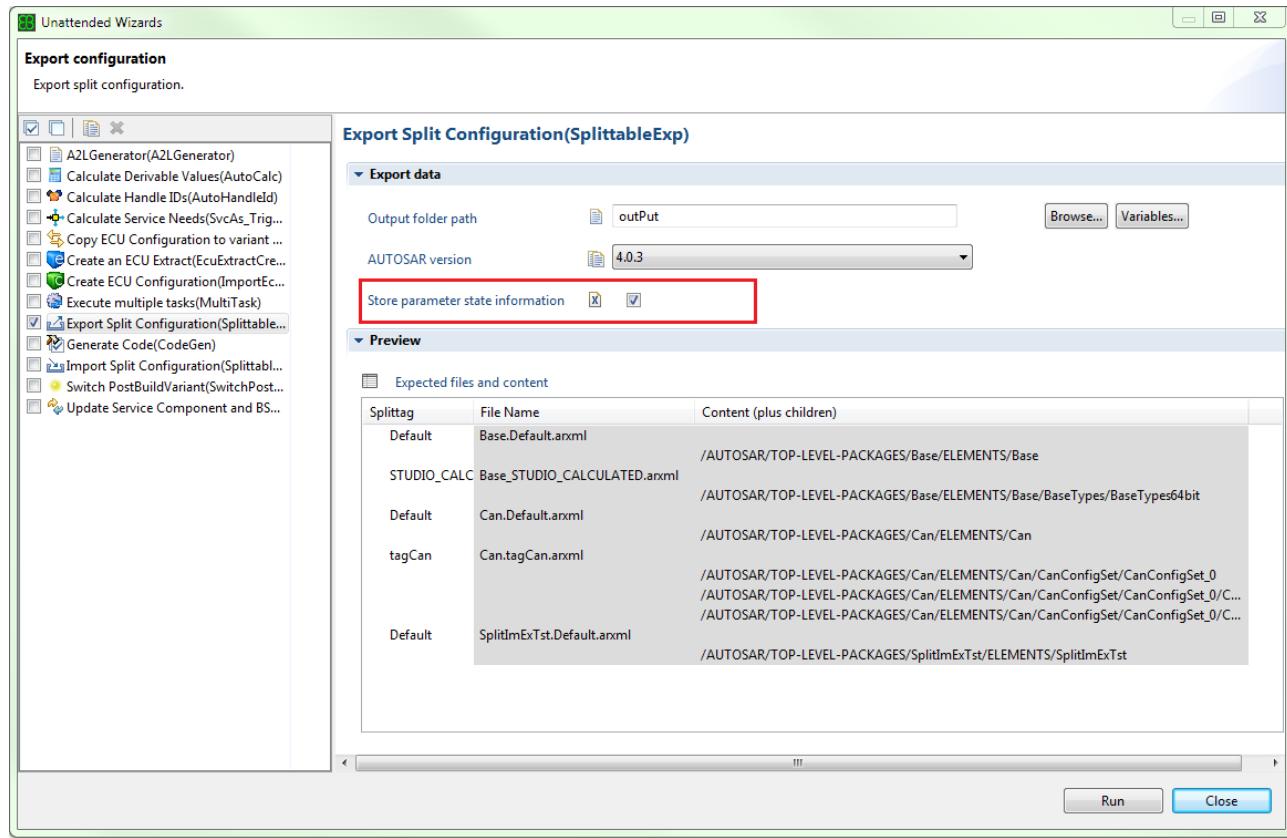


Figure 7.56. Store parameter state information

7.9.3.5. Export data filter

Export data filter options enable you to decide what kind of parameters should be exported to the respective output files. By default, the **Manually Edited** checkbox is Checked.

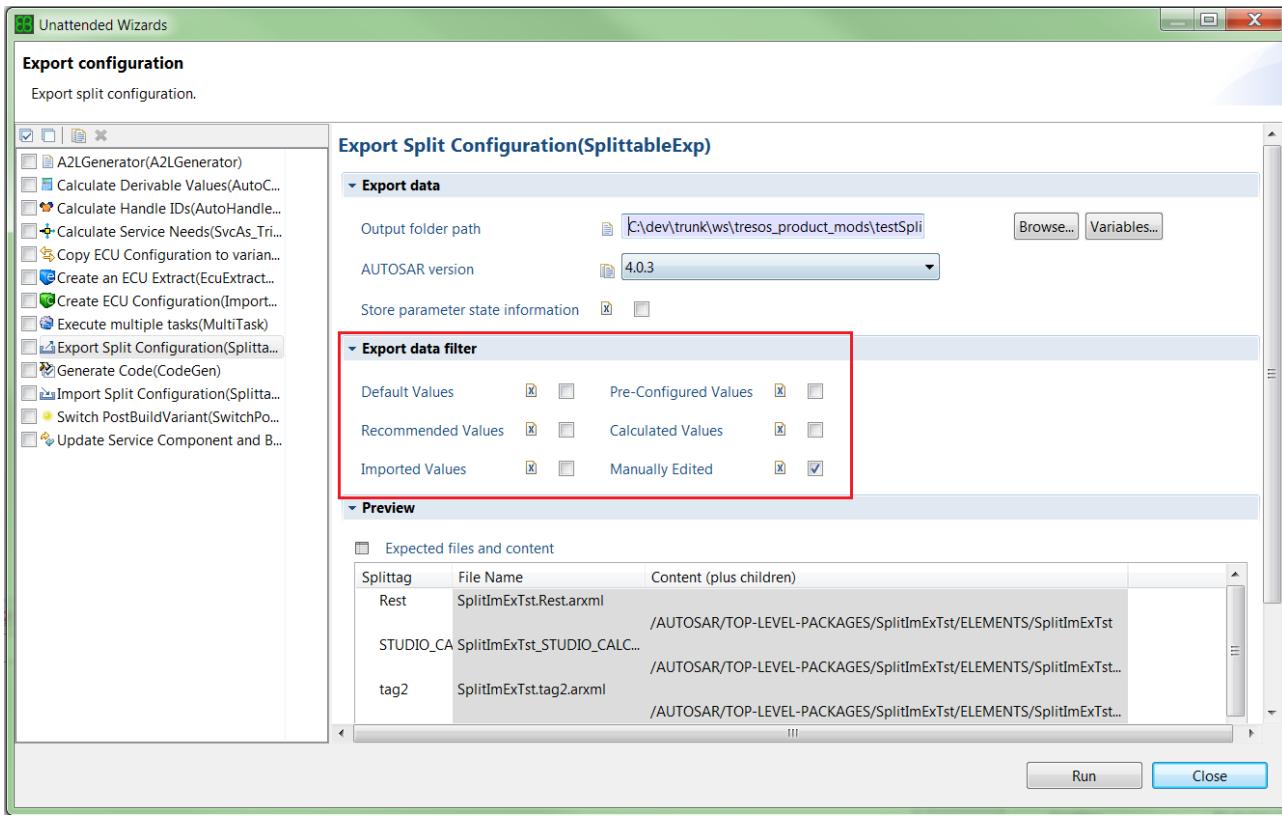


Figure 7.57. Export data filter

Field/button name	Instructions: Select the check box
Default Values	To export all elements that are set to their default value.
Pre-Configured Values	To export all elements that are set to preconfigured values.
Recommended Values	To export all elements that are set to recommended values.
Calculated Values	To export all elements that are calculated.
Imported Values	To export all elements that are imported.
Manually Edited	To export all elements that are manually edited.

7.9.3.6. Expected files and content

The **Preview** section provides an overview of the expected output. It shows the export file names and which parts of the configuration are contained in them as XPath terms. It also gives an overview over all **Splittag** custom attribute values that are used in your project.

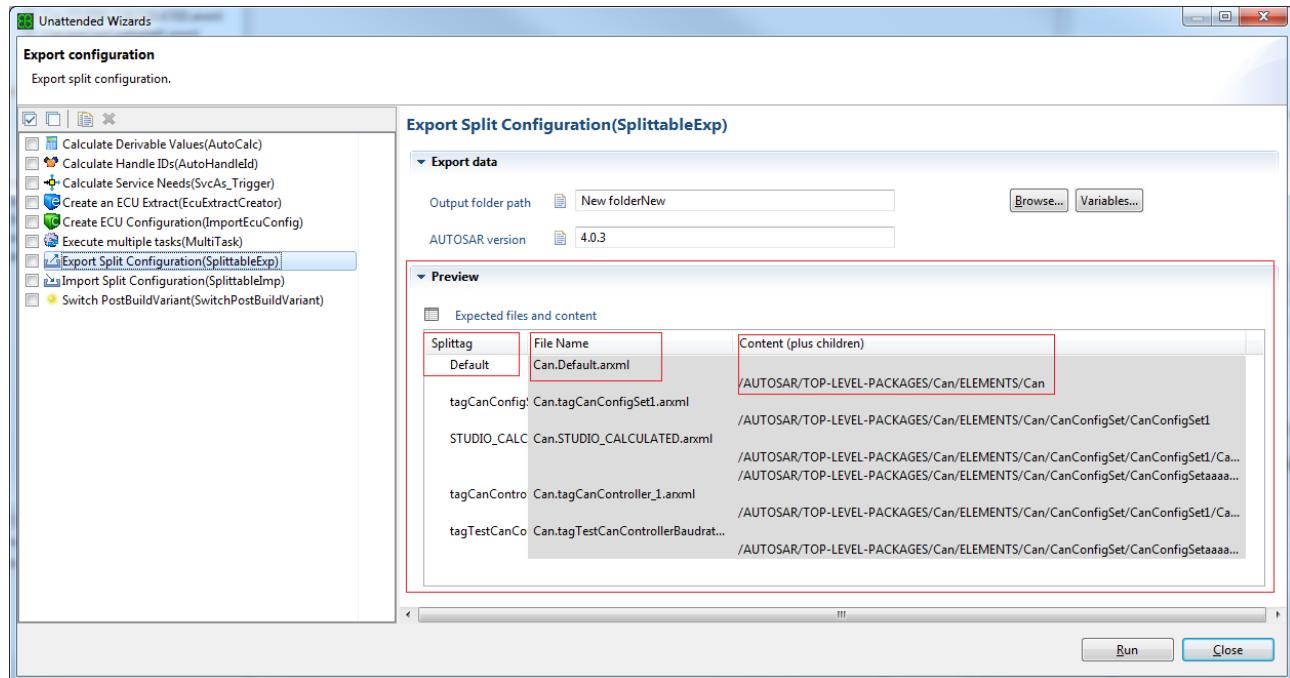


Figure 7.58. Expected files and content table

7.9.4. Configuring the Import Split Configuration wizard

This chapter describes how to configure the **Import Split Configuration** wizard to import *.arxml or *.-ecuconfig split files into the currently selected EB tresos Studio project.

7.9.4.1. Opening the Import Split Configuration wizard

1. Select a loaded ECU configuration project in the **Project Explorer** view.
2. Make sure, that the project contains the modules that you want to import.
3. On the tool bar, click **Unattended wizard configuration...** to open the **Unattended Wizards** dialog.

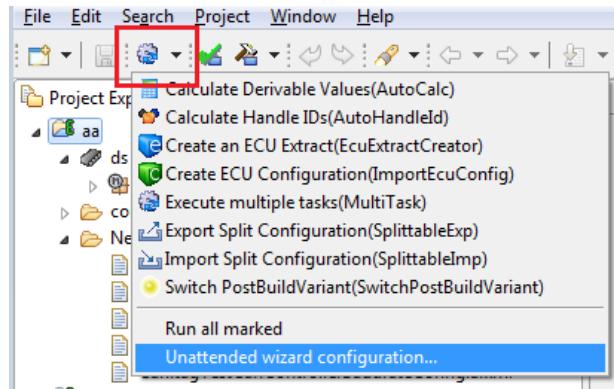


Figure 7.59. Opening the Unattended wizard configuration

4. Select the Import Split Configuration wizard.

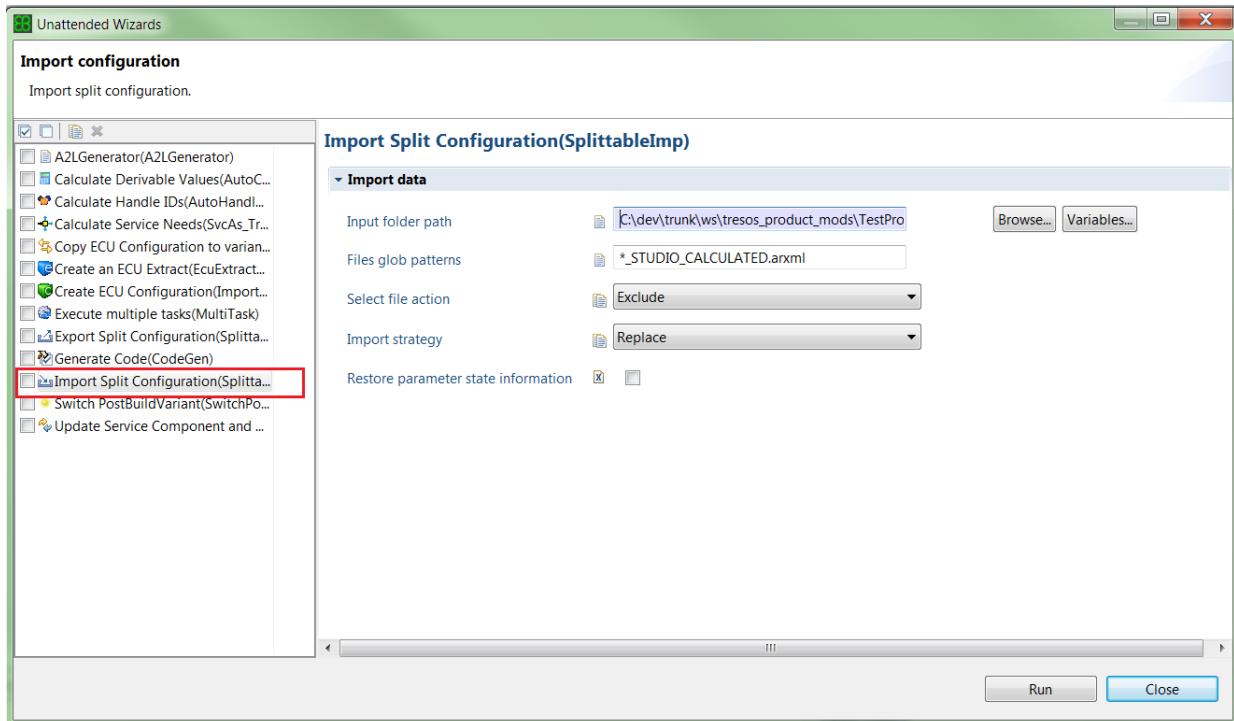


Figure 7.60. Opening the Import Split Configuration wizard

7.9.4.2. Input folder path

A valid input folder path must be given in the **Input folder path** text field of the wizard (see [Figure 7.60, “Opening the Import Split Configuration wizard”](#) Input folder path). The path can be either an absolute or a relative project path. When opening the wizard configuration for the first time, the default value of the input folder path is the project path. Any files named `*.arxml` or `*.ecuconfig` are imported from the given folder.

**NOTE****Other modules in input files**

If the input files contain configuration data for modules that are not part of the project, the import will lead to warning messages.

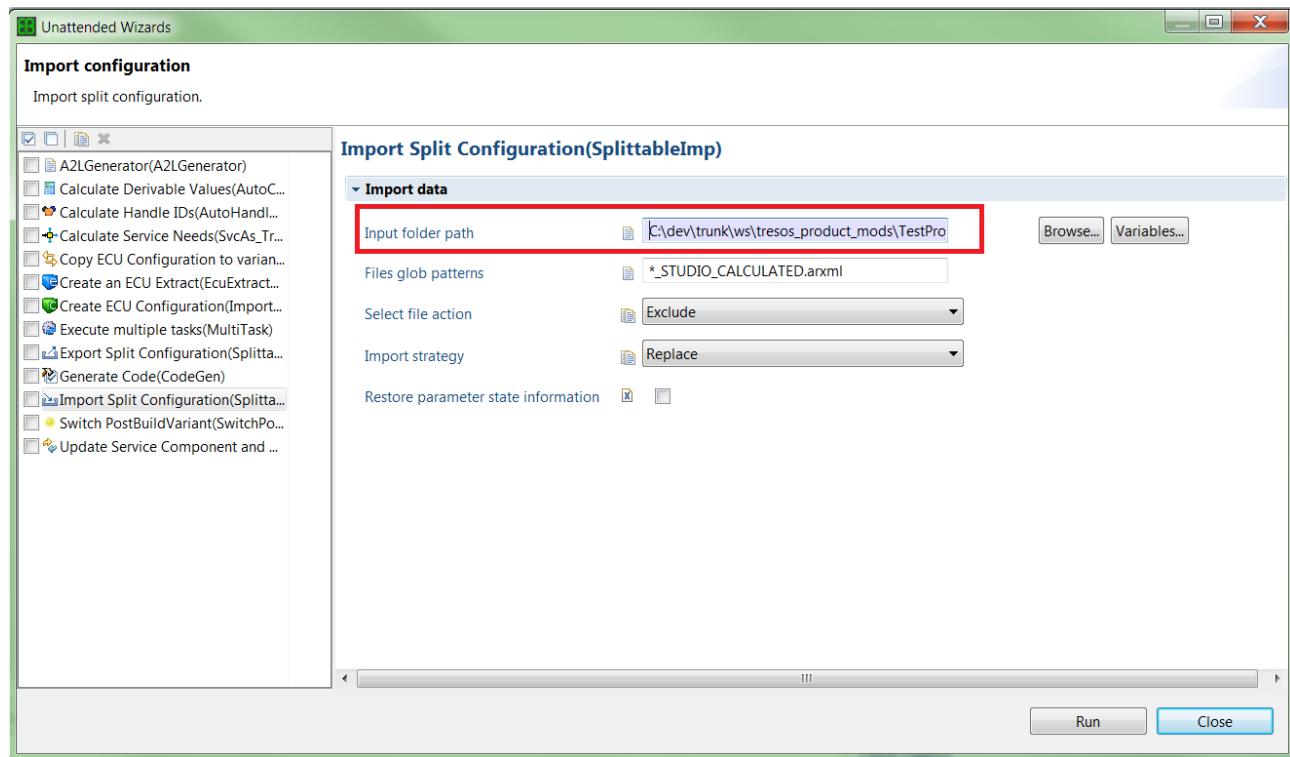


Figure 7.61. Input folder path

7.9.4.3. Files glob patterns

To exclude or include files from import, use the **Files glob patterns** field to define glob patterns. Separate them by comma, if needed.

Any configurations contained in files that match the pattern are either included or excluded based on the value of **Select file action**.

The default value is ***.STUDIO_CALCULATED.arxml**. This covers files with automatically calculated values, which stem from the exporter wizard counterpart.

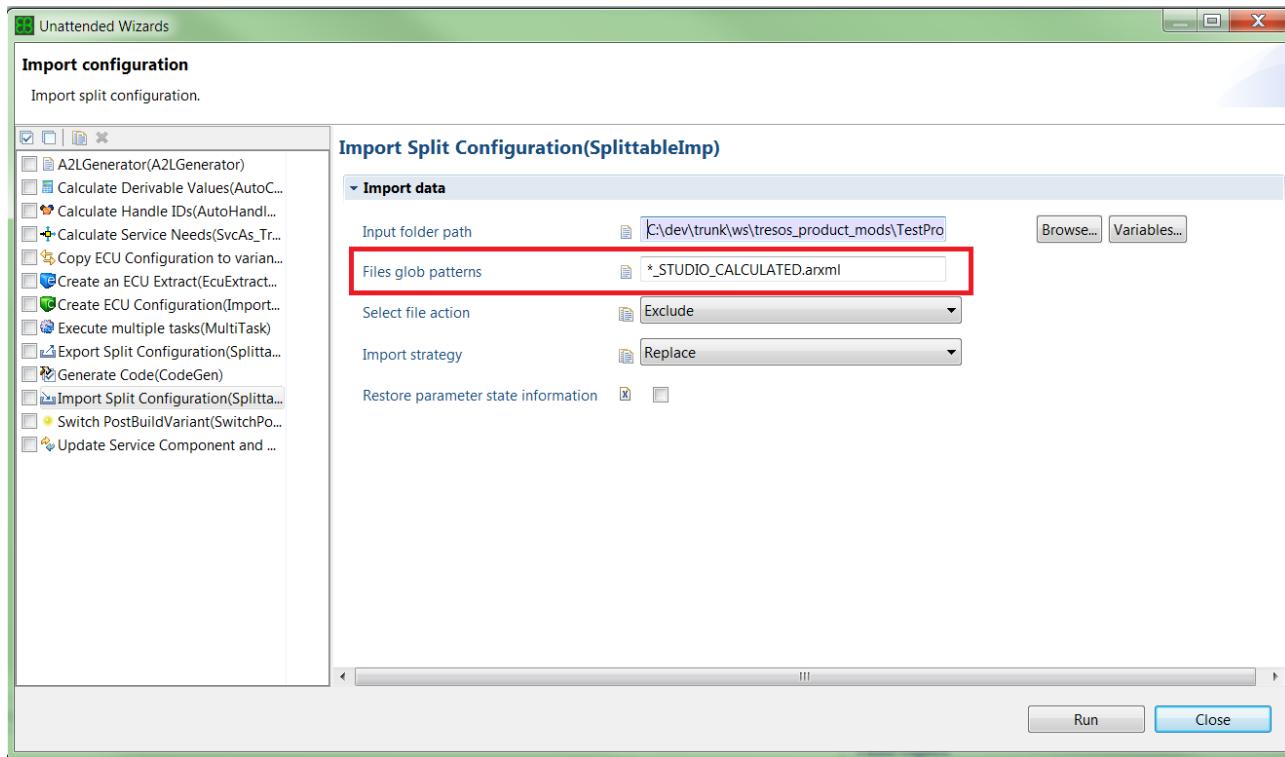


Figure 7.62. Files glob patterns

7.9.4.4. Select file action

The drop-down list box **Select file action** offers two options: **Exclude** or **Include**. By default, it is set to **Exclude**.

- ▶ **Exclude:** If this option is selected, the importer excludes the files that match glob patterns in the given **Input folder path**.
- ▶ **Include:** If this option is selected, the importer includes the files that match glob patterns in the given **Input folder path**.

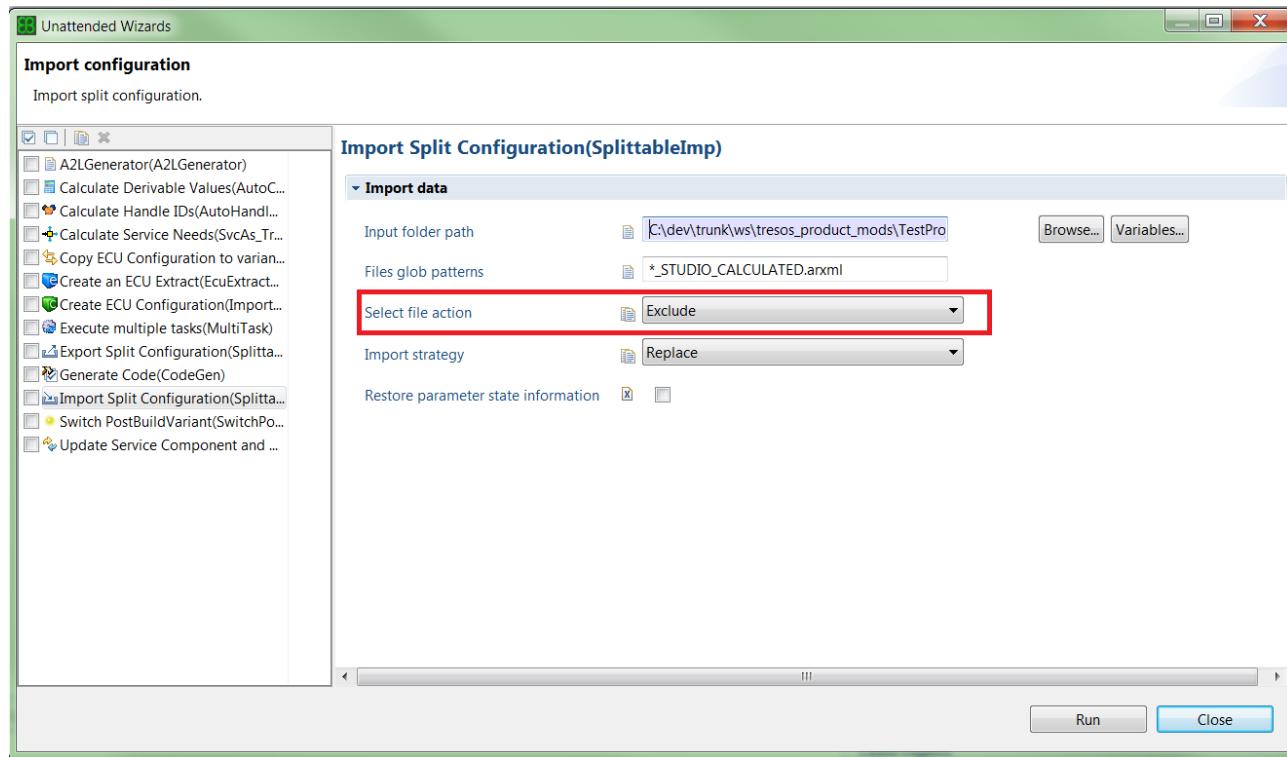


Figure 7.63. Select file action

7.9.4.5. Import strategy

The drop-down list box **Import strategy** offers two options: Replace or Merge. By default, it is set to Replace.

- ▶ Replace: If this option is selected, the importer replaces an existing module configuration completely with the given input data for the corresponding module.
- ▶ Merge: If this option is selected, the importer merges the given input data into an existing module configuration. See [Section 6.10.2.2.3, “Merge”](#) for more details about the merge strategy.

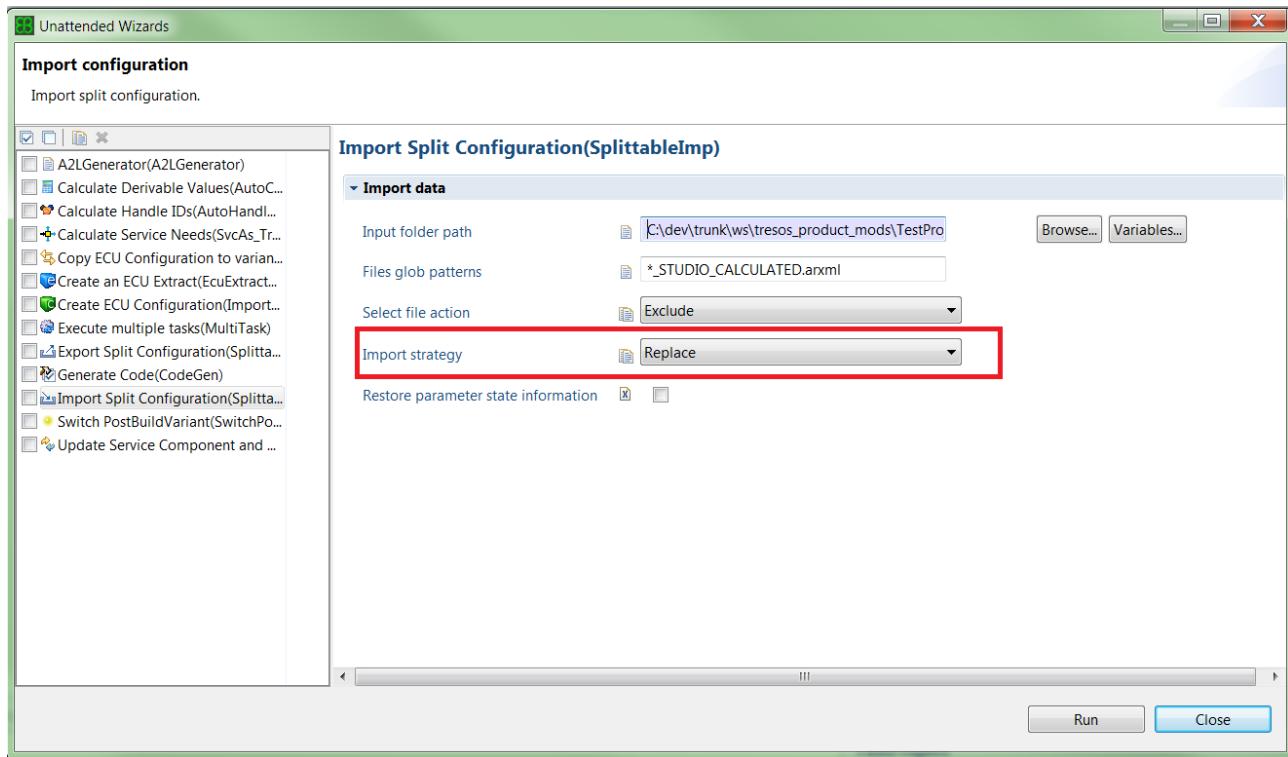


Figure 7.64. Import strategy

7.9.4.6. Restore parameter state information

You can use the checkbox **Restore parameter state information** to decide if you want to restore the parameter state information.

- ▶ By default, the checkbox is enabled if **Import strategy** is set to Replace.
- ▶ The checkbox is disabled if **Import strategy** is set to Merge.
- ▶ Checked: The importer restores the parameter state information from the ADMIN-DATA tag of the imported input files to the corresponding XDM file.
- ▶ Unchecked: The importer ignores the parameter state information to be restored to the corresponding XDM file.

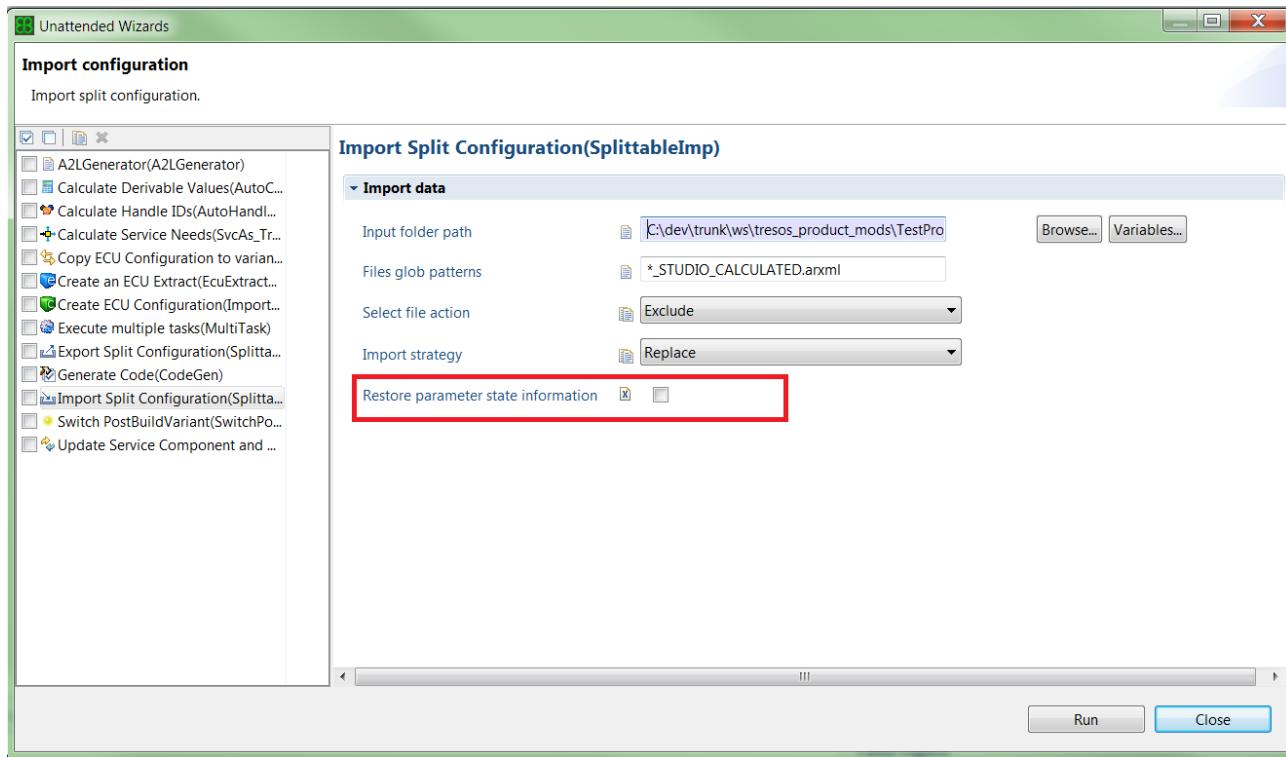


Figure 7.65. Restore parameter state information

7.10. Using the Transform XML files using XSLT wizard

7.10.1. Overview

This section explains how to use the **Transform XML files using XSLT** unattended wizard. See [Section 6.8.10, “Autoconfiguring routine jobs”](#) for more information about unattended wizards in general.

The **Transform XML files using XSLT** wizard allows you to transform single or multiple XML files to the required format by applying Extensible Stylesheet Language Transformation (XSLT). For details about the supported XSLT version 2.0, see <https://www.w3.org/TR/xslt20/>.

The content of the XML files used as input is not changed by the XSL transformation. Instead the wizard creates new files in a given output folder.



7.10.2. Configuring the Transform XML files using XSLT wizard

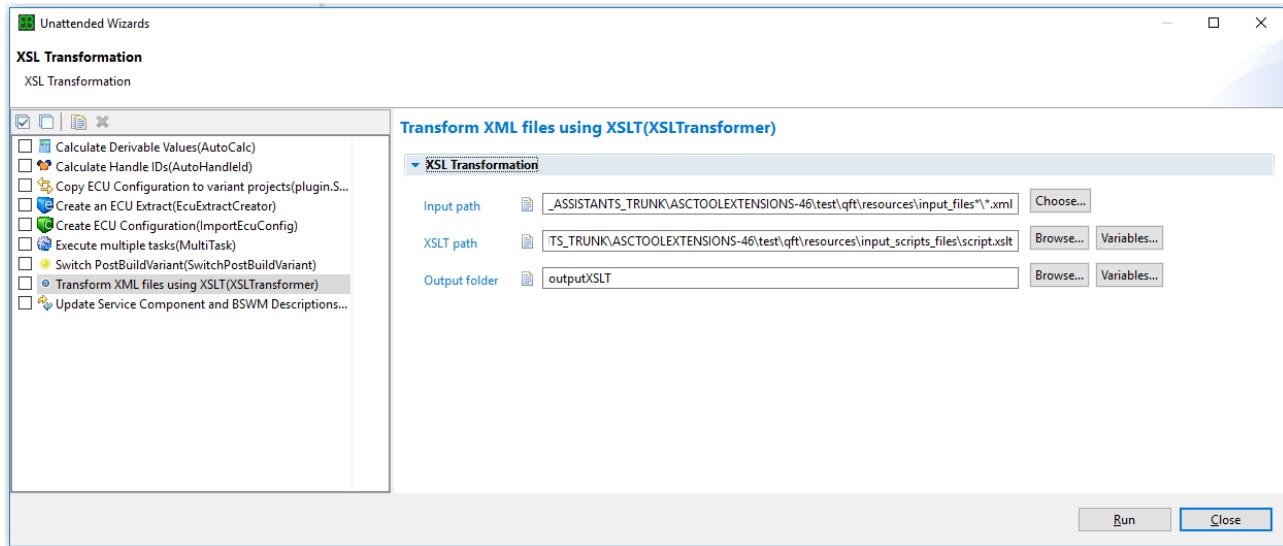


Figure 7.66. The **Transform XML files using XSLT** wizard

Label	Description
Input path	A valid input path must be given in the Input path text field of the wizard. You can use glob wildcards to select multiple XML input files.
	<p>NOTE Glob wildcard patterns  For further information about glob wildcard patterns, see http://docs.oracle.com/javase/tutorial/essential/io/fileOps-.html#glob</p>
XSLT path	A valid XSLT file path must be given in the XSLT path text field of the wizard. Supported file extensions are .xslt and .xsl .
Output folder	A valid output folder path must be given in the Output folder path text field. When you open the wizard configuration for the first time, the default value of the output folder is outputXSLT .

Table 7.1. Configuration parameters of the **Transform XML files using XSLT** wizard

**NOTE****Relative paths**

Relative paths are always resolved to the project location.

NOTE**Path variables**

You can use path variables in those text fields. For more information about path variables, see [Section 6.10.2.2.3, “Using path variables”](#).

7.10.3. Running the Transform XML files using XSLT wizard

To transform the XML input files using XSLT, take the following steps:

- ▶ Follow the instruction in [Section 6.8.10.1, “Configuring unattended wizards”](#) to open the **Unattended Wizards** configuration dialog.
- ▶ Select **Transform XML files using XSLT** in the dialog.
- ▶ Configure the wizard according to [Table 7.1, “Configuration parameters of the Transform XML files using XSLT wizard”](#).
- ▶ Click the **Choose...** button to open the dialog which supports you to add variables to the **Input path** and provides a preview of the matching files.

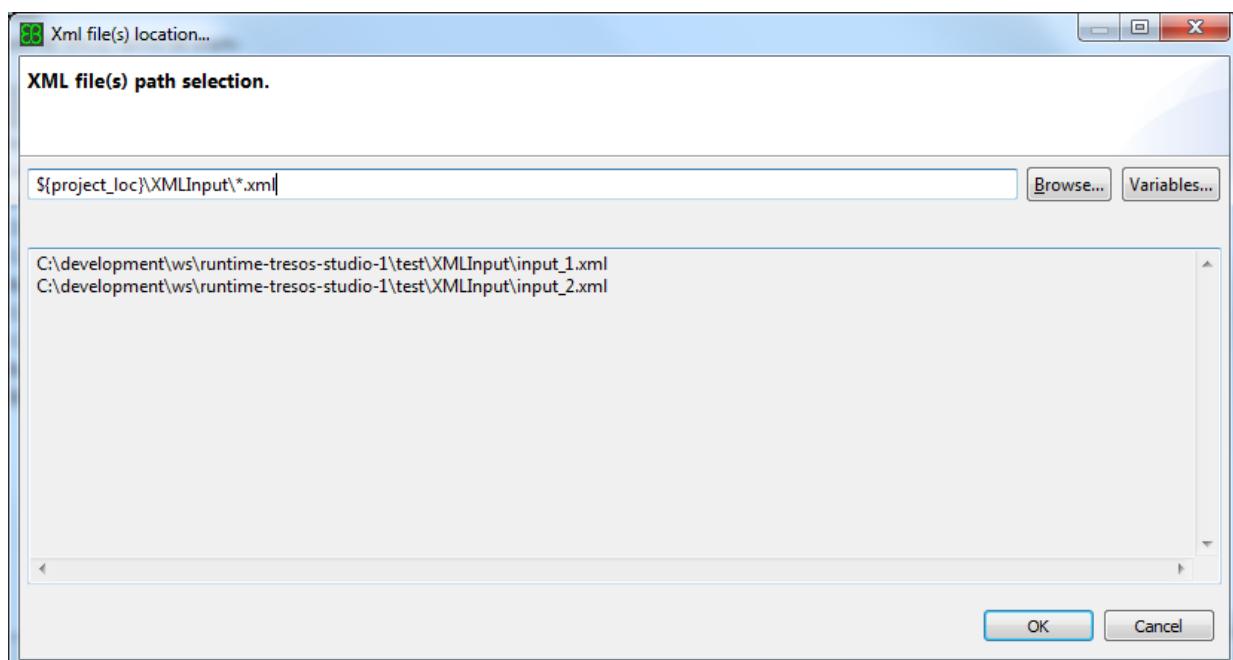


Figure 7.67. The input file selection dialog supporting variables and glob wildcards



- ▶ Click the **Run** button to start the XSL transformation.
- ▶ After the transformation is finished you find the recently created XML files in the configured **Output folder**.

NOTE**Invalid input files**

If an input file contains invalid XML content the wizard skip the file and continue the transformation for the next file. No output is generated for invalid input files.

NOTE**Unique output file names**

The wizard creates unique output file names, even if some input files have the same name.

7.11. Using the Service Needs Calculator

7.11.1. Overview

This section describes the Service Needs Calculator unattended wizard. It is intended for readers who have good knowledge of AUTOSAR.

- ▶ [Section 7.11.2, “Background information”](#) provides an overview and explains the concepts of the Service Needs Calculator (SvCAs).
- ▶ [Section 7.11.3, “Configuring the Service Needs Calculator”](#) provides configuration information.
- ▶ [Section 7.11.4, “Running the Service Needs Calculator”](#) provides information on running the Service Needs Calculator.

7.11.2. Background information

This section provides an overview of the Service Needs Calculator and explains underlying concepts.

- ▶ [Section 7.11.2.1, “The concept of service needs”](#) describes the service needs concept.
- ▶ [Section 7.11.2.2, “Functional distribution”](#) provides an overview of the distribution of the service needs calculation functionality.
- ▶ [Section 7.11.2.3, “Service needs calculation life cycle”](#) provides an overview of the life cycle calculation of the Service Needs Calculator.



- ▶ [Section 7.11.2.4, “Limitations”](#) provides an overview of the Service Needs Calculator limitations.
- ▶ [Section 7.11.2.5, “Relation to the importer info value”](#) explains the relationship between the Service Needs Calculator and the importer info value.
- ▶ [Section 7.11.2.6, “Deleting elements”](#) explains how the Service Needs Calculator proceeds when deleting elements.

7.11.2.1. The concept of service needs

In AUTOSAR, a service is a logical entity that offers general functionality to software components and BSW modules. For example, the `NvM` offers services to read and write data blocks to the memory and is regarded as a service provider. In order to use such a service, a software component or BSW module may have a configuration dependency on the service provider.

For example, to use the `NvM` services, memory blocks must be configured in the `NvM`. References to these blocks must be configured in the component itself or in the BSW module that wants to use the `NvM` services. This kind of module dependency is known in AUTOSAR as service needs.

In EB tresos AutoCore, the concept of service needs is extended to include further dependencies that do not necessarily have a direct reference in the service requester.

For example, BSW modules init functions must be configured in the `EcuM` module. But the modules themselves do not need direct `EcuM` references for this. This kind of dependency is also treated as service needs between the modules and the `EcuM`.

Service needs can be seen as a contract between modules that provide services and modules that request services. The Service Needs Calculator coordinates and provides the life cycle and graphical user interface. The contract between service providers and service requesters is communicated by exchanging XML fragments that describe the requested service needs.

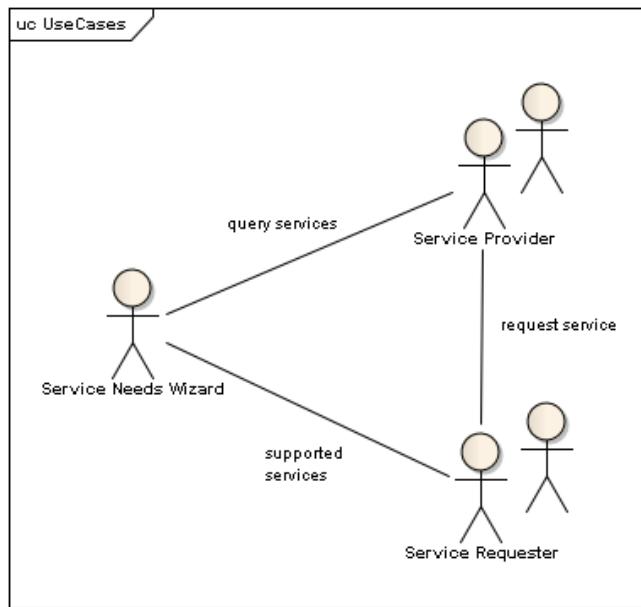


Figure 7.68. Relation between the involved parties of the Service Needs Calculator

The Service Needs Calculator is an unattended wizard which organizes the configuration of parameters that one BSW module needs in the configuration of another BSW module. These parameters configure a service which a module can offer to other modules. This is like calling a main function.

The Service Needs Calculator automatically collects the requests from the service requester and performs the necessary configuration changes in the service provider.

The Service Needs Calculator is supplied with the EB tresos AutoCore and provides support for the automatic configuration of the following service needs:

- ▶ EcuM init functions
- ▶ Dem events
- ▶ NvM blocks
- ▶ Os Tasks
- ▶ Os ISRs
- ▶ Os resources
- ▶ Os schedule tables
- ▶ Os events
- ▶ Os IOC channels
- ▶ Os spinlock
- ▶ Os alarms
- ▶ Com signals



- ▶ Com signal groups
- ▶ LdCom
- ▶ Xfrm Transformers (ComXf, E2EXf, SomelpXf)

In addition, the Service Needs Calculator provides the following features:

- ▶ If the Service Needs Calculator sets a parameter or reference automatically that you manually modify later, the Service Needs Calculator will not overwrite your manual changes if you run it again.
- ▶ The Service Needs Calculator removes containers, parameters, and references from a previous run if they are not needed anymore in the following run. This happens only if they were not modified manually.
- ▶ Several modules which require a Dem event or an NvM block have a reference to the Dem event/NvM block in their own configuration. The Service Needs Calculator sets this reference automatically.

NOTE
The Service Needs Calculator only works on request


The Service Needs Calculator updates information only on request. If it does not receive a specific request, it does not execute. The absence of a request usually means that the configuration is not complete. If the Service Needs Calculator does not create or update anything, check your configuration and make sure there are no elements missing and the request is not null.

If the Service Needs Calculator did not execute anything, you cannot expand the displayed element to view the execution results. See the example for Os Spinlock in the following picture:

(SVCOS_20) Os Spinlocks

Figure 7.69. The result when nothing was executed.

If the Service Needs Calculator updated, created or ignored something, you can expand the displayed element. See the example for Os Tasks in the following picture:

(SVCOS_1) Os Tasks

Figure 7.70. The result when something was updated.

7.11.2.2. Functional distribution

The Service Needs Calculator functionality is distributed over several mostly independent parties:

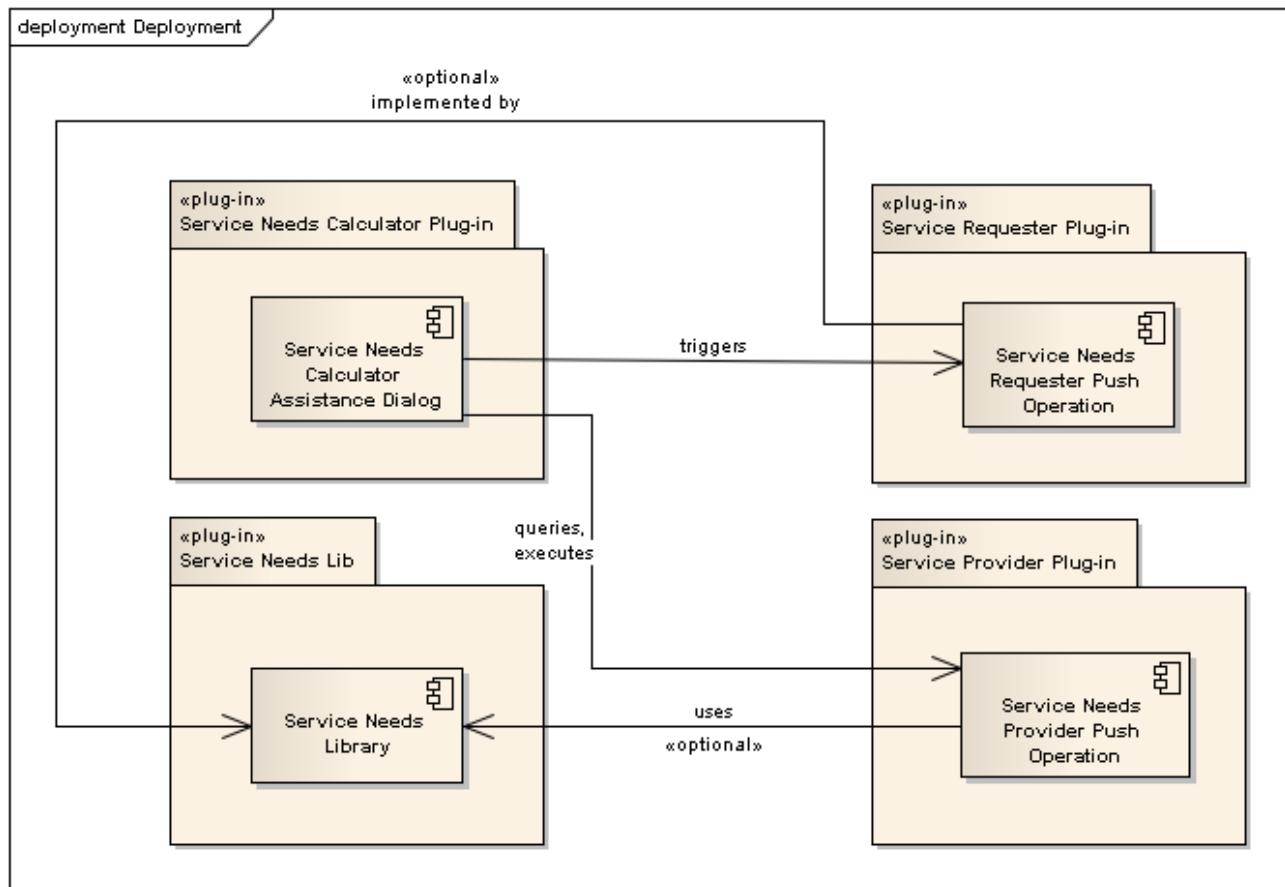


Figure 7.71. Components of the Service Needs Calculator

- ▶ **Service Needs Base Library (SvcAsLib):** hosts the basic functionality needed by any party involved in the Service Needs Calculator life cycle.
- ▶ **Service Needs Calculator (SvcAs):** hosts the configuration GUI and the wizard functionality. None of the other plug-ins shall depend on this plug-in.
- ▶ **Service Needs Providers (SvcAsProvider_TS_TxDxM...):** Hosts the service provider functionality, i.e. changes the configuration of the underlying BSW module according to the service requests. May contribute to the configuration GUI of the wizard to provide special settings.
- ▶ **Service Requesters:** BSW module plug-in that hosts the service requests.

7.11.2.3. Service needs calculation life cycle

The contract between the service requester and the service provider is communicated with a four-phase protocol controlled by the Service Needs Calculator:

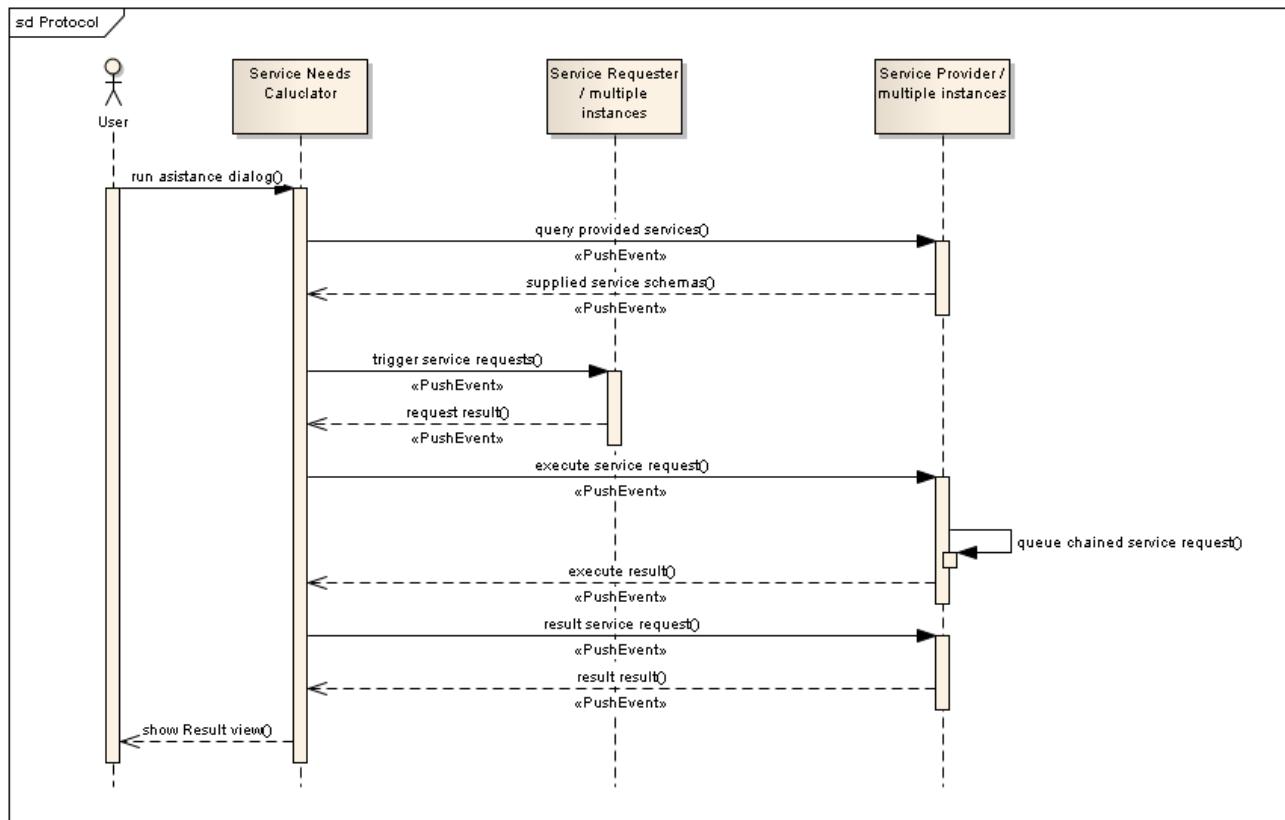


Figure 7.72. Service Needs calculation life cycle

1. Gather services

All service providers register their provided services at the push event. The services provided may be project-specific by using the condition filters in the push event registration. A provided service is added by setting a push event variable with the name `schema://<service-name>:<service-version>` to an instance of an XML schema validator. The push event is returned to the Service Needs Calculator.

2. Request phase

All service requesters register a `PushOperation` for the event using an event filter. This filter matches only the requested services with the schema URIs supported by the service providers. For each requested service, it creates an XML fragment that conforms to the schema of the service and sets the variable `req://<service-name>:<service-version>` to the XML.

3. Execution phase

The Service Needs Calculator sends the push event with all data gathered in the previous phases. For each requested service, the Service Needs Calculator adds the variable `state://<service-name>:<service-version>`. This contains a memento to which the service provider can add state information persistently.



The service providers execute the service requests. The service providers add any information needed by the service requesters to the XML sent by the service requesters by manipulating the original request in variable `req://<service-name>:<service-version>`.

4. Result phase

To return the results of executed services, the Service Needs Calculator sends a service result push event that contains all data gathered in the previous phases.

7.11.2.4. Limitations

The Service Needs Calculator has the following limitations:

- ▶ Not all init functions are automatically allocated

Description:

In AUTOSAR 4.0, only the modules `Det`, `Mcu`, `Port`, `Dio`, `Gpt`, `Wdg`, `WdgM`, `Adc`, `Icu`, and `Pwm` are initialized by the EcumFlex. All other modules are initialized by the `BswM`. These init functions cannot be configured by the Service Needs Calculator.

Rationale:

The `BswM` does not support a configurable list of init functions. The init functions have to be called in C-callout functions.

- ▶ The Service Needs Calculator does not configure the right size for `NvM` blocks

Description:

When `NvM` blocks are allocated, the Service Needs Calculator is not able to determine the correct block size in all cases. During compilation, the `NvM` checks if the configured size equals the required size by using the `sizeof` operator. If the values do not match, an error occurs. You must update the block size in the `NvM` configuration manually.

Rationale:

The block size depends on the configuration of the compiler.

7.11.2.5. Relation to the importer info value

The Service Needs Calculator can only update parameters that have the `importer info` set to `default` or `calculated`. You can find the importer info value in the `.xdm` file of your config folder or in the EB tresos Studio **Properties** view under **Information -> General -> Source** as shown in the following pictures.



```
<d:var name="ComSignalInitValue" type="STRING" value="0">
  <a:a name="ENABLE" value="TRUE"/>
  <a:a name="IMPORTER_INFO" value="@DEF"/>
</d:var>
```

Figure 7.73. Importer info value in the .xdm file

Properties for ComSignalInitValue		
	General	
Description	Name	ComSignalInitValue
Information	Type	String
Problems	Source	default value -
PostBuildVariantConditions	ORIGIN	AUTOSAR_ECUC
Comments	Multiplicity Config Class	PostBuild
CustomAttributes	Value Config Class	PostBuild
	UUID	ECUC:8fbc350f-1d96-8a4c-2ea7-fb39cbe6276
	Paths	
	XPath	/AUTOSAR/TOP-LEVEL-PACKAGES/Com/ELEMENTS/Com/ComConfig/C
	Parameter Definition	/TS_TxDxM6BR0/Com/ComConfig/ComSignal/ComSignalInitValue

Figure 7.74. Importer info value in EB tresos Studio

7.11.2.6. Deleting elements

7.11.2.6.1. Deleting elements based on the importer info value

NOTE

Importer info value is missing in .xdm file



If the importer info value is missing in the .xdm file, it means that the value for that specific parameter was manually edited. In this case, the Service Needs Calculator must ignore it.

This is equivalent to the entry **manually edited** in **Properties -> Information -> Source** in EB tresos Studio.

As of ACG-8.5, the Service Needs Calculator can put some complex information in the importer info value for each parameter that it calculated, and not only for containers it created. In this way, you always know if a specific element was calculated, i.e. updated, by the Service Needs Calculator, and not by another wizard.

This complex information from the .xdm file is shown as follows: **@CALC(SvcAs, service, version)**.



Properties for ComSignalInitValue		
Description	General	
Name	ComSignalInitValue	
Type	String	
Source	calculated by SvcAs,com.signals,1	
ORIGIN	AUTOSAR_ECU	
Multiplicity Config Class	PostBuild	
Value Config Class	PostBuild	
UUID	ECUC:8fbcb350f-1d96-8a4c-2ea7-fb39cbee6276	
Information	Paths	
	XPath	/AUTOSAR/TOP-LEVEL-PACKAGES/Com/ELEMENTS/Com/ComConfig/ComConf
	Parameter Definition	/TS_TxDxM6B3R0/Com/ComConfig/ComSignal/ComSignalInitValue

WARNING**Complex importer info value**

As of ACG-8.5, only parameters which have the complex information for importer info and are calculated by the Service Needs Calculator can be deleted if they become obsolete. One exception applies:

If a container was created by the Service Needs Calculator and has a calculated reference in the configuration and now this container becomes obsolete, its references are deleted even if the complex data from this reference importer info is missing.

Manually edited or imported data is not deleted. Only calculated data is deleted.

7.11.2.6.2. Removing Com elements

In the `Com` module, obsolete elements are removed as follows:

- ▶ If a **comSignal/comSignalGroup** has all its mappings from the Rte Editor and the `.arxml` file removed, the container is seen by the Service Needs Calculator as obsolete. However, because the signal could still be used elsewhere, the Service Needs Calculator should not remove it.

So, if a **comSignal/comSignalGroup** calculated by the Service Needs Calculator becomes obsolete and the Service Needs Calculator is triggered again, all notifications of that signal are deleted and disabled in order to avoid errors in the code generation. The container is not deleted.

The notifications deleted are:

1. ComNotification
2. ComErrorNotification
3. ComInvalidNotification
4. ComTimeoutNotification

- ▶ If a Com container is not obsolete, i.e. still has mappings, and some "calculated by the Service Needs Calculator" children of this container become obsolete, the Service Needs Calculator deletes them.

The Service Needs Calculator does not handle the `Ipdu` references for the **comSignal/comSignalGroup**. You have to manually add/delete these references.



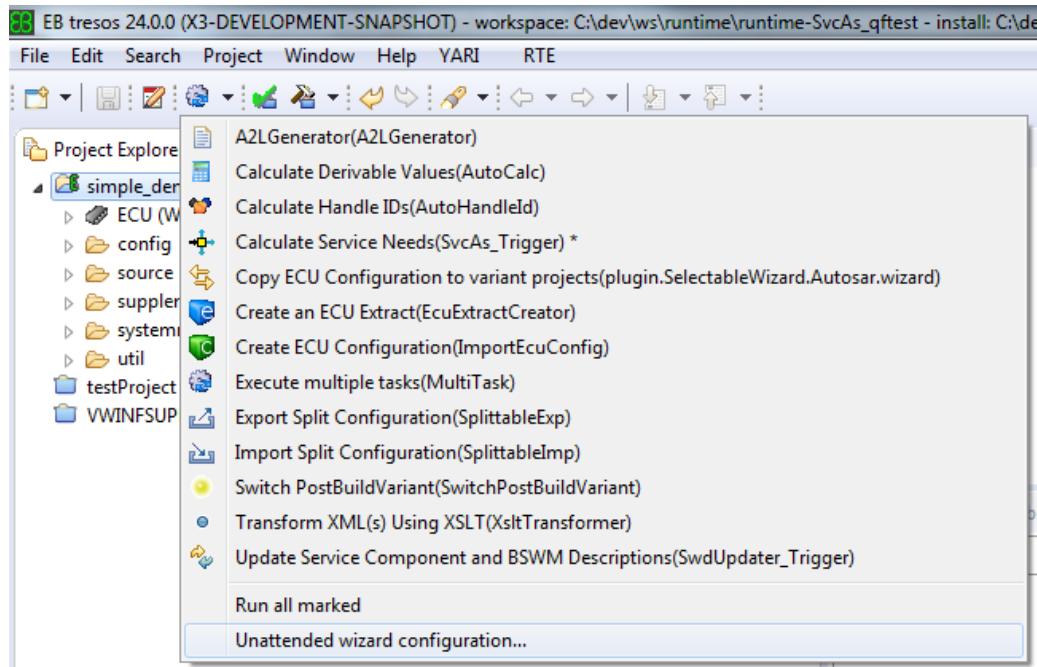
7.11.3. Configuring the Service Needs Calculator

Before using the Service Needs Calculator for the first time, you have to configure it for your project needs. Once configured, you can enable or disable it as required.

To open the Service Needs Calculator configuration dialog in EB tresos Studio, click on the expand arrow next to the menu bar button

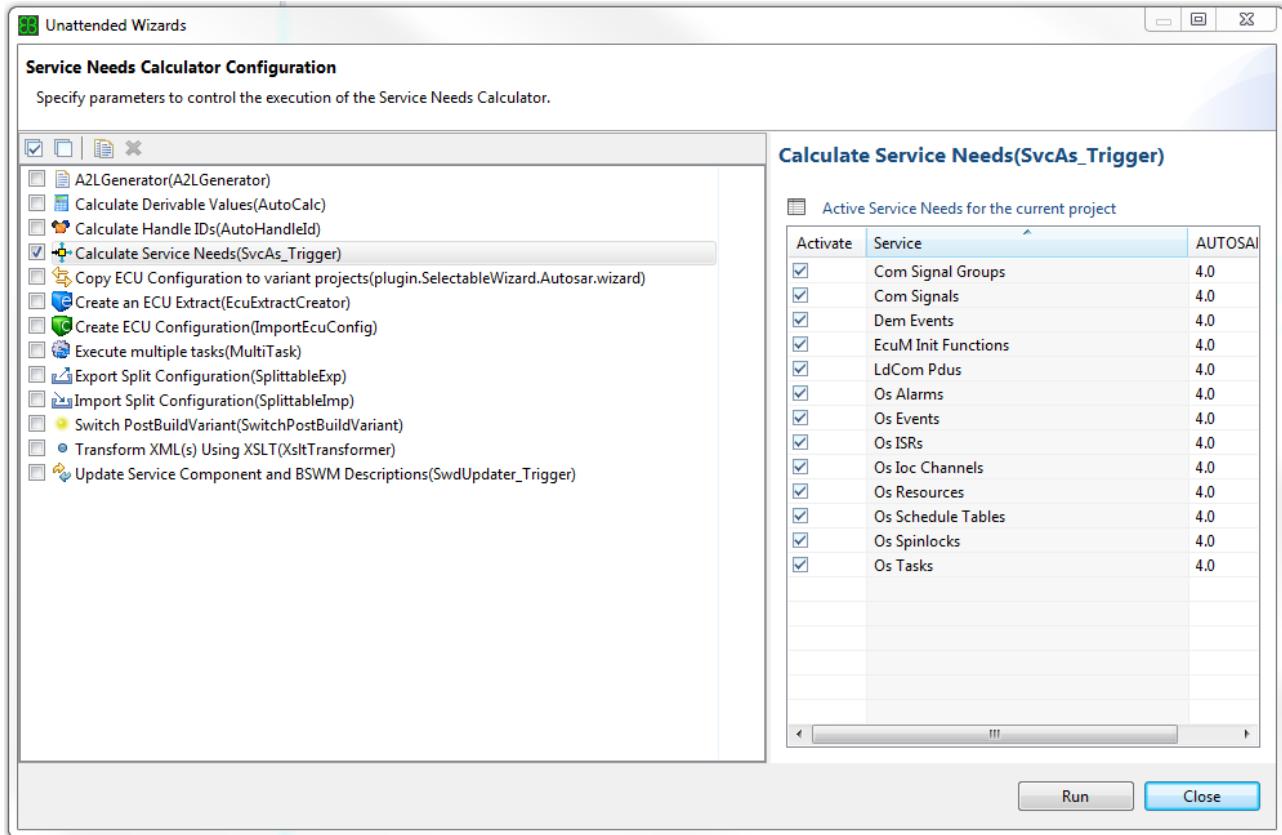


and select **Unattended wizard configuration**.



The **Unattended Wizards** configuration dialog opens. In the list of wizards, click on the label **Calculate Service Needs** to open the Service Needs Calculator configuration. To activate the Service Needs Calculator, enable the check box to the left of the label.

Open the Service Needs Calculator configuration dialog.



The content in the list **Active Service Needs for the current project** is automatically derived from your project configuration. Based on the modules configured, the list of service needs is generated. This list is visible as soon as the configuration dialog opens. You can now enable and disable the calculation for each individual service need.

For example, if you want to enable the automatic addition of Os Events, select the check box next to the corresponding entry in the list.

NOTE



The Service Needs Calculator issues a warning if OsSecondsPerTick is incorrect

If the parameter `OsSecondsPerTick` is not set correctly in the Os configuration, the Service Needs Calculator cannot configure the timing values of the schedule table correctly. In this case, you get a warning message during the execution of the Service Needs Calculator.

7.11.4. Running the Service Needs Calculator

Before you generate the project, run the Service Needs Calculator in the following cases:

- ▶ when you added new modules to your project



- ▶ when you removed modules from your project
- ▶ when you modified parameters in the configuration of a module that have an influence on the existence of a required service

For example, when you add an additional init configuration to the `CanIf`, the `CanIf` requires additional events to be configured in the `Dem`.

After loading a project, execute the following steps:

1. Run the importer `SysD`.
2. Check **Problems View**. If there are any errors, solve them all. The **Update Service Component and BSWM Descriptions** wizard can only run completely if there are no errors in the **Problems View**.
3. Run the **Update Service Component and BSWM Descriptions** wizard.
4. Run the **Create an ECU extract** wizard.
5. Open the **Rte Editor** and click on all tabs to update the data and to make sure there are no errors in the editor. If the **Rte Editor** contains any errors, the Service Needs Calculator cannot execute correctly.
6. Run the **Service Needs Calculator** wizard.

After running the Service Needs Calculator, you can find information about the added, modified or removed elements in the **Results** view as shown below. This view can also show warnings if problems occur during the calculation of the service needs.

7.11.4.1. Results after running the Service Needs Calculator

The following picture shows an example of the possible outcome of a Service Needs Calculator run.



Screenshot of EB tresos® Studio showing the 'Results' tab for a project named 'simple_demo_can_rte'. The tree view shows a hierarchy of modules and their configurations. The 'Message' panel at the bottom lists various warnings and information messages related to the configuration.

Results Tab Content:

- simple_demo_can_rte** project
- ECU (WINDOWS, WIN32X86)** module
 - Com Services** component
 - Com (V6.3.32, AS4.0.3)** component
 - Com** element
 - ComConfig** element
 - ComConfig** element
 - ComSignal** element
 - SGCounterOut_272T** element
 - ComDataInvalidAction*:NOTIFY** parameter
 - ComRxDataTimeoutAction*:NONE** parameter
 - OS/RTE** module
 - Os (V6.0.132, AS4.0.3)** component
 - Os** element
 - OsScheduleTable** element
 - SchM_DefaultScheduleTable*** element
 - OsScheduleTableExpiryPoint** element

The following results are possible:

- ▶ If a module that can be updated by the Service Needs Calculator is missing from your configuration, no information is displayed. In the picture above, this could be `LdCom` or `Xfrm`.
- ▶ If nothing was updated for a specific module, then the arrow for expanding the information is missing. See `Os Alarms` or `Os Resources` in the picture above.
- ▶ If a parameter was manually edited, the information **Ignoring manually edited parameter** is added and the Service Needs Calculator did not update the value.
- ▶ If a container was imported, the information **Ignoring imported parameter** is added. The Service Needs Calculator cannot update the name, but it can update the children values.
- ▶ If a parameter or entry becomes obsolete, the information **Removing obsolete entry** is added and the Service Needs Calculator deletes the obsolete element.



WARNING**Obsolete elements removed by the Service Needs Calculator**

If the Service Needs Calculator deletes elements and this causes errors in the **Problems View**, these elements are recognized as obsolete. The Service Needs Calculator does not receive any requests to update these elements. This means that the configuration is not complete and it needs to be resolved in order to get these requests.

You have to know what to change in the configuration in order to create/delete elements.

The Service Needs Calculator only executes requests and does not decide on what to add or remove.



8. References

8.1. Upgrading system models

In this chapter you will learn when the system model of a project is upgraded to a higher AUTOSAR version and how this is done.

The System Description Importer and the TDB Importer directly import into the system model specified in the imported files; the importers for Fibex, LDF, and DBC files convert the data contained in the imported files into a system model of AUTOSAR version 3.x.

When you create your EB tresos Studio project, you must specify the AUTOSAR release version for this project. If you then use an importer based on an older AUTOSAR major release version to import configuration data, or if you import a system description/TDB file based on an older AUTOSAR major release version, the imported data is automatically upgraded to the project's AUTOSAR release version.

The following sections describe how the data is upgraded between two specific AUTOSAR major release versions. The upgraded entities and their properties are referred to by their XML tags as specified in the respective AUTOSAR XML schema files. Note that only entities defined in the AUTOSAR system template are described, i.e. no description concerning the upgrade of entities defined in the AUTOSAR software component template is provided here.

The following sections will refer to the system model that is upgraded as *source model*, whereas the system model that is the result of the upgrade is called *target model*.

The entities and parameters of source and target model are described in the XML schema files for AUTOSAR 3.1.4 (source model) and AUTOSAR 4.0.2 (target model). In analogy to the AUTOSAR system model path definitions, the notation `<parenttag>/<childtag>` describes an XML child tag `<childtag>` within the context of its enclosing `<parenttag>`.

Wherever the importers for Fibex and LDF files provide useful system data for which no counterpart in the AUTOSAR version 3.x system model exists, the importers store the data in custom parameters (as opposed to canonical AUTOSAR parameters). Since AUTOSAR 4.x introduced parts of these parameters, the model upgrade uses the custom parameters in the source model to configure newly introduced AUTOSAR 4.x parameters in the target model. The following sections explicitly describe which data of the target model is configured during LDF or Fibex import.



8.1.1. Upgrading a system model from AUTOSAR 3.1.x to AUTOSAR 4.0.2

This section describes how an AUTOSAR 3.1.x system model is upgraded to AUTOSAR 4.0.2.

NOTE



This chapter does not describe the upgrade of AUTOSAR 3.2.x system models

This chapter does *not* apply to the upgrade of AUTOSAR 3.2.x system models. For information on upgrading your AUTOSAR 3.2.x system model, refer to [Section 8.1.2, “Upgrading a system model from AUTOSAR 3.2.x to AUTOSAR 4.0.3”](#).

If not stated explicitly, all entities in the 4.0.x target model are located in the same package as they were in the 3.1.x source model.

If not explicitly specified otherwise, the parameters `SHORT-NAME`, `CATEGORY`, and `UUID` of Identifiable entities are directly upgraded into their counterparts in the target model.

8.1.1.1. Topology

8.1.1.1.1. SYSTEM

Source model entity: `SYSTEM`

Target model entity: `SYSTEM`

Target model property	Source model property
<code>SYSTEM-VERSION</code>	<code>SYSTEM-VERSION</code>
<code>FIBEX-ELEMENT-REFS</code>	<code>FIBEX-ELEMENT-REFS</code>
<code>MAPPINGS/SYSTEM-MAPPING</code>	<code>MAPPING</code>
<code>SYSTEM/ROOT-SOFTWARE-COMPOSITIONS/ROOT-SW-COMPOSITION-PROTOTYPE</code>	<code>SYSTEM/SOFTWARE-COMPOSITION</code>

8.1.1.1.1.1. Changed or moved parameters

`FIBEX-ELEMENT-REFS`

In `FIBEX-ELEMENT-REFS`, references to the following types of source model entities are upgraded to their target model counterparts: `CAN-CLUSTER`, `COMMUNICATION-CLUSTER`, `DCM-I-PDU`, `ECU-INSTANCE`, `FLEXRAY-CLUSTER`, `FRAME`, `GATEWAY`, `I-PDU-GROUP`, `I-SIGNAL`, `LIN-CLUSTER`, `MULTIPLEXED-I-PDU`, `N-PDU`, `NM-PDU`, `SIGNAL-I-PDU`, `SUBSTITUTION-FRAME`.



8.1.1.1.2. ECU-INSTANCE

Source model entity: ECU-INSTANCE

Target model entity: ECU-INSTANCE

Target model property	Source model property
ASSOCIATED-COM-I-PDU-GROUP-REFS/ASSOCIATED-COM-I-PDU-GROUP-REF	ASSOCIATED-I-PDU-GROUP-REFS/ASSOCIATED-I-PDU-GROUP-REF
COM-CONFIGURATION-GW-TIME-BASE	COM-PROCESSING-PERIOD
COM-CONFIGURATION-RX-TIME-BASE	COM-PROCESSING-PERIOD
COM-CONFIGURATION-TX-TIME-BASE	COM-PROCESSING-PERIOD
COMM-CONTROLLERS	COMM-CONTROLLERS
CONNECTORS	CONNECTORS
DIAGNOSTIC-ADDRESS	DIAGNOSTIC-ADDRESS
SLEEP-MODE-SUPPORTED	SLEEP-MODE-SUPPORTED
WAKE-UP-OVER-BUS-SUPPORTED	WAKE-UP-OVER-BUS-SUPPORTED
No target model parameter is available.	COM-CONFIGURATION-ID
No target model parameter is available.	PDU-R-CONFIGURATION-ID
Not upgraded.	RESPONSE-ADDRESSSS/RESPONSE-ADDRESS

8.1.1.1.2.1. Reason for not upgraded source model parameter

RESPONSE-ADDRESSSS/RESPONSE-ADDRESS

The parameter RESPONSE-ADDRESSSS/RESPONSE-ADDRESS is not upgraded to the target model since the contained diagnostic addresses cannot be uniquely assigned to concrete interfaces.

8.1.1.1.3. CAN-CLUSTER

Source model entity: CAN-CLUSTER

Target model entity: CAN-CLUSTER

Target model property	Source model property
CAN-CLUSTER-CONDITIONAL/PHYSICAL CHANNELS	PHYSICAL-CHANNELS
CAN-CLUSTER-CONDITIONAL/PROTOCOL-NAME	PROTOCOL-NAME
CAN-CLUSTER-CONDITIONAL/PROTOCOL-VERSION	PROTOCOL-VERSION



Target model property	Source model property
CAN-CLUSTER-CONDITIONAL/SPEED	SPEED/1000
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT

8.1.1.3.1. Changed or moved parameters

NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED

The source model parameters NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED are configured in the target model entity NM-ECU, see [Section 8.1.1.7.2, “NM-ECU”](#).

SPEED

In the target model, the parameter SPEED is given in kbit/s, whereas the source model provides it in bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.

NM-BUS-LOAD-REDUCTION-ACTIVE and NM-WAIT-BUS-SLEEP-TIME

The parameters NM-BUS-LOAD-REDUCTION-ACTIVE and NM-WAIT-BUS-SLEEP-TIME are configured as NM-BUSLOAD-REDUCTION-ACTIVE and NM-WAIT-BUS-SLEEP-TIME in the target model entity CAN-NM-CLUSTER, see [Section 8.1.1.7.3, “CAN-NM-CLUSTER”](#).

NM-BUS-LOAD-REDUCTION-ENABLED

The parameter NM-BUS-LOAD-REDUCTION-ENABLED is configured as NM-BUSLOAD-REDUCTION-ENABLED in the target model entity CAN-NM-CLUSTER-COUPLING, see [Section 8.1.1.7.8, “FLEXRAY-NM-CLUSTER-COUPLING”](#).

NM-LOWER-CAN-ID and NM-UPPER-CAN-ID

The parameters NM-LOWER-CAN-ID and NM-UPPER-CAN-ID are configured in the target model entity CAN-NM-RANGE-CONFIG parameters LOWER-CAN-ID and UPPER-CAN-ID, which is aggregated in CAN-NM-NODE, see [Section 8.1.1.7.4, “CAN-NM-NODE”](#).

8.1.1.4. CAN-COMMUNICATION-CONTROLLER

Source model entity: CAN-COMMUNICATION-CONTROLLER

Target model entity: CAN-COMMUNICATION-CONTROLLER

Target model property	Source model property
CATEGORY	CATEGORY
CAN-COMMUNICATION-CONTROLLER-CONDITIONAL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/SYNC-JUMP-WIDTH	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/SYNC-JUMP-WIDTH



Target model property	Source model property
CAN-COMMUNICATION-CONTROLLER-CONDITION-AL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-1	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-1
CAN-COMMUNICATION-CONTROLLER-CONDITION-AL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-2	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-2
No target model parameter is available.	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/NUMBER-OF-SAMPLES
No target model parameter is available.	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/PROPAGATION-DELAY
CAN-COMMUNICATION-CONTROLLER/CAN-CONTROLLER-CONFIGURATION-REQUIREMENTS	No source model parameter is available.

8.1.1.5. CAN-COMMUNICATION-CONNECTOR

Source model entity: COMMUNICATION-CONNECTOR

Target model entity: CAN-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES

8.1.1.5.1. Changed or moved parameters

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity CAN-PHYSICAL-CHANNEL.

NM-ADDRESS and TP-ADDRESS

The source model parameters NM-ADDRESS and TP-ADDRESS are configured in the target model entities CAN-NM-NODE and TP-ADDRESS, see [Section 8.1.1.7.4, “CAN-NM-NODE”](#).

8.1.1.6. CAN-PHYSICAL-CHANNEL

Source model entities: PHYSICAL-CHANNEL, COMMUNICATION-CONNECTOR



Target model entity: CAN-PHYSICAL-CHANNEL

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-TRIGGERINGS	FRAME-TRIGGERINGSS
PDU-TRIGGERINGS	I-PDU-TRIGGERINGS
I-SIGNAL-TRIGGERINGS	I-SIGNAL-TRIGGERINGS

8.1.1.6.1. Changed or moved parameters

COMM-CONNECTORS

The target model parameter COMM-CONNECTORS is configured by retrieving all target model CAN-COMMUNICATION-CONNECTOR entities that meet the following condition: The source model counterpart of the CAN-COMMUNICATION-CONNECTOR references the source model counterpart of the upgraded CAN-PHYSICAL-CHANNEL via CHANNEL-REF.

CAN-TP-CONNECTION-CHANNEL

The CAN-TP-CONNECTION-CHANNEL entities aggregated in TP-CHALLENS are upgraded to CAN-TP-CONNECTION and CAN-TP-CHANNEL entities in the target model, aggregated in CAN-TP-CONFIG.

8.1.1.7. FLEXRAY-CLUSTER

Source model entity: FLEXRAY-CLUSTER

Target model entity: FLEXRAY-CLUSTER/FLEXRAY-CLUSTER-CONDITIONAL

Target model property	Source model property
PHYSICAL-CHANNELS	PHYSICAL-CHANNELS
PROTOCOL-NAME	PROTOCOL-NAME
PROTOCOL-VERSION	PROTOCOL-VERSION
SPEED	SPEED/1000
ACTION-POINT-OFFSET	ACTION-POINT-OFFSET
BIT	BIT
CAS-RX-LOW-MAX	CAS-RX-LOW-MAX
COLD-START-ATTEMPTS	COLD-START-ATTEMPTS
CYCLE	CYCLE
CYCLE-COUNT-MAX	(NUMBER-OF-CYCLES - 1) or 63 if NUMBER-OF-CYCLES is not available



Target model property	Source model property
DYNAMIC-SLOT-IDLE-PHASE	DYNAMIC-SLOT-IDLE-PHASE
LISTEN-NOISE	LISTEN-NOISE
MACRO-PER-CYCLE	MACRO-PER-CYCLE
MACROTICK-DURATION	MACROTICK-DURATION
MAX-WITHOUT-CLOCK-CORRECTION-FATAL	MAX-WITHOUT-CLOCK-CORRECTION-FATAL
MAX-WITHOUT-CLOCK-CORRECTION-PASSIVE	MAX-WITHOUT-CLOCK-CORRECTION-PASSIVE
MINISLOT-ACTION-POINT-OFFSET	MINISLOT-ACTION-POINT-OFFSET
MINISLOT-DURATION	MINISLOT-DURATION
NETWORK-IDLE-TIME	NETWORK-IDLE-TIME
NETWORK-MANAGEMENT-VECTOR-LENGTH	NETWORK-MANAGEMENT-VECTOR-LENGTH
NUMBER-OF-MINISLOTS	NUMBER-OF-MINISLOTS
NUMBER-OF-STATIC-SLOTS	NUMBER-OF-STATIC-SLOTS
OFFSET-CORRECTION-START	OFFSET-CORRECTION-START
PAYLOAD-LENGTH-STATIC	PAYLOAD-LENGTH-STATIC
SAMPLE-CLOCK-PERIOD	SAMPLE-CLOCK-PERIOD
STATIC-SLOT-DURATION	STATIC-SLOT-DURATION
SYMBOL-WINDOW	SYMBOL-WINDOW
SYMBOL-WINDOW-ACTION-POINT-OFFSET	ACTION-POINT-OFFSET
SYNC-FRAME-ID-COUNT-MAX	SYNC-NODE-MAX
TRANSMISSION-START-SEQUENCE-DURATION	TRANSMISSION-START-SEQUENCE-DURATION
WAKEUP-RX-IDLE	WAKE-UP-SYMBOL-RX-IDLE
WAKEUP-RX-LOW	WAKE-UP-SYMBOL-RX-LOW
WAKEUP-RX-WINDOW	WAKE-UP-SYMBOL-RX-WINDOW
WAKEUP-TX-ACTIVE	WAKE-UP-SYMBOL-TX-LOW
WAKEUP-TX-IDLE	WAKE-UP-SYMBOL-TX-IDLE
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT
No target model parameter is available.	BUS-GUARDIAN-ENABLE-PART
No target model parameter is available.	CAS-RX-LOW-MIN
No target model parameter is available.	MACRO-INITIAL-OFFSET
No target model parameter is available.	MAX-INITIALISATION-ERROR



Target model property	Source model property
No target model parameter is available.	MAX-PROPAGATION-DELAY
No target model parameter is available.	MIN-PROPAGATION-DELAY
No target model parameter is available.	OFFSET-CORRECTION-MAX
DETECT-NIT-ERROR	No source model parameter is available.
IGNORE-AFTER-TX	No source model parameter is available.
SAFETY-MARGIN	No source model parameter is available.

8.1.1.7.1. Changed or moved parameters

NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED

The source model parameters NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED are configured in the target model entity NM-ECU, see [Section 8.1.1.7.2, "NM-ECU"](#).

SPEED

In the target model, the parameter SPEED is given in kBit/s, whereas the source model provides it in Bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.

8.1.1.8. FLEXRAY-COMMUNICATION-CONTROLLER

Source model entity: FLEXRAY-COMMUNICATION-CONTROLLER

Target model entity: FLEXRAY-COMMUNICATION-CONTROLLER/FLEXRAY-COMMUNICATION-CONTROLLER-CONDITIONAL

Target model property	Source model property
CATEGORY	CATEGORY
ACCEPTED-STARTUP-RANGE	ACCEPTED-STARTUP-RANGE
ALLOW-HALT-DUE-TO-CLOCK	ALLOW-HALT-DUE-TO-CLOCK
ALLOW-PASSIVE-TO-ACTIVE	ALLOW-PASSIVE-TO-ACTIVE
CLUSTER-DRIFT-DAMPING	CLUSTER-DRIFT-DAMPING
DECODING-CORRECTION	DECODING-CORRECTION
DELAY-COMPENSATION-A	DELAY-COMPENSATION-A
DELAY-COMPENSATION-B	DELAY-COMPENSATION-B
EXTERN-OFFSET-CORRECTION	EXTERN-OFFSET-CORRECTION
EXTERN-RATE-CORRECTION	EXTERN-RATE-CORRECTION
KEY-SLOT-ID	KEY-SLOT-ID



Target model property	Source model property
KEY-SLOT-ONLY-ENABLED	SINGLE-SLOT-ENABLED
KEY-SLOT-USED-FOR-START-UP	KEY-SLOT-USED-FOR-START-UP
KEY-SLOT-USED-FOR-SYNC	KEY-SLOT-USED-FOR-SYNC
LATEST-TX	LATEST-TX
LISTEN-TIMEOUT	LISTEN-TIMEOUT
MACRO-INITIAL-OFFSET-A	MACRO-INITIAL-OFFSET-A
MACRO-INITIAL-OFFSET-B	MACRO-INITIAL-OFFSET-B
MAXIMUM-DYNAMIC-PAYLOAD-LENGTH	MAXIMUM-DYNAMIC-PAYLOAD-LENGTH
MICRO-INITIAL-OFFSET-A	MICRO-INITIAL-OFFSET-A
MICRO-INITIAL-OFFSET-B	MICRO-INITIAL-OFFSET-B
MICRO-PER-CYCLE	MICRO-PER-CYCLE
MICROTICK-DURATION	MICROTICK-DURATION
OFFSET-CORRECTION-OUT	OFFSET-CORRECTION-OUT
RATE-CORRECTION-OUT	RATE-CORRECTION-OUT
SAMPLES-PER-MICROTICK	SAMPLES-PER-MICROTICK
WAKE-UP-PATTERN	WAKE-UP-PATTERN
No target model parameter is available.	DYNAMIC-SEGMENT-ENABLE
No target model parameter is available.	MAX-DRIFT
No target model parameter is available.	MICRO-PER-MACRO-NOM
No target model parameter is available.	START-UP-NODE
No target model parameter is available.	SYNC-SLOT
EXTERNAL-SYNC	No source model parameter is available.
FALL-BACK-INTERNAL	No source model parameter is available.
FLEXRAY-FIFOS	No source model parameter is available.
NM-VECTOR-EARLY-UPDATE	No source model parameter is available.
SECOND-KEY-SLOT-ID	No source model parameter is available.
TWO-KEY-SLOT-MODE	No source model parameter is available.

8.1.1.9. FLEXRAY-COMMUNICATION-CONNECTOR

Source model entity: FLEX-RAY-COMMUNICATION-CONNECTOR

Target model entity: FLEXRAY-COMMUNICATION-CONNECTOR



Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
WAKE-UP-CHANNEL	WAKE-UP-CHANNEL

8.1.1.9.1. Changed or moved parameters

ECU-COMM-PORT-INSTANCES

In ECU-COMM-PORT-INSTANCES, the aggregated source model entities are upgraded to their target model counterparts: FRAME-PORT, I-PDU-PORT, SIGNAL-PORT.

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity FLEXRAY-PHYSICAL-CHANNEL.

NM-ADDRESS and TP-ADDRESS

The source model parameters NM-ADDRESS and TP-ADDRESS are configured in the target model entities FLEXRAY-NM-NODE and TP-ADDRESS, see [Section 8.1.1.7.7, “FLEXRAY-NM-NODE”](#).

8.1.1.10. FLEXRAY-PHYSICAL-CHANNEL

Source model entities: PHYSICAL-CHANNEL, FLEX-RAY-COMMUNICATION-CONNECTOR

Target model entity: FLEXRAY-PHYSICAL-CHANNEL

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-TRIGGERINGS	FRAME-TRIGGERINGSS
PDU-TRIGGERINGS	I-PDU-TRIGGERINGS
I-SIGNAL-TRIGGERINGS	I-SIGNAL-TRIGGERINGS
CHANNEL-NAME	CHANNEL-NAME

8.1.1.10.1. Changed or moved parameters

COMM-CONNECTORS

The target model parameter COMM-CONNECTORS is configured by retrieving all target model FLEX-RAY-COMMUNICATION-CONNECTOR entities that meet the following condition: The source model counterpart of



the FLEX-RAY-COMMUNICATION-CONNECTOR references the source model counterpart of the upgraded FLEXRAY-PHYSICAL-CHANNEL via CHANNEL-REF.

FLEX-RAY-TP-CHANNEL entities in TP-CHANNELS

The FLEX-RAY-TP-CHANNEL entities aggregated in TP-CHANNELS are upgraded to FLEXRAY-TP-CONNECTION-CONTROL entities in the target model, aggregated in FLEXRAY-TP-CONFIG, see [Section 8.1.1.5.1, "FLEXRAY-TP-CONFIG"](#).

8.1.1.11. LIN-CLUSTER

Source model entity: LIN-CLUSTER

Target model entity: LIN-CLUSTER/LIN-CLUSTER-CONDITIONAL

Target model property	Source model property
PHYSICAL-CHANNELS	PHYSICAL-CHANNELS
PROTOCOL-NAME	PROTOCOL-NAME
PROTOCOL-VERSION	PROTOCOL-VERSION
SPEED	SPEED/1000
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT

8.1.1.11.1. Changed or moved parameters

SPEED

In the target model, the parameter SPEED is given in kbit/s, whereas the source model provides it in bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.

LIN-SCHEDULE-TABLE entities of the source model parameter SCHEDULE-TABLES

The LIN-SCHEDULE-TABLE entities aggregated in the source model in SCHEDULE-TABLES are moved to the corresponding LIN-PHYSICAL-CHANNEL in the target model, see [Section 8.1.1.16, "LIN-PHYSICAL-CHANNEL"](#).

8.1.1.12. LIN-MASTER

Source model entity: LIN-MASTER

Target model entity: LIN-MASTER/LIN-MASTER-CONDITIONAL

Target model property	Source model property
CATEGORY	CATEGORY



Target model property	Source model property
TIME-BASE	TIME-BASE
TIME-BASE-JITTER	TIME-BASE-JITTER
PROTOCOL-VERSION	No source model parameter is available.

8.1.1.13. LIN-SLAVE

Source model entity: LIN-SLAVE

Target model entity: LIN-SLAVE/LIN-SLAVE-CONDITIONAL

Target model property	Source model property
CATEGORY	CATEGORY
CONFIGURED-NAD	CONFIGURED-NAD
LIN-ERROR-RESPONSE	LIN-ERROR-RESPONSE
PROTOCOL-VERSION	PROTOCOL-VERSION
SUPPLIER-ID	Source model parameters are available via LDF import only.
FUNCTION-ID	Source model parameters are available via LDF import only.
VARIANT-ID	Source model parameters are available via LDF import only.

8.1.1.13.1. Reasons for not configured target model parameters

SUPPLIER-ID, FUNCTION-ID, VARIANT-ID

The target model parameters SUPPLIER-ID, FUNCTION-ID, VARIANT-ID have no counterparts in the source model. Therefore they are only configured if the imported data originates from an LDF import, see [Section 6.10.2.7, “Importing from the LDF format”](#).

8.1.1.14. LIN-ERROR-RESPONSE

Source model entity: LIN-ERROR-RESPONSE

Target model entity: LIN-ERROR-RESPONSE

Target model property	Source model property
FRAME-TRIGGERING-REF	FRAME-TRIGGERING-REF



Target model property	Source model property
RESPONSE-ERROR-POSITION	RESPONSE-ERROR-POSITION

8.1.1.15. LIN-COMMUNICATION-CONNECTOR

Source model entity: COMMUNICATION-CONNECTOR

Target model entity: LIN-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
LIN-CONFIGURABLE-FRAMES/LIN-CONFIGURABLE-FRAME	Source model parameters are available via LDF import only.
LIN-ORDERED-CONFIGURABLE-FRAMES/LIN-ORDERED-CONFIGURABLE-FRAME	Source model parameters are available via LDF import only.

8.1.1.15.1. Reason for not configured target model parameters

LIN-CONFIGURABLE-FRAMES, and LIN-CONFIGURABLE-FRAME, LIN-ORDERED-CONFIGURABLE-FRAMES, and LIN-ORDERED-CONFIGURABLE-FRAME

The target model parameters LIN-CONFIGURABLE-FRAMES/LIN-CONFIGURABLE-FRAME and LIN-ORDERED-CONFIGURABLE-FRAMES/LIN-ORDERED-CONFIGURABLE-FRAME are only configured if the imported data originates from an LDF import. See [8] for details about the LDF file format.

8.1.1.15.2. Changed or moved parameters

ECU-COMM-PORT-INSTANCES

In ECU-COMM-PORT-INSTANCES, the aggregated source model entities are upgraded into their target model counterparts: FRAME-PORT, I-PDU-PORT, SIGNAL-PORT.

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity LIN-PHYSICAL-CHANNEL.

TP-ADDRESS

The source model parameter TP-ADDRESS is configured in the target model entity TP-ADDRESS, see [Section 8.1.1.6.1, "LIN-TP-CONFIG".](#)

**INITIAL-NAD**

The target model parameter **INITIAL-NAD** is configured using data from LDF imports directly. If no LDF file has been imported or the LDF file does not contain the required data, the **CONFIGURED-NAD** parameter value of the **LIN-SLAVE** that is referenced via **COMM-CONTROLLER-REF** is taken as source.

8.1.1.16. LIN-PHYSICAL-CHANNEL

Source model entities: PHYSICAL-CHANNEL, COMMUNICATION-CONNECTOR

Target model entity: LIN-PHYSICAL-CHANNEL

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-TRIGGERINGS	FRAME-TRIGGERINGSS
PDU-TRIGGERINGS	I-PDU-TRIGGERINGS
I-SIGNAL-TRIGGERINGS	I-SIGNAL-TRIGGERINGS

8.1.1.16.1. Changed or moved parameters**COMM-CONNECTORS**

The target model parameter **COMM-CONNECTORS** is configured by retrieving all target model **COMMUNICATION-CONNECTOR** entities that meet the following condition: The source model counterpart of the **COMMUNICATION-CONNECTOR** references the source model counterpart of the upgraded **LIN-PHYSICAL-CHANNEL** via **CHANNEL-REF**.

LIN-TP-CHANNEL entities aggregated in TP-CHANNELS

The **LIN-TP-CHANNEL** entities aggregated in **TP-CHANNELS** are upgraded to **LIN-TP-CONNECTION** entities in the target model, which are now aggregated in **LIN-TP-CONFIG**, see [Section 8.1.1.6.1, “LIN-TP-CONFIG”](#).

LIN-PHYSICAL-CHANNEL

The **LIN-PHYSICAL-CHANNEL** aggregates in its parameter **SCHEDULE-TABLES** all **LIN-SCHEDULE-TABLE** entities the source model entity **LIN-CLUSTER** aggregates in its corresponding parameter.

8.1.1.2. Communication**8.1.1.2.1. CAN-FRAME-TRIGGERING**

Source model entity: CAN-FRAME-TRIGGERING



Target model entity: CAN-FRAME-TRIGGERING

Target model property	Source model property
CAN-ADDRESSING-MODE	CAN-ADDRESSING-MODE
CATEGORY	CATEGORY
IDENTIFIER	IDENTIFIER
FRAME-PORT-REFS / FRAME-PORT-REF	FRAME-PORT-REFS / FRAME-PORT-REF
FRAME-REF	FRAME-REF
PDU-TRIGGERINGS / PDU-TRIGGERING-REF-CONDITIONAL	I-PDU-TRIGGERING-REFS / I-PDU-TRIGGERING-REF

8.1.1.2.2. FLEXRAY-FRAME-TRIGGERING

Source model entity: FLEXRAY-FRAME-TRIGGERING

Target model entity: FLEXRAY-FRAME-TRIGGERING

Target model property	Source model property
ABSOLUTELY-SCHEDULED-TIMINGS / FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING	ABSOLUTELY-SCHEDULED-TIMINGS / ABSOLUTE-LY-SCHEDULED-TIMING
CATEGORY	CATEGORY
PAYLOAD-PREAMBLE-INDICATOR	PAYLOAD-PREAMBLE-INDICATOR
FRAME-PORT-REFS / FRAME-PORT-REF	FRAME-PORT-REFS / FRAME-PORT-REF
FRAME-REF	FRAME-REF
PDU-TRIGGERINGS / PDU-TRIGGERING-REF-CONDITIONAL	I-PDU-TRIGGERING-REFS / I-PDU-TRIGGERING-REF
ALLOW-DYNAMIC-L-SDU-LENGTH	No source model parameter is available.
MESSAGE-ID	No source model parameter is available.

8.1.1.2.3. FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING

Source model entity: ABSOLUTELY-SCHEDULED-TIMING

Target model entity: FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING

Target model property	Source model property
COMMUNICATION-CYCLE / CYCLE-COUNTER	COMMUNICATION-CYCLE / CYCLE-COUNTER



Target model property	Source model property
COMMUNICATION-CYCLE/CYCLE-REPETITION/ BASE-CYCLE	COMMUNICATION-CYCLE/CYCLE-REPETITION/ BASE-CYCLE
COMMUNICATION-CYCLE/CYCLE-REPETITION/ CYCLE-REPETITION	COMMUNICATION-CYCLE/CYCLE-REPETITION/ CYCLE-REPETITION
SLOT-ID	SLOT-ID

8.1.1.2.4. LIN-FRAME-TRIGGERING

Source model entity: LIN-FRAME-TRIGGERING

Target model entity: LIN-FRAME-TRIGGERING

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-PORT-REFS/FRAME-PORT-REF	FRAME-PORT-REFS/FRAME-PORT-REF
FRAME-REF	FRAME-REF
IDENTIFIER	IDENTIFIER
LIN-CHECKSUM	CHECKSUM-TYPE
PDU-TRIGGERINGS/PDU-TRIGGERING-REF- CONDITIONAL	I-PDU-TRIGGERING-REFS/I-PDU-TRIG- GERING-REF

8.1.1.2.4.1. Changed or moved parameters

RELATIVELY-SCHEDULED-TIMING

The source model RELATIVELY-SCHEDULED-TIMING entities aggregated in LIN-FRAME-TRIGGERING/RELATIVELY-SCHEDULED-TIMINGS are now aggregated in the target model entity LIN-SCHEDULE-TABLE, see [Section 8.1.1.2.10, "LIN-SCHEDULE-TABLE"](#).

8.1.1.2.5. CAN-FRAME

Source model entity: FRAME

Target model entity: CAN-FRAME

Target model property	Source model property
CATEGORY	CATEGORY



Target model property	Source model property
FRAME-LENGTH	FRAME-LENGTH
PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAPPING	PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAPPING

8.1.1.2.5.1. Changed or moved parameters

FRAME-TRIGGERING

If the FRAME-TRIGGERING referencing FRAME is of type CAN-FRAME-TRIGGERING, the FRAME is upgraded to a CAN-FRAME in the target model.

If FRAME-TRIGGERINGS of different cluster types reference one and the same FRAME in the source model, a copy of the FRAME is created in the target model for each cluster type. If, for instance, one CAN-FRAME-TRIGGERING and one FLEXRAY-FRAME-TRIGGERING reference the same FRAME element in the source model, the FRAME element is upgraded to one CAN-FRAME and one FLEXRAY-FRAME in the target model.

8.1.1.2.6. FLEXRAY-FRAME

Source model entity: FRAME

Target model entity: FLEXRAY-FRAME

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-LENGTH	FRAME-LENGTH
PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAPPING	PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAPPING

8.1.1.2.6.1. Changed or moved parameters

FRAME-TRIGGERING

If the FRAME-TRIGGERING referencing a FRAME is of type FLEXRAY-FRAME-TRIGGERING, the FRAME is upgraded to a FLEXRAY-FRAME in the target model.

If FRAME-TRIGGERINGS of different cluster types reference one and the same FRAME in the source model, a copy of the FRAME is created in the target model for each cluster type. If, for instance, one CAN-FRAME-TRIGGERING and one FLEXRAY-FRAME-TRIGGERING reference the same FRAME element in the source model, the FRAME element is upgraded to one CAN-FRAME and one FLEXRAY-FRAME in the target model.



8.1.1.2.7. LIN-UNCONDITIONAL-FRAME

Source model entity: FRAME

Target model entity: LIN-UNCONDITIONAL-FRAME

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-LENGTH	FRAME-LENGTH
PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAP-PING	PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAP-PING

8.1.1.2.7.1. Changed or moved parameters

FRAME-TRIGGERING

If the FRAME-TRIGGERING referencing a FRAME is of type LIN-FRAME-TRIGGERING, the FRAME is upgraded to a LIN-UNCONDITIONAL-FRAME in the target model.

If FRAME-TRIGGERINGS of different cluster types reference one and the same FRAME in the source model, a copy of the FRAME is created in the target model for each cluster type. If, for instance, one CAN-FRAME-TRIGGERING and one FLEXRAY-FRAME-TRIGGERING reference the same FRAME element in the source model, the FRAME element is upgraded to one CAN-FRAME and one FLEXRAY-FRAME in the target model.

8.1.1.2.8. LIN-SPORADIC-FRAME

Source model entity: SUBSTITUTION-FRAME

Target model entity: LIN-SPORADIC-FRAME

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-LENGTH	FRAME-LENGTH
PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAP-PING	PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAP-PING
SUBSTITUTED-FRAME-REFS / SUBSTITUTED-FRAME-REF	SUBSTITUTED-FRAME-REFS / SUBSTITUTED-FRAME-REF

**NOTE****The parameters FRAME-LENGTH, and PDU-TO-FRAME-MAPPINGS/PDU-TO-FRAME-MAPPING might not bear useful information**

The parameters FRAME-LENGTH, and PDU-TO-FRAME-MAPPINGS/PDU-TO-FRAME-MAPPING only exist due to inheritance peculiarities of the source and target models. In the context of a LIN-SPORADIC-FRAME or SUBSTITUTION-FRAME, they do not bear useful information.

8.1.1.2.8.1. Changed or moved parameters**SUBSTITUTION-FRAME**

If the SUBSTITUTION-FRAME in the source model has its parameter SUBSTITUTION-TYPE set to SPORADIC, it is upgraded to a LIN-SPORADIC-FRAME in the target model.

8.1.1.2.9. LIN-EVENT-TRIGGERED-FRAME

Source model entity: SUBSTITUTION-FRAME

Target model entity: LIN-EVENT-TRIGGERED-FRAME

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-LENGTH	FRAME-LENGTH
PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAPPING	PDU-TO-FRAME-MAPPINGS / PDU-TO-FRAME-MAPPING
LIN-UNCONDITIONAL-FRAME-REFS / LIN-UN-CONDITIONAL-FRAME-REF	SUBSTITUTED-FRAME-REFS / SUBSTITUTED-FRAME-REF
COLLISION-RESOLVING-SCHEDULE-REF	Source model parameters are available via LDF 2.1 import only.

**NOTE****The parameters FRAME-LENGTH, and PDU-TO-FRAME-MAPPINGS/PDU-TO-FRAME-MAPPING might not contain useful information**

The parameters FRAME-LENGTH, and PDU-TO-FRAME-MAPPINGS/PDU-TO-FRAME-MAPPING only exist due to inheritance peculiarities of the source and target models. In the context of a LIN-EVENT-TRIGGERED-FRAME or SUBSTITUTION-FRAME, they do not bear useful information.

8.1.1.2.9.1. Changed or moved parameters**SUBSTITUTION-FRAME**

If the SUBSTITUTION-FRAME in the source model has its parameter SUBSTITUTION-TYPE set to EVENT-TRIGGERED, it is upgraded to a LIN-EVENT-TRIGGERED-FRAME in the target model.

8.1.1.2.10. LIN-SCHEDULE-TABLE

Source model entity: LIN-SCHEDULE-TABLE

Target model entity: LIN-SCHEDULE-TABLE

Target model property	Source model property
CATEGORY	CATEGORY
RUN-MODE	RUN-MODE
RESUME-POSITION	No source model parameter is available.
No target model parameter is available.	PRIORITY

8.1.1.2.10.1. Changed or moved parameters**LIN-SCHEDULE-TABLE/TABLE-ENTRYS**

The target model parameter LIN-SCHEDULE-TABLE/TABLE-ENTRYS aggregates all source model RELATIVELY-SCHEDULED-TIMING entities that reference the corresponding LIN-SCHEDULE-TABLE in the source model.

8.1.1.2.11. ASSIGN-FRAME-ID

Source model entity: ASSIGN-FRAME-ID-TIMING

Target model entity: ASSIGN-FRAME-ID



Target model property	Source model property
ASSIGNED-CONTROLLER-REF	ASSIGNED-CONTROLLER-REF
DELAY	DELAY
POSITION-IN-TABLE	POSITION-IN-TABLE
ASSIGNED-FRAME-TRIGGERING-REF	ASSIGNED-FRAME-TRIGGERING-REF

8.1.1.2.12. UNASSIGN-FRAME-ID

Source model entity: UNASSIGN-FRAME-ID-TIMING

Target model entity: UNASSIGN-FRAME-ID

Target model property	Source model property
ASSIGNED-CONTROLLER-REF	ASSIGNED-CONTROLLER-REF
DELAY	DELAY
POSITION-IN-TABLE	POSITION-IN-TABLE
UNASSIGNED-FRAME-TRIGGERING-REF	UNASSIGNED-FRAME-TRIGGERING-REF

8.1.1.2.13. ASSIGN-NAD

Source model entity: ASSIGN-NAD-TIMING

Target model entity: ASSIGN-NAD

Target model property	Source model property
ASSIGNED-CONTROLLER-REF	ASSIGNED-CONTROLLER-REF
DELAY	DELAY
POSITION-IN-TABLE	POSITION-IN-TABLE
NEW-NAD	NEW-NAD

8.1.1.2.14. FREE-FORMAT

Source model entity: DATA-TIMING

Target model entity: FREE-FORMAT

Target model property	Source model property
ASSIGNED-CONTROLLER-REF	ASSIGNED-CONTROLLER-REF



Target model property	Source model property
DELAY	DELAY
POSITION-IN-TABLE	POSITION-IN-TABLE
BYTE-VALUES/BYTE-VALUE	FREE-FORMAT/BYTE-VALUES/BYTE-VALUE

8.1.1.2.15. ASSIGN-FRAME-ID-RANGE

Source model entity: No AUTOSAR source model entity is available. The data is only available via LDF 2.-1 file imports.

Target model entity: ASSIGN-FRAME-ID-RANGE

Target model property	Source model property
ASSIGNED-CONTROLLER-REF	Source model parameters are available via LDF 2.1 import only.
DELAY	Source model parameters are available via LDF 2.1 import only.
POSITION-IN-TABLE	Source model parameters are available via LDF 2.1 import only.
FRAME-PIDS/FRAME-PID/INDEX	Source model parameters are available via LDF 2.1 import only.
FRAME-PIDS/FRAME-PID/PID	Source model parameters are available via LDF 2.1 import only.
START-INDEX	Source model parameters are available via LDF 2.1 import only.

8.1.1.2.16. CONDITIONAL-CHANGE-NAD

Source model entity: No AUTOSAR source model entity is available. The data is only available via LDF 2.-1 file imports.

Target model entity: CONDITIONAL-CHANGE-NAD

Target model property	Source model property
ASSIGNED-CONTROLLER-REF	Source model parameters are available via LDF 2.1 import only.
DELAY	Source model parameters are available via LDF 2.1 import only.



Target model property	Source model property
POSITION-IN-TABLE	Source model parameters are available via LDF 2.1 import only.
BYTE	Source model parameters are available via LDF 2.1 import only.
ID	Source model parameters are available via LDF 2.1 import only.
INVERT	Source model parameters are available via LDF 2.1 import only.
MASK	Source model parameters are available via LDF 2.1 import only.
NEW-NAD	Source model parameters are available via LDF 2.1 import only.

8.1.1.2.17. DATA-DUMP-ENTRY

Source model entity: No AUTOSAR source model entity is available. The data is only available via LDF 2.-1 file imports.

Target model entity: DATA-DUMP-ENTRY

Target model property	Source model property
ASSIGNED-CONTROLLER-REF	Source model parameters are available via LDF 2.1 import only.
DELAY	Source model parameters are available via LDF 2.1 import only.
POSITION-IN-TABLE	Source model parameters are available via LDF 2.1 import only.
BYTE-VALUES/BYTE-VALUE	Source model parameters are available via LDF 2.1 import only.

8.1.1.2.18. SAVE-CONFIGURATION-ENTRY

Source model entity: No AUTOSAR source model entity is available. The data is only available via LDF 2.-1 file imports.

Target model entity: SAVE-CONFIGURATION-ENTRY



Target model property	Source model property
ASSIGNED-CONTROLLER-REF	Source model parameters are available via LDF 2.1 import only.
DELAY	Source model parameters are available via LDF 2.1 import only.
POSITION-IN-TABLE	Source model parameters are available via LDF 2.1 import only.

8.1.1.2.19. PDU-TRIGGERING

Source model entity: I-PDU-TRIGGERING

Target model entity: PDU-TRIGGERING

Target model property	Source model property
CATEGORY	CATEGORY
I-PDU-PORT-REFS / I-PDU-PORT-REF	I-PDU-PORT-REFS / I-PDU-PORT-REF
I-PDU-REF	I-PDU-REF
I-SIGNAL-TRIGGERINGS / I-SIGNAL-TRIGGERING-REF-CONDITIONAL	I-SIGNAL-TRIGGERING-REFS / I-SIGNAL-TRIGGERING-REF
No target model parameter is available.	TIMING-REQUIREMENT

8.1.1.2.20. I-SIGNAL-TRIGGERING

Source model entity: I-SIGNAL-TRIGGERING

Target model entity: I-SIGNAL-TRIGGERING

Target model property	Source model property
CATEGORY	CATEGORY
I-SIGNAL-GROUP-REF	SIGNAL-REF
I-SIGNAL-REF	SIGNAL-REF
I-SIGNAL-PORT-REFS / I-SIGNAL-PORT-REF	I-SIGNAL-PORT-REFS / I-SIGNAL-PORT-REF



8.1.1.2.20.1. Changed or moved parameters

SIGNAL-REF

The source model parameter SIGNAL-REF is upgraded to the target parameter I-SIGNAL-REF if the referenced source model entity represents an I-SIGNAL that references in turn a SYSTEM-SIGNAL. If the referenced I-SIGNAL references a SYSTEM-SIGNAL-GROUP, the parameter is upgraded to I-SIGNAL-GROUP-REF.

8.1.1.2.21. DCM-I-PDU

Source model entity: DCM-I-PDU

Target model entity: DCM-I-PDU

Target model property	Source model property
CATEGORY	CATEGORY
LENGTH	LENGTH
DIAG-PDU-TYPE	No source model parameter is available.

8.1.1.2.21.1. Changed or moved parameters

LENGTH

During the conversion process, the parameter LENGTH given in units of bits in the source model is converted to bytes as required by the target model.

8.1.1.2.22. I-SIGNAL-I-PDU

Source model entity: SIGNAL-I-PDU

Target model entity: I-SIGNAL-I-PDU

Target model property	Source model property
CATEGORY	CATEGORY
I-PDU-TIMING-SPECIFICATIONS/I-PDU-TIMING	I-PDU-TIMING-SPECIFICATION
I-SIGNAL-TO-PDU-MAPPINGS/I-SIGNAL-TO-I-PDU-MAPPING	SIGNAL-TO-PDU-MAPPINGS/I-SIGNAL-TO-I-PDU-MAPPING



Target model property	Source model property
LENGTH	LENGTH
UNUSED-BIT-PATTERN	UNUSED-BIT-PATTERN
PDU-COUNTERS/SIGNAL-I-PDU-COUNTER	No source model parameter is available.
PDU-REPLICATIONS/SIGNAL-I-PDU-REPLICATION	No source model parameter is available.

8.1.1.2.22.1. Changed or moved parameters

LENGTH

During the conversion process, the parameter LENGTH given in units of bits in the source model is converted to bytes as required by the target model.

8.1.1.2.23. MULTIPLEXED-I-PDU

Source model entity: MULTIPLEXED-I-PDU

Target model entity: MULTIPLEXED-I-PDU

Target model property	Source model property
CATEGORY	CATEGORY
DYNAMIC-PARTS/DYNAMIC-PART	DYNAMIC-PART
LENGTH	LENGTH
SELECTOR-FIELD-BYTE-ORDER	SELECTOR-FIELD-BYTE-ORDER
SELECTOR-FIELD-LENGTH	SELECTOR-FIELD-LENGTH
SELECTOR-FIELD-START-POSITION	SELECTOR-FIELD-START-POSITION
STATIC-PARTS/STATIC-PART	STATIC-PART
TRIGGER-MODE	TRIGGER-MODE
UNUSED-BIT-PATTERN	UNUSED-BIT-PATTERN

8.1.1.2.23.1. DYNAMIC-PART and STATIC-PART: differences between source model and the EB tresos Studio system model

The system model provided by EB tresos Studio differs from the source model in such a way that it allows multiple DYNAMIC-PART elements per MULTIPLEXED-I-PDU. In the upgrade process, these multi-



ple DYNAMIC-PART elements are aggregated into one single DYNAMIC-PARTS/DYNAMIC-PART element, which in turn aggregates multiple SEGMENT-POSITION and multiple DYNAMIC-PART-ALTERNATIVES/DYNAMIC-PART-ALTERNATIVE parameters.

For each unique selector field code found in any DYNAMIC-PART, one DYNAMIC-PART-ALTERNATIVE element is created, which contains the code in its SELECTOR-FIELD-CODE parameter. In the target model, DYNAMIC-PART-ALTERNATIVE/I-PDU-REF references the I-PDU that this selector field code is addressing. DYNAMIC-PART-ALTERNATIVE/INITIAL-DYNAMIC-PART is set to TRUE for the selector field code configured to represent the initially selected DYNAMIC-PART.

Moreover, the system model provided by EB tresos Studio allows multiple STATIC-PART elements per MULTIPLEXED-I-PDU. In the upgrade process, these multiple STATIC-PART elements are aggregated to one single STATIC-PARTS/STATIC-PART element, which in turn aggregates multiple SEGMENT-POSITION parameters.

8.1.1.2.23.2. Changed or moved parameters

LENGTH

During the conversion process, the parameter LENGTH given in units of bits in the source model is converted to bytes as required by the target model.

8.1.1.2.24. DYNAMIC-PART

Source model entity: DYNAMIC-PART

Target model entity: DYNAMIC-PART

Target model property	Source model property
DYNAMIC-PART-ALTERNATIVES/DYNAMIC-PART-ALTERNATIVE	DYNAMIC-PART-ALTERNATIVES/DYNAMIC-PART-ALTERNATIVE
SEGMENT-POSITIONS/SEGMENT-POSITION	SEGMENT-POSITIONS/SEGMENT-POSITION

NOTE

The source model differs from the EB tresos Studio system model



The EB tresos Studio internal system model differs from the source model in how MULTIPLEXED-I-PDUS are stored, see [Section 8.1.1.2.23, "MULTIPLEXED-I-PDU"](#).

8.1.1.2.25. DYNAMIC-PART-ALTERNATIVE

Source model entity: DYNAMIC-PART-ALTERNATIVE



Target model entity: DYNAMIC-PART-ALTERNATIVE

Target model property	Source model property
I-PDU-REF	I-PDU-REF
SELECTOR-FIELD-CODE	SELECTOR-FIELD-CODE
INITIAL-DYNAMIC-PART	INITIAL-DYNAMIC-PART

NOTE
The source model differs from the EB tresos Studio system model


The EB tresos Studio internal system model differs from the source model in how MULTIPLEXED-I-PDUS are stored, see [Section 8.1.1.2.23, "MULTIPLEXED-I-PDU"](#).

8.1.1.2.26. STATIC-PART

Source model entity: STATIC-PART

Target model entity: STATIC-PART

Target model property	Source model property
I-PDU-REF	I-PDU-REF
SEGMENT-POSITIONS/SEGMENT-POSITION	SEGMENT-POSITIONS/SEGMENT-POSITION

NOTE
The source model differs from the EB tresos Studio system model


The EB tresos Studio internal system model differs from the source model in how MULTIPLEXED-I-PDUS are stored, see [Section 8.1.1.2.23, "MULTIPLEXED-I-PDU"](#).

8.1.1.2.27. SEGMENT-POSITION

Source model entity: SEGMENT-POSITION

Target model entity: SEGMENT-POSITION

Target model property	Source model property
SEGMENT-BYTE-ORDER	SEGMENT-BYTE-ORDER
SEGMENT-LENGTH	SEGMENT-LENGTH
SEGMENT-POSITION	SEGMENT-POSITION

**NOTE****The source model differs from the EB tresos Studio system model**

The EB tresos Studio internal system model differs from the source model in how MULTIPLEXED-I-PDUS are stored, see [Section 8.1.1.2.23, "MULTIPLEXED-I-PDU".](#)

8.1.1.2.28. N-PDU

Source model entity: N-PDU

Target model entity: N-PDU

Target model property	Source model property
CATEGORY	CATEGORY
LENGTH	LENGTH

8.1.1.2.28.1. Changed or moved parameters

LENGTH

During the upgrade process, the parameter LENGTH given in units of bits in the source model is converted to bytes as required by the target model.

8.1.1.2.29. NM-PDU

Source model entity: NM-PDU

Target model entity: NM-PDU

Target model property	Source model property
CATEGORY	CATEGORY
LENGTH	LENGTH
UNUSED-BIT-PATTERN	UNUSED-BIT-PATTERN

If a Fibex file has been imported into the EB tresos Studio source model, the following target model parameters are set additionally:

Target model property	Fibex model property
NM-DATA-INFORMATION	NM-PDU-USAGE/CONTAINS-DATA
NM-VOTE-INFORMATION	NM-PDU-USAGE/CONTAINS-VOTE
I-SIGNAL-TO-I-PDU-MAPPINGS/I-SIGNAL-TO-I-PDU-MAPPING	PDU/SIGNAL-INSTANCES



8.1.1.2.29.1. Changed or moved parameters

LENGTH

During the conversion process, the parameter LENGTH given in units of bits in the source model is converted to bytes as required by the target model.

NM-CBV-POSITION, NM-NID-POSITION, and NM-USER-DATA-LENGTH

The source model parameters NM-CBV-POSITION, NM-NID-POSITION, and NM-USER-DATA-LENGTH are used to configure the target model entity CAN-NM-CLUSTER, see [Section 8.1.1.7.3, “CAN-NM-CLUSTER”](#).

8.1.1.2.30. PDU-TO-FRAME-MAPPING

Source model entity: PDU-TO-FRAME-MAPPING

Target model entity: PDU-TO-FRAME-MAPPING

Target model property	Source model property
CATEGORY	CATEGORY
PACKING-BYTE-ORDER	PACKING-BYTE-ORDER
PDU-REF	PDU-REF
START-POSITION	START-POSITION
UPDATE-INDICATION-BIT-POSITION	UPDATE-INDICATION-BIT-POSITION

8.1.1.2.31. I-PDU-TIMING

Source model entity: I-PDU-TIMING

Target model entity: I-PDU-TIMING

Target model property	Source model property
MINIMUM-DELAY	MINIMUM-DELAY
TRANSMISSION-MODE-DECLARATION	TRANSMISSION-MODE-DECLARATION
No target model parameter is available.	REQUEST-CONTROLLED-TIMING

8.1.1.2.31.1. Changed or moved parameters

I-PDU-TIMING

In the target model, one TRANSMISSION-MODE-DECLARATION is added to the I-PDU-TIMING, even if no counterpart exists in the source model. This newly created TRANSMISSION-MODE-DECLARATION enti-



ty aggregates via TRANSMISSION-MODE-TRUE-TIMING the parameters EVENT-CONTROLLED-TIMING and CYCLIC-TIMING, which are aggregated directly in the source model I-PDU-TIMING.

8.1.1.2.32. I-SIGNAL-I-PDU-GROUP

Source model entity: I-PDU-GROUP

Target model entity: I-SIGNAL-I-PDU-GROUP

Target model property	Source model property
COMMUNICATION-DIRECTION	COMMUNICATION-DIRECTION
COMMUNICATION-MODE	COMMUNICATION-MODE
CONTAINED-I-SIGNAL-I-PDU-GROUP-REFS/CONTAINED-I-SIGNAL-I-PDU-GROUP-REF	CONTAINED-I-PDU-GROUPS-REFS/CONTAINED-I-PDU-GROUPS-REF
I-SIGNAL-I-PDUS/I-SIGNAL-I-PDU-REF-CONDITIONAL	I-PDU-REFS/I-PDU-REF

8.1.1.2.33. I-SIGNAL-TO-I-PDU-MAPPING

Source model entity: I-SIGNAL-TO-I-PDU-MAPPING

Target model entity: I-SIGNAL-TO-I-PDU-MAPPING

Target model property	Source model property
CATEGORY	CATEGORY
I-SIGNAL-GROUP-REF	SIGNAL-REF
I-SIGNAL-REF	SIGNAL-REF
PACKING-BYTE-ORDER	PACKING-BYTE-ORDER
START-POSITION	START-POSITION
TRANSFER-PROPERTY	TRANSFER-PROPERTY
UPDATE-INDICATION-BIT-POSITION	UPDATE-INDICATION-BIT-POSITION

8.1.1.2.33.1. Changed or moved parameters

SIGNAL-REF

The source model parameter SIGNAL-REF is upgraded to the target model parameter I-SIGNAL-REF if the referenced source model entity represents an I-SIGNAL that in turn references a SYSTEM-SIGNAL. If



the referenced I-SIGNAL references a SYSTEM-SIGNAL-GROUP, the parameter is upgraded to I-SIGNAL-GROUP-REF.

TRANSFER-PROPERTY

If a Fibex 3.x file has been imported into the EB tresos Studio source model, TRANSFER-PROPERTY is set according to its representation in Fibex files. Fibex files currently only support TRIGGERED-ON-CHANGE as parameter value, or no value at all.

8.1.1.2.34. I-SIGNAL

Source model entity: I-SIGNAL

Target model entity: I-SIGNAL

Target model property	Source model property
CATEGORY	CATEGORY
SYSTEM-SIGNAL-REF	SYSTEM-SIGNAL-REF

8.1.1.2.34.1. Changed or moved parameters

I-SIGNAL

If the source model I-SIGNAL references a SYSTEM-SIGNAL via SYSTEM-SIGNAL-REF, an I-SIGNAL is created in the target model.

LENGTH

The target model parameter LENGTH is retrieved from the SYSTEM-SIGNAL referenced via SYSTEM-SIGNAL-REF in the source model.

DATA-TYPE-POLICY

The value of the target model parameter DATA-TYPE-POLICY is set to the value LEGACY.

INIT-VALUE

The target model parameter INIT-VALUE is configured by retrieving the SYSTEM-SIGNAL of the source model I-SIGNAL, and then obtaining the entity referenced via INIT-VALUE-REF, which can be one of the following types:

- ▶ ARRAY-SPECIFICATION
- ▶ BOOLEAN-LITERAL
- ▶ CHAR-LITERAL
- ▶ CONSTANT-REFERENCE
- ▶ INTEGER-LITERAL



- ▶ OPAQUE-LITERAL
- ▶ REAL-LITERAL
- ▶ RECORD-SPECIFICATION, or
- ▶ STRING-LITERAL

If the entity is of the type STRING-LITERAL, the content of its VALUE parameter is upgraded to a TEXT-VALUE-SPECIFICATION. In all other cases, the content of VALUE is upgraded to a NUMERICAL-VALUE-SPECIFICATION.

NETWORK-REPRESENTATION-PROPS

For the configuration of the target model parameter NETWORK-REPRESENTATION-PROPS, the data type of the source SYSTEM-SIGNAL, referenced via DATA-TYPE-REF is retrieved. If this data type contains a SW-DATA-DEF-PROPS/INVALID-VALUE parameter, a NETWORK-REPRESENTATION-PROPS parameter is created for the target model entity I-SIGNAL. It contains in SW-DATA-DEF-PROPS-CONDITIONAL/IMPLEMENTATION-DATA-TYPE-REF and SW-DATA-DEF-PROPS-CONDITIONAL/BASE-TYPE-REF references to the upgraded data type entities in the target model. Moreover, the parameter INVALID-VALUE is configured. If the source model SW-DATA-DEF-PROPS/INVALID-VALUE contains a STRING-LITERAL, this is upgraded to an TEXT-VALUE-SPECIFICATION. In all other cases it is upgraded to a NUMERICAL-VALUE-SPECIFICATION.

If the IMPLEMENTATION-DATA-TYPE that is referenced in SW-DATA-DEF-PROPS-CONDITIONAL/IMPLEMENTATION-DATA-TYPE-REF owns a reference to a COMPU-METHOD in SW-DATA-DEF-PROPS/SW-DATA-DEF-PROPS-VARIANTS/SW-DATA-DEF-PROPS-CONDITIONAL/COMPU-METHOD-REF, this COMPU-METHOD reference is also set in NETWORK-REPRESENTATION-PROPS/SW-DATA-DEF-PROPS-VARIANTS/SW-DATA-DEF-PROPS-CONDITIONAL/COMPU-METHOD-REF.

8.1.1.2.35. I-SIGNAL-GROUP

Source model entity: I-SIGNAL

Target model entity: I-SIGNAL-GROUP

Target model property	Source model property
CATEGORY	CATEGORY
SYSTEM-SIGNAL-REF	SYSTEM-SIGNAL-REF

8.1.1.2.35.1. Changed or moved parameters

I-SIGNAL

If the source model I-SIGNAL references a SYSTEM-SIGNAL-GROUP via SYSTEM-SIGNAL-REF, an I-SIGNAL-GROUP is created in the target model.

**I-SIGNAL-GROUP**

The parameters INIT-VALUE and NETWORK-REPRESENTATION-PROPS are not configured for target model I-SIGNAL-GROUPS.

I-SIGNAL-REFS/I-SIGNAL-REF

I-SIGNAL-REFS/I-SIGNAL-REF references all target model I-SIGNALS belonging to the I-SIGNAL-GROUP. An I-SIGNAL belongs to an I-SIGNAL-GROUP if the SYSTEM-SIGNAL of the I-SIGNAL also belongs to the SYSTEM-SIGNAL-GROUP of the I-SIGNAL-GROUP. The SYSTEM-SIGNAL of the I-SIGNAL and the SYSTEM-SIGNAL-GROUP of the I-SIGNAL-GROUP are both referenced via SYSTEM-SIGNAL-REF. A SYSTEM-SIGNAL-GROUP references its SYSTEM-SIGNALS in SYSTEM-SIGNAL-REFS/SYSTEM-SIGNAL-REF.

8.1.1.2.36. SYSTEM-SIGNAL

Source model entity: SYSTEM-SIGNAL

Target model entity: SYSTEM-SIGNAL

Target model property	Source model property
CATEGORY	CATEGORY

8.1.1.2.36.1. Changed or moved parameters**DATA-TYPE-REF**

The source model parameter DATA-TYPE-REF is configured in the target model I-SIGNAL entity parameter NETWORK-REPRESENTATION-PROPS/SW-DATA-DEF-PROPS-CONDITIONAL/IMPLEMENTATION-DATA-TYPE-REF, and NETWORK-REPRESENTATION-PROPS/SW-DATA-DEF-PROPS-CONDITIONAL/BASE-TYPE-REF, see [Section 8.1.1.2.34, "I-SIGNAL"](#).

LENGTH

The source model parameter LENGTH is configured in the target model I-SIGNAL entity parameter LENGTH, see [Section 8.1.1.2.34, "I-SIGNAL"](#).

DYNAMIC-LENGTH

The value of the target model parameter DYNAMIC-LENGTH is set to the value false.

8.1.1.2.37. I-SIGNAL-PORT

Source model entity: SIGNAL-PORT

Target model entity: I-SIGNAL-PORT



Target model property	Source model property
CATEGORY	CATEGORY
COMMUNICATION-DIRECTION	COMMUNICATION-DIRECTION
TIMEOUT	TIMEOUT

8.1.1.2.38. FRAME-PORT

Source model entity: FRAME-PORT

Target model entity: FRAME-PORT

Target model property	Source model property
CATEGORY	CATEGORY
COMMUNICATION-DIRECTION	COMMUNICATION-DIRECTION

8.1.1.2.39. I-PDU-PORT

Source model entity: I-PDU-PORT

Target model entity: I-PDU-PORT

Target model property	Source model property
CATEGORY	CATEGORY
COMMUNICATION-DIRECTION	COMMUNICATION-DIRECTION

8.1.1.2.40. TRANSMISSION-MODE-DECLARATION

Source model entity: TRANSMISSION-MODE-DECLARATION

Target model entity: TRANSMISSION-MODE-DECLARATION

Target model property	Source model property
TRANSMISSION-MODE-CONDITIONS / TRANSMISSION-MODE-CONDITION	TRANSMISSION-MODE-CONDITIONS / TRANSMISSION-MODE-CONDITION
TRANSMISSION-MODE-FALSE-TIMING	TRANSMISSION-MODE-FALSE-TIMING



8.1.1.2.40.1. Changed or moved parameters

TRANSMISSION-MODE-DECLARATION **and** an TRANSMISSION-MODE-TRUE-TIMING

A TRANSMISSION-MODE-DECLARATION **and** a TRANSMISSION-MODE-TRUE-TIMING are always added to the aggregating I-PDU-TIMING, see [Section 8.1.1.2.31, “I-PDU-TIMING”](#) for details.

If the TRANSMISSION-MODE-DECLARATION has been newly created, one TRANSMISSION-MODE-CONDITIONS/TRANSMISSION-MODE-CONDITION is added, containing a DATA-FILTER of type ALWAYS.

8.1.1.2.41. TRANSMISSION-MODE-CONDITION

Source model entity: TRANSMISSION-MODE-CONDITION

Target model entity: TRANSMISSION-MODE-CONDITION

Target model property	Source model property
DATA-FILTER	DATA-FILTER
I-SIGNAL-IN-I-PDU-REF	SIGNAL-IN-I-PDU-REF

8.1.1.2.42. TRANSMISSION-MODE-FALSE-TIMING

Source model entity: TRANSMISSION-MODE-FALSE-TIMING

Target model entity: TRANSMISSION-MODE-TIMING

Target model property	Source model property
CYCLIC-TIMING	CYCLIC-TIMING
EVENT-CONTROLLED-TIMING	EVENT-CONTROLLED-TIMING

8.1.1.2.43. EVENT-CONTROLLED-TIMING

Source model entity: EVENT-CONTROLLED-TIMING

Target model entity: EVENT-CONTROLLED-TIMING

Target model property	Source model property
NUMBER-OF-REPETITIONS	NUMBER-OF-REPEATS
REPETITION-PERIOD	REPETITION-PERIOD



8.1.1.2.44. CYCLIC-TIMING

Source model entity: CYCLIC-TIMING

Target model entity: CYCLIC-TIMING

Target model property	Source model property
TIME-OFFSET	STARTING-TIME
TIME-PERIOD	REPEATING-TIME
No target model parameter is available.	FINAL-REPETITIONS

8.1.1.2.45. TIME-RANGE-TYPE

Source model entity: TIME-RANGE-TYPE

Target model entity: TIME-RANGE-TYPE

Target model property	Source model property
TOLERANCE/ABSOLUTE-TOLERANCE	TOLERANCE/ABSOLUTE-TOLERANCE
TOLERANCE/RELATIVE-TOLERANCE	TOLERANCE/RELATIVE-TOLERANCE

8.1.1.2.46. ABSOLUTE-TOLERANCE

Source model entity: ABSOLUTE-TOLERANCE

Target model entity: ABSOLUTE-TOLERANCE

Target model property	Source model property
ABSOLUTE	ABSOLUTE

8.1.1.2.47. RELATIVE-TOLERANCE

Source model entity: RELATIVE-TOLERANCE

Target model entity: RELATIVE-TOLERANCE

Target model property	Source model property
RELATIVE	RELATIVE



8.1.1.3. Gateway

8.1.1.3.1. FRAME-MAPPING

Source model entity: FRAME-MAPPING

Target model entity: FRAME-MAPPING

Target model property	Source model property
SOURCE-FRAME-REF	SOURCE-FRAME-REF
TARGET-FRAME-REF	TARGET-FRAME-REF

8.1.1.3.2. I-PDU-MAPPING

Source model entity: I-PDU-MAPPING

Target model entity: I-PDU-MAPPING

Target model property	Source model property
SOURCE-I-PDU-REF	SOURCE-I-PDU/SOURCE-I-PDU-REF
TARGET-I-PDU/TARGET-I-PDU-REF	TARGET-I-PDU/TARGET-I-PDU-REF
No target model parameter is available.	SOURCE-I-PDU-REF/PDU-MAPPINGS-DE-FAULT-VALUE

8.1.1.3.3. TARGET-I-PDU-REF

Source model entity: TARGET-I-PDU-REF

Target model entity: TARGET-I-PDU-REF

Target model property	Source model property
DEFAULT-VALUE	DEFAULT-VALUE
TARGET-I-PDU-REF	TARGET-I-PDU-REF

8.1.1.3.4. PDU-MAPPING-DEFAULT-VALUE

Source model entity: PDU-MAPPING-DEFAULT-VALUE

Target model entity: PDU-MAPPING-DEFAULT-VALUE



Target model property	Source model property
DEFAULT-VALUE-ELEMENTS/DEFAULT-VALUE-ELEMENT	DEFAULT-VALUE-ELEMENTS/DEFAULT-VALUE-ELEMENT

8.1.1.3.5. DEFAULT-VALUE-ELEMENT

Source model entity: DEFAULT-VALUE-ELEMENT

Target model entity: DEFAULT-VALUE-ELEMENT

Target model property	Source model property
ELEMENT-BYTE-VALUE	ELEMENT-BYTE-VALUE
ELEMENT-POSITION	ELEMENT-POSITION

8.1.1.3.6. I-SIGNAL-MAPPING

Source model entity: SIGNAL-MAPPING

Target model entity: I-SIGNAL-MAPPING

Target model property	Source model property
SOURCE-SIGNAL-REF	SOURCE-SIGNAL-REF
TARGET-SIGNAL-REF	TARGET-SIGNAL-REF

8.1.1.4. CAN Transport Layer

8.1.1.4.1. CAN-TP-CONFIG

For each CAN-CLUSTER in the source model that has at least one CAN-TP-CONNECTION-CHANNEL aggregated in its PHYSICAL-CHANNEL, one CAN-TP-CONFIG element is created in the target model. It resides in the same AR-PACKAGE as the related CAN-CLUSTER.

8.1.1.4.1.1. New parameters in the target model

CAN-TP-ECU

For each ECU-INSTANCE in the source model that is attached to the PHYSICAL-CHANNEL, one CAN-TP-ECU element is created in the target model. It references the target model counterpart of the source model



ECU-INSTANCE in ECU-INSTANCE-REF. All created CAN-TP-ECU elements are aggregated in CAN-TP-CONFIG/TP-ECUS.

CAN-TP-ECU/CYCLE-TIME-MAIN-FUNCTION

If a Fibex file has been imported into the EB tresos Studio source model, CAN-TP-ECU/CYCLE-TIME-MAIN-FUNCTION is additionally set in the target model using the Fibex parameter TP-ECU/CYCLE-TIME-MAIN-FUNCTION.

CAN-TP-NODE

For each source model COMMUNICATION-CONNECTOR attached to a PHYSICAL-CHANNEL of a CAN-CLUSTER for which a CAN-TP-CONFIG has been created in the target model, one CAN-TP-NODE is added. In CONNECTOR-REF, CAN-TP-NODE references the COMMUNICATION-CONNECTOR counterpart in the target model. CAN-TP-NODE is aggregated in CAN-TP-CONFIG/TP-NODES. Its parameter ST-MIN is set to the maximum MINIMUM-SEPARATION-TIME value of all CAN-TP-CONNECTIONS the CAN-TP-NODE is receiving.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional CAN-TP-NODE parameter values are set:

Target model property	Fibex model property
MAX-FC-WAIT	TP-NODE/MAX-FC-WAIT
ST-MIN	TP-NODE/STMIN
TIMEOUT-AR	TP-NODE/TIMEOUT-AR
TIMEOUT-AS	TP-NODE/TIMEOUT-AS

CAN-TP-ADDRESS

An additional entity CAN-TP-ADDRESS is created, which the CAN-TP-NODE references via TP-ADDRESS-REF. The CAN-TP-ADDRESS contains in its TP-ADDRESS parameter the value obtained from the COMMUNICATION-CONNECTOR parameter TP-ADDRESS. It is aggregated in CAN-TP-CONFIG/TP-ADDRESSS.

8.1.1.4.1.2. Changed or moved parameters

TP-NODE/STMIN

TP-NODE/STMIN is upgraded from its representation according to the ISO 15765-2 specification as given in Fibex to seconds as required by the target model.

8.1.1.4.2. CAN-TP-CONNECTION

Source model entity: CAN-TP-CONNECTION-CHANNEL



Target model entity: CAN-TP-CONNECTION

Target model property	Source model property
ADDRESSING-FORMAT	ADDRESSING-FORMAT
DATA-PDU-REF	DATA-PDU-REF
FLOW-CONTROL-PDU-REF	FLOW-CONTROL-PDU-REF
MAX-BLOCK-SIZE	BLOCK-SIZE
PADDING-ACTIVATION	PADDING-ACTIVATION
TP-SDU-REF	TP-SDU-REF
Not upgraded.	MULTICAST-ADDRESSING
MULTICAST-REF	No source model parameter is available.

The created CAN-TP-CONNECTION is aggregated in CAN-TP-CONFIG/TP-CONNECTIONS.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional CAN-TP-CONNECTION parameter values are set:

Target model property	Fibex model property
TIMEOUT-BS	TP-CHANNEL/TIMEOUT-BS
TIMEOUT-CR	TP-CHANNEL/TIMEOUT-CR
TRANSMIT-CANCELLATION	TP-CHANNEL/TRANSMIT-CANCELLATION
MULTICAST-REF	TP-CHANNEL/TP-CONNECTIONS/TP-CONNECTION/TP-MULTICAST-CONNECTION/
No target model parameter is available.	MULTICAST-ADDRESSING

8.1.1.4.2.1. New parameters in the target model

TRANSMITTER-REF

The parameter TRANSMITTER-REF references the CAN-TP-NODE that has been created in the target model for the COMMUNICATION-CONNECTOR, which is referenced by SOURCE-REF in the source model.

RECEIVER-REFS/RECEIVER-REF

The parameter RECEIVER-REFS/RECEIVER-REF references the CAN-TP-NODE that has been created in the target model for the COMMUNICATION-CONNECTOR referenced by TARGET-REF in the source model.

CAN-TP-CHANNEL

For every created CAN-TP-CONNECTION, one additional CAN-TP-CHANNEL is created, which the CAN-TP-CONNECTION references in CAN-TP-CHANNEL-REF. The CHANNEL-MODE of the CAN-TP-CHANNEL is set to HALF-DUPLEX-MODE. It is aggregated in CAN-TP-CONFIG/TP-CHANNELS.



TP-CHANNEL/CHANNEL-ID

If a Fibex file has been imported into the EB tresos Studio source model, CAN-TP-CHANNEL/CHANNEL-ID is set additionally to its Fibex counterpart TP-CHANNEL/CHANNEL-ID.

8.1.1.4.2.2. Changed or moved parameters

MINIMUM-SEPARATION-TIME

The parameter MINIMUM-SEPARATION-TIME is upgraded to the receiver CAN-TP-NODE, see [Section 8.1.1.4.1, “CAN-TP-CONFIG”](#).

8.1.1.5. FlexRay Transport Layer**8.1.1.5.1. FLEXRAY-TP-CONFIG**

For each FLEXRAY-CLUSTER in the source model that contains at least one FLEX-RAY-TP-CHANNEL in one of its PHYSICAL-CHANNELS, one FLEXRAY-TP-CONFIG element is created in the target model. It resides in the same AR-PACKAGE as the related FLEXRAY-CLUSTER.

8.1.1.5.1.1. New parameters in the target model

FLEXRAY-TP-ECU

For each ECU-INSTANCE in the source model which is attached to one of the PHYSICAL-CHANNELS, one FLEXRAY-TP-ECU element is created in the target model. In ECU-INSTANCE-REF it references the target model counterpart of the source model ECU-INSTANCE. All created FLEXRAY-TP-ECU elements are aggregated in FLEXRAY-TP-CONFIG/TP-ECUS.

FLEXRAY-TP-ECU/CYCLE-TIME-MAIN-FUNCTION

If a Fibex file has been imported into the EB tresos Studio source model, FLEXRAY-TP-ECU/CYCLE-TIME-MAIN-FUNCTION is additionally set using the Fibex parameter TP-ECU/CYCLE-TIME-MAIN-FUNCTION.

FLEXRAY-TP-NODE

For each source model COMMUNICATION-CONNECTOR attached to any PHYSICAL-CHANNEL of a FLEXRAY-CLUSTER for which a FLEXRAY-TP-CONFIG has been created in the target model, one FLEXRAY-TP-NODE is added. In CONNECTOR-REFS/CONNECTOR-REF, it references the COMMUNICATION-CONNECTOR counterpart in the target model and is aggregated in FLEXRAY-TP-CONFIG/TP-NODES.

TP-ADDRESS

An additional entity TP-ADDRESS is created which the FLEXRAY-TP-NODE references via TP-ADDRESS-REF. The TP-ADDRESS contains in its TP-ADDRESS parameter the value obtained from the COM-



MUNICATION-CONNECTOR parameter TP-ADDRESS. It is aggregated in FLEXRAY-TP-CONFIG/TP-AD-DRESSS.

8.1.1.5.2. FLEXRAY-TP-CONNECTION

Source model entity: FLEX-RAY-TP-CONNECTION

Target model entity: FLEXRAY-TP-CONNECTION

Target model property	Source model property
DIRECT-TP-SDU-REF	DIRECT-TP-SDU-REF
REVERSED-TP-SDU-REF	REVERSED-TP-SDU-REF
BANDWIDTH-LIMITATION	No source model parameter is available.

The created FLEXRAY-TP-CONNECTION is aggregated in FLEXRAY-TP-CONFIG/TP-CONNECTIONS.

8.1.1.5.2.1. New parameters in the target model

FLEXRAY-TP-PDU-POOL

For every created FLEXRAY-TP-CONNECTION, one additional FLEXRAY-TP-PDU-POOL is created and aggregated in FLEXRAY-TP-CONFIG/PDU-POOLS. The FLEXRAY-TP-CONNECTION references two FLEXRAY-TP-PDU-POOLS in TX-PDU-POOL-REF and RX-PDU-POOL-REF.

Via RX-PDU-POOL-REF, FLEXRAY-TP-PDU-POOL/N-PDU-REFS references all target model counter-parts of the source model N-PDUS that the ECU-INSTANCE receives and that are referenced in the source model either in FLEX-RAY-TP-CONNECTION/FLOW-CONTROL-PDU-REF or in FLEX-RAY-TP-CONNECTION/TRANSMIT-PDU-REFS/TRANSMIT-PDU-REF.

Via TX-PDU-POOL-REF, FLEXRAY-TP-PDU-POOL/N-PDU-REFS references all target model counter-parts of the source model N-PDUS that the ECU-INSTANCE sends and that are referenced in the source model either in FLEX-RAY-TP-CONNECTION/FLOW-CONTROL-PDU-REF or in FLEX-RAY-TP-CONNECTION/TRANSMIT-PDU-REFS/TRANSMIT-PDU-REF.

TRANSMITTER-REF

The parameter TRANSMITTER-REF references the FLEXRAY-TP-NODE that has been created in the target model for the COMMUNICATION-CONNECTOR referenced by SOURCE-REF in the source model.

RECEIVER-REFS/RECEIVER-REF

The parameter RECEIVER-REFS/RECEIVER-REF references the FLEXRAY-TP-NODE that has been created in the target model for the COMMUNICATION-CONNECTOR referenced by TARGET-REF in the source model.



FLEXRAY-TP-CONNECTION/MULTICAST-REF

If a Fibex file has been imported into the EB tresos Studio source model, FLEXRAY-TP-CONNECTION/MULTICAST-REF is set according to the content of the Fibex parameter TP-CHANNEL/TP-CONNECTIONS/TP-CONNECTION/TP-MULTICAST-CONNECTION.

8.1.1.5.3. FLEXRAY-TP-CONNECTION-CONTROL

Source model entity: FLEX-RAY-TP-CHANNEL

Target model entity: FLEXRAY-TP-CONNECTION-CONTROL

Target model property	Source model property
ACK-TYPE	ACK-TYPE
No target model parameter is available.	EXTENDED-ADDRESSING
No target model parameter is available.	MAXIMUM-MESSAGE-LENGTH
No target model parameter is available.	MINIMUM-SEPARATION-TIME
No target model parameter is available.	MULTICAST-SEGMENTATION
No target model parameter is available.	PDU-POOLS
No target model parameter is available.	TIME-CS
No target model parameter is available.	TRANSMIT-CANCELLATION

The created FLEXRAY-TP-CONNECTION-CONTROL is aggregated in FLEXRAY-TP-CONFIG/TP-CONNECTION-CONTROLS.

In the source model, the FLEX-RAY-TP-CHANNEL aggregates all its FLEX-RAY-TP-CONNECTION elements. In the target model, the FLEXRAY-TP-CONNECTION elements reference the FLEXRAY-TP-CONNECTION-CONTROL they belong to via TP-CONNECTION-CONTROL-REF.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional FLEXRAY-TP-CONNECTION-CONTROL parameter values are set:

Target model property	Fibex model property
MAX-BUFFER-SIZE	TP-CHANNEL/MAX-BLOCK-SIZE
TIMEOUT-BS	TP-CHANNEL/TIMEOUT-BS
MAX-RETRIES	TP-CHANNEL/MAX-RETRIES
TIMEOUT-CR	TP-CHANNEL/TIMEOUT-CR
TIMEOUT-AR	max (TP-NODE/TIMEOUT-AR)
TIMEOUT-AS	max (TP-NODE/TIMEOUT-AS)
MAX-AR	max (TP-NODE/MAX-AR)



Target model property	Fibex model property
MAX-AS	max(TP-NODE/MAX-AS)
MAX-FR-IF	max(TP-NODE/MAX-FRIF)
TIME-BUFFER	max(TP-NODE/TIME-BUFFER)
TIME-FR-IF	max(TP-NODE/TIME-FRIF)
MAX-FC-WAIT	max(TP-NODE/BUFFER-REQUEST)

For the configuration of TIMEOUT-AR, TIMEOUT-AS, MAX-AR, MAX-AS, MAX-FR-IF, TIME-BUFFER, TIME-FR-IF, MAX-FC-WAIT the maximum of the respective Fibex parameter values of all TP-NODES connected to the TP-CHANNEL is taken.

8.1.1.5.3.1. New parameters in the target model

FLEX-RAY-TP-CONNECTION

Every target model counterpart of the source model FLEX-RAY-TP-CONNECTION which is aggregated in FLEX-RAY-TP-CHANNEL/TP-CONNECTIONS/FLEX-RAY-TP-CONNECTION, references the FLEXRAY-TP-CONNECTION-CONTROL element in its parameter TP-CONNECTION-CONTROL-REF.

8.1.1.6. LIN Transport Layer

8.1.1.6.1. LIN-TP-CONFIG

For each LIN-CLUSTER in the source model that contains at least one LIN-TP-CHANNEL in its PHYSICAL-CHANNEL, one LIN-TP-CONFIG element is created in the target model. It resides in the same AR-PAGE as the related LIN-CLUSTER.

8.1.1.6.1.1. New parameters in the target model

LIN-TP-NODE

For each source model COMMUNICATION-CONNECTOR attached to a PHYSICAL-CHANNEL of a LIN-CLUSTER for which a LIN-TP-CONFIG has been created in the target model, one LIN-TP-NODE is added. In CONNECTOR-REF, it references the COMMUNICATION-CONNECTOR counterpart in the target model. It is aggregated in LIN-TP-CONFIG/TP-NODES.

Target model property	Source model property
MAX-NUMBER-OF-RESP-PENDING-FRAMES	No source model parameter is available.
P-2-MAX	No source model parameter is available.



Target model property	Source model property
P-2-TIMING	No source model parameter is available.

TP-ADDRESS

An additional entity TP-ADDRESS is created which the LIN-TP-NODE references via TP-ADDRESS-REF. The LIN-TP-ADDRESS contains in its TP-ADDRESS parameter the value obtained from the COMMUNICATION-CONNECTOR parameter TP-ADDRESS. It is aggregated in LIN-TP-CONFIG/TP-ADDRESSSS.

8.1.1.6.2. LIN-TP-CONNECTION

Source model entity: LIN-TP-CHANNEL

Target model entity: LIN-TP-CONNECTION

Target model property	Source model property
DATA-PDU-REF	DATA-PDU-REF
FLOW-CONTROL-REF	FLOW-CONTROL-REF
LIN-TP-N-SDU-REF	LIN-TP-N-SDU-REF
No target model parameter is available..	MULTICAST-ADDRESSING
MULTICAST-REF	No source model parameter is available.

The created LIN-TP-CONNECTION is aggregated in LIN-TP-CONFIG/TP-CONNECTIONS.

8.1.1.6.2.1. New parameters in the target model

TRANSMITTER-REF

The parameter TRANSMITTER-REF references the LIN-TP-NODE that has been created in the target model for the COMMUNICATION-CONNECTOR referenced by SOURCE-REF in the source model.

RECEIVER-REFS/RECEIVER-REF

The parameter RECEIVER-REFS/RECEIVER-REF references the LIN-TP-NODE that has been created in the target model for the COMMUNICATION-CONNECTOR referenced by TARGET-REF in the source model.

8.1.1.7. Network Management

8.1.1.7.1. NM-CONFIG

For every SYSTEM element in the source model, one NM-CONFIG element is created, which resides in the same AR-PACKAGE as the related SYSTEM.



8.1.1.7.2. NM-ECU

For every ECU-INSTANCE in the source model, one NM-ECU instance is created in the target model. The related ECU-INSTANCE is referenced in ECU-INSTANCE-REF.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional NM-ECU parameter values are set:

Target model property	Fibex model property
NM-REMOTE-SLEEP-IND-ENABLED	NM-ECU/REMOTE-SLEEP-INDICATION-ENABLED
NM-PASSIVE-MODE-ENABLED	NM-ECU/PASSIVE-MODE-ENABLED
NM-BUS-SYNCHRONIZATION-ENABLED	NM-ECU/BUS-SYNCHRONIZATION-ENABLED
NM-REPEAT-MSG-IND-ENABLED	NM-ECU/NM-ECU-EXTENSIONS/NM-ECU-EXTENSION/REPEAT-MSG-INDICATION-ENABLED
NM-MULTIPLE-CHANNELS-ENABLED	NM-ECU/MULTIPLE-CHANNELS-ENABLED
NM-COM-CONTROL-ENABLED	NM-ECU/COM-CONTROL-ENABLED
NM-STATE-CHANGE-IND-ENABLED	NM-ECU/STATE-CHANGE-INDICATION-ENABLED
NM-PDU-RX-INDICATION-ENABLED	NM-ECU/PDU-RX-INDICATION-ENABLED

NOTE


NM-REPEAT-MSG-IND-ENABLED is only configured for NM-ECUs connected to CAN-NM-CLUSTERS

The parameter NM-REPEAT-MSG-IND-ENABLED is only configured for NM-ECUs connected to CAN-NM-CLUSTERS.

8.1.1.7.2.1. New parameters in the target model

NM-NODE-DETECTION-ENABLED

NM-NODE-DETECTION-ENABLED is configured by taking the maximum of NM-NODE-DETECTION-ENABLED of any FLEXRAY-COMMUNICATION-CONTROLLER of the ECU-INSTANCE (supported by AUTOSAR 3.1 rev 0 system description file imports only). If none of these FLEXRAY-COMMUNICATION-CONTROLLER elements has the value set, the maximum NM-NODE-DETECTION-ENABLED value of all COMMUNICATION-CLUSTER elements to which the ECU-INSTANCE is connected, is taken.

NM-NODE-ID-ENABLED

NM-NODE-ID-ENABLED is determined by taking the maximum NM-NODE-ID-ENABLED value of all COMMUNICATION-CLUSTER elements to which the ECU-INSTANCE is connected (supported by AUTOSAR 3.1 rev 0 system description file imports only).

NM-USER-DATA-ENABLED

NM-USER-DATA-ENABLED is configured by taking the maximum NM-USER-DATA-ENABLED value of all CAN-COMMUNICATION-CONTROLLERS and FLEXRAY-COMMUNICATION-CONTROLLERS to which the ECU-INSTANCE is connected.



8.1.1.7.3. CAN-NM-CLUSTER

Source model entity: CAN-CLUSTER

Target model entity: CAN-NM-CLUSTER

Target model property	Source model property
NM-BUSLOAD-REDUCTION-ACTIVE	NM-BUS-LOAD-REDUCTION-ACTIVE
NM-NETWORK-TIMEOUT	NM-TIMEOUT-TIME
NM-REPEAT-MESSAGE-TIME	NM-REPEAT-MESSAGE-STATE-TIME
NM-REMOTE-SLEEP-INDICATION-TIME	NM-REMOTE-SLEEP-INDICATION-TIME
NM-WAIT-BUS-SLEEP-TIME	NM-WAIT-BUS-SLEEP-TIME
NM-CHANNEL-SLEEP-MASTER	No source model parameter is available.
NM-IMMEDIATE-NM-CYCLE-TIME	No source model parameter is available.
NM-IMMEDIATE-NM-TRANSMISSIONS	No source model parameter is available.

The created CAN-NM-CLUSTER is aggregated in NM-CONFIG/NM-CLUSTERS. It references the related CAN-CLUSTER in COMMUNICATION-CLUSTER-REF.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional CAN-NM-CLUSTER parameter values are set:

Target model property	Fibex model property
NM-CHANNEL-ID	NM-CLUSTER/CHANNEL-ID
NM-CHANNEL-ACTIVE	NM-CLUSTER/CHANNEL-ACTIVE
NM-MESSAGE-TIMEOUT-TIME	NM-CLUSTER/MSG-TIMEOUT-TIME

8.1.1.7.3.1. New parameters in the target model

NM-MSG-CYCLE-TIME and NM-REMOTE-SLEEP-INDICATION-TIME

The EB tresos Studio internal system model additionally configures the parameters NM-MSG-CYCLE-TIME and NM-REMOTE-SLEEP-INDICATION-TIME if the source model has been imported from a 3.1 rev 0 AUTOSAR system description file which provides the corresponding source parameters in CAN-CLUSTER/NM-REMOTE-SLEEP-INDICATION-TIME and CAN-CLUSTER/NM-MSG-CYCLE-TIME.

NM-CBV-POSITION, NM-NID-POSITION

NM-CBV-POSITION, NM-NID-POSITION are calculated by taking the maximum NM-CBV-POSITION, NM-NID-POSITION value of any NM-PDU transmitted over the CAN-CLUSTER in the source model, then dividing the result by eight.



NM-USER-DATA-LENGTH

NM-USER-DATA-LENGTH is calculated by taking the maximum NM-USER-DATA-LENGTH value of any NM-PDU transmitted over the CAN-CLUSTER.

8.1.1.7.4. CAN-NM-NODE

For every CAN-COMMUNICATION-CONTROLLER connected to a CAN-CLUSTER in the source model, one CAN-NM-NODE entity is created. It is aggregated in CAN-NM-CLUSTER/NM-NODES.

8.1.1.7.4.1. New parameters in the target model

NM-RANGE-CONFIG

If the COMMUNICATION-CLUSTER to which the CAN-COMMUNICATION-CONTROLLER is connected defines values for both NM-LOWER-CAN-ID and NM-UPPER-CAN-ID, one NM-RANGE-CONFIG element is added to the CAN-NM-NODE, having set LOWER-CAN-ID and UPPER-CAN-ID accordingly.

CONTROLLER-REF

CONTROLLER-REF is configured to reference the target model counterpart of the CAN-COMMUNICATION-CONTROLLER.

NM-NODE-ID

The parameter NM-NODE-ID is derived from the NM-ADDRESS parameter value of COMMUNICATION-CONNECTOR referencing the source model counterpart of the CAN-COMMUNICATION-CONTROLLER.

NM-IF-ECU-REF

NM-IF-ECU-REF is configured to reference the NM-ECU element, which has been created for the source model ECU-INSTANCE that aggregates the CAN-COMMUNICATION-CONTROLLER this CAN-NM-NODE has been created for.

RX-NM-PDU-REFS/RX-NM-PDU-REF

RX-NM-PDU-REFS/RX-NM-PDU-REF is configured to reference all NM-PDUS the CAN-COMMUNICATION-CONTROLLER is receiving.

TX-NM-PDU-REFS/TX-NM-PDU-REF

TX-NM-PDU-REFS/TX-NM-PDU-REF is configured to reference all NM-PDUS the CAN-COMMUNICATION-CONTROLLER is sending.

NM-MSG-REDUCED-TIME

NM-MSG-REDUCED-TIME is configured by CAN-COMMUNICATION-CONTROLLER/NM-MSG-REDUCED-TIME (supported by AUTOSAR 3.1 rev 0 system description file imports only).

NM-MSG-CYCLE-OFFSET

NM-MSG-CYCLE-OFFSET is configured by CAN-COMMUNICATION-CONTROLLER/NM-MSG-CYCLE-OFFSET (supported by AUTOSAR 3.1 rev 0 system description file imports only).



8.1.1.7.5. CAN-NM-CLUSTER-COUPING

CAN-NM-CLUSTER-COUPING defines parameters which have to be set consistently among two coupled CAN-NM-CLUSTER elements. *Coupled* means that least one NM-ECU exists, which is connected to both CAN-NM-CLUSTER elements. The coupling is transitive, which means that if CAN-NM-CLUSTER A is coupled to CAN-NM-CLUSTER B and CAN-NM-CLUSTER B is coupled to CAN-NM-CLUSTER C, CAN-NM-CLUSTER A is also coupled to CAN-NM-CLUSTER C.

For each of the coupled sets of CAN-NM-CLUSTER elements, and for each CAN-NM-CLUSTER not coupled to any other CAN-NM-CLUSTER, one element is created, aggregated in CAN-NM-CONFIG/NM-CLUSTER-COUPLINGS.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional CAN-NM-CLUSTER-COUPING parameter values are set (provided that they are defined consistently in the imported Fibex file):

Target model property	Fibex model property
NM-IMMEDIATE-RESTART-ENABLED	NM-CLUSTER-COUPINGS/NM-CLUSTER-COUPLING/IMMEDIATE-RESTART-ENABLED

8.1.1.7.5.1. New parameters in the target model

COUPLED-CLUSTER-REFS/COUPLED-CLUSTER-REF

COUPLED-CLUSTER-REFS/COUPLED-CLUSTER-REF is configured to reference all CAN-NM-CLUSTERS that are coupled.

NM-BUSLOAD-REDUCTION-ENABLED

If the NM-BUS-LOAD-REDUCTION-ENABLED parameters of the CAN-CLUSTER counterparts of the coupled CAN-NM-CLUSTERS yield a consistent value, the NM-BUSLOAD-REDUCTION-ENABLED of the CAN-NM-CLUSTER-COUPING is configured accordingly.

8.1.1.7.6. FLEXRAY-NM-CLUSTER

One FLEXRAY-NM-CLUSTER is created per FLEXRAY-CLUSTER in the source model. It is aggregated in NM-CONFIG/NM-CLUSTERS. It references the related FLEXRAY-CLUSTER in COMMUNICATION-CLUSTER-REF.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional FLEXRAY-NM-CLUSTER parameter values are set:

Target model property	Fibex model property
NM-CHANNEL-ID	NM-CLUSTER/CHANNEL-ID
NM-MESSAGE-TIMEOUT-TIME	NM-CLUSTER/MSG-TIMEOUT-TIME



Target model property	Fibex model property
NM-DATA-ENABLED	NM-CLUSTER/DATA-ENABLED
NM-MAIN-FUNCTION-PERIOD	max (NM-NODE/CYCLE-TIME-MAIN-FUNCTION)

For the configuration of NM-MAIN-FUNCTION-PERIOD, the maximum CYCLE-TIME-MAIN-FUNCTION value of all NM-NODES connected to the FLEXRAY-NM-CLUSTER is taken.

8.1.1.7.6.1. New parameters in the target model

NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPEAT-MESSAGE-TIME, NM-REPETITION-CYCLE, NM-VOTING-CYCLE, NM-DATA-CYCLE, NM-READY-SLEEP-COUNT

NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPEAT-MESSAGE-TIME, NM-REPETITION-CYCLE, NM-VOTING-CYCLE, NM-DATA-CYCLE, NM-READY-SLEEP-COUNT are determined by taking the respective parameter values of the source model FLEXRAY-CLUSTER (supported by AUTOSAR 3.1 rev 0 system description file imports only).

NM-CONTROL-BIT-VECTOR-ACTIVE

NM-CONTROL-BIT-VECTOR-ACTIVE is determined by taking the maximum NM-CONTROL-BIT-VECTOR-ENABLED parameter value of all source model FLEXRAY-COMMUNICATION-CONTROLLERS which are connected to the FLEXRAY-CLUSTER.

8.1.1.7.7. FLEXRAY-NM-NODE

For every FLEXRAY-COMMUNICATION-CONTROLLER connected to a FLEXRAY-CLUSTER in the source model, one FLEXRAY-NM-NODE entity is created. It is aggregated in FLEXRAY-NM-CLUSTER/NM-NODES.

8.1.1.7.7.1. New parameters in the target model

CONTROLLER-REF

CONTROLLER-REF is configured to reference the target model counterpart of the FLEXRAY-COMMUNICATION-CONTROLLER.

NM-NODE-ID

The parameter NM-NODE-ID is derived from the maximum NM-ADDRESS parameter value of all COMMUNICATION-CONNECTORS referencing the source model counterpart of the FLEXRAY-COMMUNICATION-CONTROLLER.

NM-IF-ECU-REF

NM-IF-ECU-REF is configured to reference the NM-ECU element which has been created for the source model ECU-INSTANCE aggregating the FLEXRAY-COMMUNICATION-CONTROLLER this FLEXRAY-NM-NODE has been created for.

**RX-NM-PDU-REFS/RX-NM-PDU-REF**

RX-NM-PDU-REFS/RX-NM-PDU-REF is configured to reference all NM-PDUS the FLEXRAY-COMMUNICATION-CONTROLLER is receiving.

TX-NM-PDU-REFS/TX-NM-PDU-REF

TX-NM-PDU-REFS/TX-NM-PDU-REF is configured to reference all NM-PDUS the FLEXRAY-COMMUNICATION-CONTROLLER is sending.

FLEXRAY-NM-NODE/NM-INSTANCE-ID

If a Fibex file has been imported into the EB tresos Studio source model, **FLEXRAY-NM-NODE/NM-INSTANCE-ID** is set according to the content of the Fibex parameter **NM-NODE/INSTANCE-ID**.

8.1.1.7.8. FLEXRAY-NM-CLUSTER-COUPING

FLEXRAY-NM-CLUSTER-COUPING defines parameters which have to be set consistently among two FLEXRAY-NM-CLUSTER elements which are coupled. [Section 8.1.1.7.5, “CAN-NM-CLUSTER-COUPING”](#) describes the peculiarities of the CAN cluster analogy CAN-NM-CLUSTER-COUPING.

For each of the coupled sets of FLEXRAY-NM-CLUSTER elements, and for each FLEXRAY-NM-CLUSTER not coupled to any other FLEXRAY-NM-CLUSTER, one FLEXRAY-NM-CLUSTER-COUPING element is created, aggregated in FLEXRAY-NM-CONFIG/NM-CLUSTER-COUPINGS.

If a Fibex file has been imported into the EB tresos Studio source model, the following additional FLEXRAY-NM-CLUSTER-COUPING parameter values are set (provided that they are defined consistently in the imported Fibex file):

Target model property	Fibex model property
NM-SCHEDULE-VARIANT	NM-CLUSTER-COUPINGS/NM-CLUSTER-COUPLING/SCHEDULE-VARIANT
NM-CONTROL-BIT-VECTOR-ENABLED	NM-CLUSTER-COUPINGS/NM-CLUSTER-COUPLING/CONTROL-BIT-VECTOR-ENABLED
NM-DATA-DISABLED	NM-CLUSTER-COUPINGS/NM-CLUSTER-COUPLING/NM-DATA-DISABLED

8.1.1.8. Software Component Description**8.1.1.8.1. COMPOSITION-SW-COMPONENT-TYPE**

Source model entity: COMPOSITION-TYPE



Target model entity: COMPOSITION-SW-COMPONENT-TYPE

Target model property	Source model property
POR TS/P-PORT-PROTOTYPE	POR TS/P-PORT-PROTOTYPE
POR TS/R-PORT-PROTOTYPE	POR TS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	No source model parameter is available.
COMPONENTS/SW-COMPONENT-PROTOTYPE	COMPONENTS/COMPONENT-PROTOTYPE
CONNECTORS/ASSEMBLY-SW-CONNECTOR	CONNECTORS/ASSEMBLY-CONNECTOR-PROTOTYPE
/CONNECTORS/DELEGATION-SW-CONNECTOR	CONNECTORS/DELEGATION-CONNECTOR-PROTOTYPE
No target model parameter is available.	CONNECTORS/SERVICE-CONNECTOR-PROTOTYPE

8.1.1.8.2. APPLICATION-SW-COMPONENT-TYPE

Source model entities: APPLICATION-SOFTWARE-COMPONENT-TYPE, SENSOR-ACTUATOR-SOFT-WARE-COMPONENT-TYPE

Target model entity: APPLICATION-SW-COMPONENT-TYPE

Target model property	Source model property
POR TS/P-PORT-PROTOTYPE	POR TS/P-PORT-PROTOTYPE
POR TS/R-PORT-PROTOTYPE	POR TS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	No source model parameter is available.

8.1.1.8.2.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.1.8.3. SERVICE-SW-COMPONENT-TYPE

Source model entity: SERVICE-COMPONENT-TYPE

Target model entity: SERVICE-SW-COMPONENT-TYPE



Target model property	Source model property
POR TS/P-PORT-PROTOTYPE	POR TS/P-PORT-PROTOTYPE
POR TS/R-PORT-PROTOTYPE	POR TS/R-PORT-PROTOTYPE
POR T-GROUPS/POR T-GROUP	No source model parameter is available.
No Upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MOD-ULE-DESCRIPTION-REF

8.1.1.8.3.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.1.8.4. COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE

Source model entity: COMPLEX-DEVICE-DRIVER-COMPONENT-TYPE

Target model entity: COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE

Target model property	Source model property
POR TS/P-PORT-PROTOTYPE	POR TS/P-PORT-PROTOTYPE
POR TS/R-PORT-PROTOTYPE	POR TS/R-PORT-PROTOTYPE
POR T-GROUPS/POR T-GROUP	No source model parameter is available.
No Upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MOD-ULE-DESCRIPTION-REF
No Upgrade is currently implemented.	HARDWARE-ELEMENT-REFS/HARDWARE-ELE-MENT-REF

8.1.1.8.4.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.



8.1.1.8.5. ECU-ABSTRACTION-SW-COMPONENT-TYPE

Source model entity: ECU-ABSTRACTION-COMPONENT-TYPE

Target model entity: ECU-ABSTRACTION-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	No source model parameter is available.
No upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MODULE-DESCRIPTION-REF
No upgrade is currently implemented.	HARDWARE-ELEMENT-REFS/HARDWARE-ELEMENT-REF

8.1.1.8.5.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.1.8.6. PARAMETER-SW-COMPONENT-TYPE

Source model entity: CALPRM-COMPONENT-TYPE

Target model entity: PARAMETER-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	No source model parameter is available.
CONSTANT-MAPPING-REFS/CONSTANT-MAPPING-REF	No source model parameter is available.
DATA-TYPE-MAPPING-REFS/DATA-TYPE-MAPPING-REF	Refer to Section 8.1.1.8.37.4, “Creation of target model DATA-TYPE-MAPPING-REFS/DATA-TYPE-MAPPING-REF” .



Target model property	Source model property
INSTANTIATION-DATA-DEF-PROPSS/INSTANTIATION-DATA-DEF-PROPS	No source model parameter is available.

8.1.1.8.6.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.1.8.7. R-PORT-PROTOTYPE

Source model entity: R-PORT-PROTOTYPE

Target model entity: R-PORT-PROTOTYPE

Target model property	Source model property
REQUIRED-COM-SPECS/CLIENT-COM-SPEC	REQUIRED-COM-SPECS/CLIENT-COM-SPEC
REQUIRED-COM-SPECS/PARAMETER-REQUIRED-COM-SPEC	REQUIRED-COM-SPECS/PARAMETER-REQUIRED-COM-SPEC
REQUIRED-COM-SPECS/QUEUED-RECEIVER-COM-SPEC	REQUIRED-COM-SPECS/QUEUED-RECEIVER-COM-SPEC
REQUIRED-COM-SPECS/NONQUEUED-RECEIVER-COM-SPEC	REQUIRED-COM-SPECS/UNQUEUED-RECEIVER-COM-SPEC
REQUIRED-COM-SPECS/NV-REQUIRED-COM-SPEC	No source model parameter is available.
REQUIRED-INTERFACE-TREF	REQUIRED-INTERFACE-TREF

8.1.1.8.8. P-PORT-PROTOTYPE

Source model entity: P-PORT-PROTOTYPE

Target model entity: P-PORT-PROTOTYPE

Target model property	Source model property
PROVIDED-COM-SPECS/MODE-SWITCH-SENDER-COM-SPEC	PROVIDED-COM-SPECS/MODE-SWITCH-COM-SPEC
PROVIDED-COM-SPECS/NONQUEUED-SENDER-COM-SPEC	PROVIDED-COM-SPECS/UNQUEUED-SENDER-COM-SPEC



Target model property	Source model property
PROVIDED-COM-SPECS/NV-PROVIDE-COM-SPEC	No source model parameter is available.
PROVIDED-COM-SPECS/PARAMETER-PROVIDE-COM-SPEC	PROVIDED-COM-SPECS/PARAMETER-PROVIDE-COM-SPEC
PROVIDED-COM-SPECS/QUEUED-SENDER-COM-SPEC	PROVIDED-COM-SPECS/QUEUED-SENDER-COM-SPEC
PROVIDED-COM-SPECS/SERVER-COM-SPEC	PROVIDED-COM-SPECS/SERVER-COM-SPEC
PROVIDED-INTERFACE-TREF	PROVIDED-INTERFACE-TREF

8.1.1.8.9. CLIENT-SERVER-INTERFACE

Source model entity: CLIENT-SERVER-INTERFACE

Target model entity: CLIENT-SERVER-INTERFACE

Target model property	Source model property
OPERATIONS/CLIENT-SERVER-OPERATION	OPERATIONS/OPERATION-PROTOTYPE
POSSIBLE-ERRORS/APPLICATION-ERROR	POSSIBLE-ERRORS/APPLICATION-ERROR
SERVICE-KIND	No source model parameter is available.

8.1.1.8.10. ASSEMBLY-SW-CONNECTOR

Source model entity: ASSEMBLY-CONNECTOR-PROTOTYPE

Target model entity: ASSEMBLY-SW-CONNECTOR

Target model property	Source model property
PROVIDER-IREF/COMPONENT-PROTOTYPE-REF	PROVIDER-IREF/COMPONENT-PROTOTYPE-REF
PROVIDER-IREF/P-PORT-PROTOTYPE-REF	PROVIDER-IREF/P-PORT-PROTOTYPE-REF
REQUESTER-IREF/COMPONENT-PROTOTYPE-REF	REQUESTER-IREF/COMPONENT-PROTOTYPE-REF
REQUESTER-IREF/R-PORT-PROTOTYPE-REF	REQUESTER-IREF/R-PORT-PROTOTYPE-REF
No source model parameter is available.	MAPPING-REF

8.1.1.8.11. DELEGATION-SW-CONNECTOR

Source model entity: DELEGATION-CONNECTOR-PROTOTYPE



Target model entity: DELEGATION-SW-CONNECTOR

Target model property	Source model property
INNER-PORT-IREF/CONTEXT-COMPONENT-REF	INNER-PORT-IREF/COMPONENT-PROTOTYPE-REF
INNER-PORT-IREF/TARGET-P-PORT-REF	INNER-PORT-IREF/PORT-PROTOTYPE-REF
OUTER-PORT-REF	OUTER-PORT-REF
No source model parameter is available.	MAPPING-REF

8.1.1.8.12. SERVICE-CONNECTOR-PROTOTYPE

AUTOSAR 3.1.x SERVICE-CONNECTOR-PROTOTYPE elements are not upgraded because no counterpart element exists in AUTOSAR 4.0. If such SENDER-RECEIVER-INTERFACE elements are encountered, a warning is issued.

8.1.1.8.13. PARAMETER-INTERFACE

Source model entity: CALPRM-INTERFACE

Target model entity: PARAMETER-INTERFACE

Target model property	Source model property
IS-SERVICE	IS-SERVICE
CALPRM-ELEMENTS/CALPRM-ELEMENT-PROTOTYPE	PARAMETERS/PARAMETER-DATA-PROTOTYPE
SERVICE-KIND	No source model parameter is available.

8.1.1.8.14. MODE-SWITCH-INTERFACE

Source model entity: SENDER-RECEIVER-INTERFACE

Target model entity: MODE-SWITCH-INTERFACE

For every SENDER-RECEIVER-INTERFACE element in the source model that contains one or more MODE-GROUPS, but no DATA-ELEMENT, one MODE-SWITCH-INTERFACE is created. Since the target model only supports the definition of one single MODE-DECLARATION-GROUP-PROTOTYPE, only the first MODE-DECLARATION-GROUP-PROTOTYPE encountered in the source model is upgraded.



The upgrade of SENDER-RECEIVER-INTERFACE elements which in turn contain MODE-GROUPS elements and DATA-ELEMENTS at the same time is not supported. If such SENDER-RECEIVER-INTERFACE elements are encountered, none of the MODE-GROUPS is upgraded and a warning is issued.

Target model property	Source model property
IS-SERVICE	IS-SERVICE
MODE-GROUP	MODE-GROUPS/MODE-DECLARATION-GROUP-PROTOTYPE
SERVICE-KIND	No source model parameter is available.

8.1.1.8.15. MODE-GROUP

Source model entity: MODE-GROUP

Target model entity: MODE-GROUP

Target model property	Source model property
TYPE-TREF	TYPE-TREF

8.1.1.8.16. SW-COMPONENT-PROTOTYPE

Source model entity: COMPONENT-PROTOTYPE

Target model entity: SW-COMPONENT-PROTOTYPE

Target model property	Source model property
TYPE-TREF	TYPE-TREF

8.1.1.8.17. VARIABLE-DATA-PROTOTYPE

Source model entity: DATA-ELEMENT-PROTOTYPE

Target model entity: VARIABLE-DATA-PROTOTYPE

Target model property	Source model property
SW-DATA-DEF-PROPS	SW-DATA-DEF-PROPS
TYPE-TREF	TYPE-TREF
No target model parameter is available.	IS-QUEUED



Target model property	Source model property
INIT-VALUE	No source model parameter is available.

8.1.1.8.17.1. Reason for not upgraded source model parameter IS-QUEUED

IS-QUEUED

The parameter IS-QUEUED is not upgraded to the target model, the target model entity VARIABLE-DATA-PROTOTYPE does not contain it. If IS-QUEUED is set to true in the source model, the parameter SW-DATA-DEF-PROPS/SW-IMPL-POLICY is set to QUEUED.

8.1.1.8.18. ARGUMENT-DATA-PROTOTYPE

Source model entity: ARGUMENT-PROTOTYPE

Target model entity: ARGUMENT-DATA-PROTOTYPE

Target model property	Source model property
SW-DATA-DEF-PROPS	SW-DATA-DEF-PROPS
TYPE-TREF	TYPE-TREF
DIRECTION	DIRECTION
SERVER-ARGUMENT-IMPL-POLICY	No source model parameter is available.

8.1.1.8.19. PARAMETER-DATA-PROTOTYPE

Source model entity: CALPRM-ELEMENT-PROTOTYPE

Target model entity: PARAMETER-DATA-PROTOTYPE

Target model property	Source model property
SW-DATA-DEF-PROPS	SW-DATA-DEF-PROPS
TYPE-TREF	TYPE-TREF

8.1.1.8.19.1. Configuration of target model parameter INIT-VALUE

If the source model contains a LOCAL-PARAMETER-INIT-VALUE-ASSIGNMENT which references in PARAMETER-REF the source model CALPRM-ELEMENT-PROTOTYPE and in INIT-VALUE-REF an initial value, the target model PARAMETER-DATA-PROTOTYPE references this value in INIT-VALUE/CONSTANT-REF.



8.1.1.8.20. CLIENT-SERVER-OPERATION

Source model entity: OPERATION-PROTOTYPE

Target model entity: CLIENT-SERVER-OPERATION

Target model property	Source model property
ARGUMENTS/ARGUMENT-DATA-PROTOTYPE	ARGUMENTS/ARGUMENT-PROTOTYPE
POSSIBLE-ERROR-REFS/POSSIBLE-ERROR-REF	POSSIBLE-ERROR-REFS/POSSIBLE-ERROR-REF

8.1.1.8.21. APPLICATION-ERROR

Source model entity: APPLICATION-ERROR

Target model entity: APPLICATION-ERROR

Target model property	Source model property
ERROR-CODE	ERROR-CODE

8.1.1.8.22. NONQUEUED-RECEIVER-COM-SPEC

Source model entity: UNQUEUED-RECEIVER-COM-SPEC

Target model entity: NONQUEUED-RECEIVER-COM-SPEC

Target model property	Source model property
ALIVE-TIMEOUT	ALIVE-TIMEOUT
No target model parameter is available.	HANDLE-INVALID
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF
No target model parameter is available.	RESYNC-TIME
DATA-ELEMENT-REF	DATA-ELEMENT-REF
FILTER	FILTER
HANDLE-NEVER-RECEIVED	No source model parameter is available, the target model parameter is set to false.
ENABLE-UPDATE	No source model parameter is available, the target model parameter is set to false.
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.



Target model property	Source model property
HANDLE-TIMEOUT-TYPE	No source model parameter is available, the target model parameter is set to NONE.
NETWORK-REPRESENTATION	No source model parameter is available.
USES-END-TO-END-PROTECTION	No source model parameter is available, the target model parameter is set to true.

8.1.1.8.22.1. Processing of source model parameter HANDLE-INVALID

If the parameter HANDLE-INVALID is provided in the source model, an INVALIDATION-POLICY element is created in the related target model SENDER-RECEIVER-INTERFACE. The INVALIDATION-POLICY parameter HANDLE-INVALID is configured according to the source model UNQUEUED-RECEIVER-COM-SPEC parameter HANDLE-INVALID if all UNQUEUED-RECEIVER-COM-SPEC elements belonging to the source model SENDER-RECEIVER-INTERFACE and DATA-ELEMENT are providing the same HANDLE-INVALID value.

8.1.1.8.23. QUEUED-RECEIVER-COM-SPEC

Source model entity: QUEUED-RECEIVER-COM-SPEC

Target model entity: QUEUED-RECEIVER-COM-SPEC

Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
No target model parameter is available.	FILTER
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.
NETWORK-REPRESENTATION	No source model parameter is available.
QUEUE-LENGTH	QUEUE-LENGTH
USES-END-TO-END-PROTECTION	No source model parameter is available, the target model parameter is set to true.

8.1.1.8.24. CLIENT-COM-SPEC

Source model entity: CLIENT-COM-SPEC

Target model entity: CLIENT-COM-SPEC



Target model property	Source model property
No target model parameter is available.	DATA-ELEMENT-REF
No target model parameter is available.	FILTER
OPERATION-REF	OPERATION-REF

8.1.1.8.25. PARAMETER-REQUIRE-COM-SPEC

Source model entity: PARAMETER-REQUIRE-COM-SPEC

Target model entity: PARAMETER-REQUIRE-COM-SPEC

Target model property	Source model property
No target model parameter is available.	DATA-ELEMENT-REF
No target model parameter is available.	FILTER
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF
PARAMETER-REF	PARAMETER-REF

8.1.1.8.25.1. Upgrade of source model INIT-VALUE

The target model contains 0 as INIT-VALUE if the source model does not provide INIT-VALUE-REF.

8.1.1.8.26. QUEUED-SENDER-COM-SPEC

Source model entity: QUEUED-SENDER-COM-SPEC

Target model entity: QUEUED-SENDER-COM-SPEC

Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
TRANSMISSION-ACKNOWLEDGE	TRANSMISSION-ACKNOWLEDGE
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.
NETWORK-REPRESENTATION	No source model parameter is available.
USES-END-TO-END-PROTECTION	No source model parameter is available, the target model parameter is set to true.



8.1.1.8.27. NONQUEUED-SENDER-COM-SPEC

Source model entity: UNQUEUED-SENDER-COM-SPEC

Target model entity: NONQUEUED-SENDER-COM-SPEC

Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
TRANSMISSION-ACKNOWLEDGE	TRANSMISSION-ACKNOWLEDGE
No target model parameter is available.	CAN-INVALIDATE
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.
NETWORK-REPRESENTATION	No source model parameter is available.
USES-END-TO-END-PROTECTION	No source model parameter is available, the target model parameter is set to true.

8.1.1.8.28. SERVER-COM-SPEC

Source model entity: SERVER-COM-SPEC

Target model entity: SERVER-COM-SPEC

Target model property	Source model property
No target model parameter is available.	DATA-ELEMENT-REF
No target model parameter is available.	TRANSMISSION-ACKNOWLEDGE
OPERATION-REF	OPERATION-REF
QUEUE-LENGTH	QUEUE-LENGTH

8.1.1.8.29. MODE-SWITCH-SENDER-COM-SPEC

Source model entity: MODE-SWITCH-COM-SPEC

Target model entity: MODE-SWITCH-SENDER-COM-SPEC

Target model property	Source model property
MODE-GROUP-REF	MODE-GROUP-REF



Target model property	Source model property
MODE-SWITCHED-ACK	MODE-SWITCHED-ACK
QUEUE-LENGTH	QUEUE-LENGTH

8.1.1.8.30. PARAMETER-PROVIDE-COM-SPEC

Source model entity: PARAMETER-PROVIDE-COM-SPEC

Target model entity: PARAMETER-PROVIDE-COM-SPEC

Target model property	Source model property
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF
PARAMETER-REF	PARAMETER-REF

8.1.1.8.30.1. Upgrade of source model INIT-VALUE

The target model contains 0 as INIT-VALUE if the source model does not provide INIT-VALUE-REF.

8.1.1.8.31. TRANSMISSION-ACKNOWLEDGE

Source model entity: TRANSMISSION-ACKNOWLEDGE

Target model entity: TRANSMISSION-ACKNOWLEDGE

Target model property	Source model property
TIMEOUT	TIMEOUT

8.1.1.8.32. DATA-FILTER

Source model entity: TRANSMISSION-MODE-CONDITION/DATA-FILTER

Target model entity: DATA-FILTER

Target model property	Source model property
MASK	MASK
MAX	MAX



Target model property	Source model property
MIN	MIN
OFFSET	OFFSET
PERIOD	PERIOD
X	X

8.1.1.8.32.1. Configuration of target model parameter DATA-FILTER-TYPE

DATA-FILTER-TYPE is configured depending on the source model element below TRANSMISSION-MODE-CONDITION/DATA-FILTER. The following values are converted:

- ▶ ALWAYS
- ▶ MASKED-NEW-DIFFERS-MASKED-OLD
- ▶ MASKED-NEW-DIFFERS-X
- ▶ MASKED-NEW-EQUALS-MASKED-OLD
- ▶ MASKED-NEW-EQUALS-X
- ▶ NEVER
- ▶ NEW-IS-DIFFERENT
- ▶ NEW-IS-EQUAL
- ▶ NEW-IS-GREATER
- ▶ NEW-IS-GREATER-OR-EQUAL
- ▶ NEW-IS-LESS
- ▶ NEW-IS-LESS-OR-EQUAL
- ▶ NEW-IS-OUTSIDE
- ▶ NEW-IS-WITHIN
- ▶ ONE-EVERY-N

8.1.1.8.33. VARIABLE-ACCESS

The target model entity VARIABLE-ACCESS is created out of different source model entities. The source model entities and the employed parameter mapping rules are listed below.

Source model entities: DATA-READ-ACCESS, DATA-RECEIVE-POINT

Target model entity: VARIABLE-ACCESS



Target model property	Source model property
ACCESSED-VARIABLE/AUTOSAR-VARIABLE-IREF/PORT-PROTOTYPE-REF	DATA-ELEMENT-IREF/R-PORT-PROTOTYPE-REF
ACCESSED-VARIABLE/AUTOSAR-VARIABLE-IREF/TARGET-DATA-PROTOTYPE-REF	DATA-ELEMENT-IREF/DATA-ELEMENT-PROTOTYPE-REF

Source model entities: DATA-WRITE-ACCESS, DATA-SEND-POINT

Target model entity: VARIABLE-ACCESS

Target model property	Source model property
ACCESSED-VARIABLE/AUTOSAR-VARIABLE-IREF/PORT-PROTOTYPE-REF	DATA-ELEMENT-IREF/P-PORT-PROTOTYPE-REF
ACCESSED-VARIABLE/AUTOSAR-VARIABLE-IREF/TARGET-DATA-PROTOTYPE-REF	DATA-ELEMENT-IREF/DATA-ELEMENT-PROTOTYPE-REF

8.1.1.8.34. PARAMETER-ACCESS

The target model entity PARAMETER-ACCESS is created out of different source model entities. The source model entities and the employed parameter mapping rules are listed below.

Source model entity: CALPRM-ACCESS

Target model entity: PARAMETER-ACCESS

Target model property	Source model property
PARAMETER-ACCESS/ACCESSED-PARAMETER/AUTOSAR-PARAMETER-IREF/PORT-PROTOTYPE-REF	CALPRM-ACCESS-IREF/PORT-PROTOTYPE-REF
PARAMETER-ACCESS/ACCESSED-PARAMETER/AUTOSAR-PARAMETER-IREF/TARGET-DATA-PROTOTYPE-REF	CALPRM-ACCESS-IREF/CALPRM-ELEMENT-PROTOTYPE-REF

Source model entity: SHARED-CALPRM-ACCESS-REF

Target model entity: PARAMETER-ACCESS

Target model property	Source model property
PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	SHARED-CALPRM-ACCESS-REF

Source model entity: PER-INSTANCE-CALPRM-ACCESS-REF



Target model entity: PARAMETER-ACCESS

Target model property	Source model property
PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	PER-INSTANCE-CALPRM-ACCESS-REF

8.1.1.8.34.1. SHORT-NAME of target model ACCESSED-VARIABLE

Note that for all upgrade variants, an ACCESSED-PARAMETER is created in the target model. For all variants, the ACCESSED-PARAMETER/SHORT-NAME is set to PA<index>, where <index> is an index providing for the uniqueness of ACCESSED-PARAMETER/SHORT-NAME in the target model.

8.1.1.8.35. RUNNABLE-ENTITY

Source model entity: RUNNABLE-ENTITY

Target model entity: RUNNABLE-ENTITY

Target model property	Source model property
Currently no upgrade is available.	BSW-ENTITY-REF
PARAMETER-ACCESSSS/PARAMETER-ACCESS/ACCESSED-PARAMETER	CALPRM-ACCESSSS/CALPRM-ACCESS
CAN-BE-INVOKED-CONCURRENTLY	CAN-BE-INVOKED-CONCURRENTLY
CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF	CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF
DATA-READ-ACCESSSS/VARIABLE-ACCESS	DATA-READ-ACCESSSS/DATA-READ-ACCESS
DATA-RECEIVE-POINT-BY-ARGUMENTS/VARIABLE-ACCESS	DATA-RECEIVE-POINTS/DATA-RECEIVE-POINT
DATA-SEND-POINTS/VARIABLE-ACCESS	DATA-SEND-POINTS/DATA-SEND-POINT
DATA-WRITE-ACCESSSS/VARIABLE-ACCESS	DATA-WRITE-ACCESSSS/DATA-WRITE-ACCESS
MINIMUM-START-INTERVAL	MINIMUM-START-INTERVAL
MODE-SWITCH-POINTS/MODE-SWITCH-POINT	MODE-SWITCH-POINTS/MODE-SWITCH-POINT
PARAMETER-ACCESSSS/PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	PER-INSTANCE-CALPRM-ACCESS-REFS/PER-INSTANCE-CALPRM-ACCESS-REF
RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF	RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF



Target model property	Source model property
SERVER-CALL-POINTS/ASYNCHRONOUS-SERVER-CALL-POINT	SERVER-CALL-POINTS/ASYNCHRONOUS-SERVER-CALL-POINT
SERVER-CALL-POINTS/SYNCHRONOUS-SERVER-CALL-POINT	SERVER-CALL-POINTS/SYNCHRONOUS-SERVER-CALL-POINT
PARAMETER-ACCESSSS/PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	SHARED-CALPRM-ACCESS-REFS/SHARED-CALPRM-ACCESS-REF
SYMBOL	SYMBOL
WAIT-POINTS/WAIT-POINT	WAIT-POINTS/WAIT-POINT
WRITTEN-LOCAL-VARIABLES/VARIABLE-ACCESS/ACCESSED-VARIABLE/LOCAL-VARIABLE-REF	WRITTEN-VARIABLE-REFS/WRITTEN-VARIABLE-REF
READ-LOCAL-VARIABLES/VARIABLE-ACCESS/ACCESSED-VARIABLE/LOCAL-VARIABLE-REF	READ-VARIABLE-REFS/READ-VARIABLE-REF

8.1.1.8.35.1. Migration of source model DATA-RECEIVE-POINT elements

Note that all source model DATA-RECEIVE-POINT elements are migrated to DATA-RECEIVE-POINT-BY-ARGUMENTS/VARIABLE-ACCESS elements in the target model, i.e. there is no conversion to the target model entities DATA-RECEIVE-POINT-BY-VALUES/VARIABLE-ACCESS.

8.1.1.8.35.2. Creation of target model ASYNCHRONOUS-SERVER-CALL-RESULT-POINT elements

Note that for every source model ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT entity which either references a RUNNABLE-ENTITY via START-ON-EVENT-REF or references a SERVER-CALL-POINT of the RUNNABLE-ENTITY via EVENT-SOURCE-REF, one ASYNCHRONOUS-SERVER-CALL-RESULT-POINTS/ASYNCHRONOUS-SERVER-CALL-RESULT-POINT is created in the target model.

8.1.1.8.36. WAIT-POINT

Source model entity: WAIT-POINT

Target model entity: WAIT-POINT

Target model property	Source model property
TIMEOUT	TIMEOUT



Target model property	Source model property
TRIGGER-REF	TRIGGER-REF

8.1.1.8.37. SWC-INTERNAL-BEHAVIOR

Source model entity: INTERNAL-BEHAVIOR

Target model entity: SWC-INTERNAL-BEHAVIOR

Target model property	Source model property
No migration is required, see Section 8.1.1.8.2, “APPLICATION-SW-COMPONENT-TYPE” .	COMPONENT-REF
EVENTS	EVENTS
EXCLUSIVE-AREAS/EXCLUSIVE-AREA	EXCLUSIVE-AREAS/EXCLUSIVE-AREA
HANDLE-TERMINATION-AND-RESTART	No source model parameter is available, the target model parameter is set to NOSUPPORT.
PER-INSTANCE-PARAMETERS/PARAMETER-DATA-PROTOTYPE	PER-INSTANCE-CALPRMS/CALPRM-ELEMENT-PROTOTYPE
PER-INSTANCE-MEMORYS/PER-INSTANCE-MEMORY	PER-INSTANCE-MEMORYS/PER-INSTANCE-MEMORY
PORT-API-OPTIONS/PORT-API-OPTION	PORT-API-OPTIONS/PORT-API-OPTION
RUNNABLES/RUNNABLE-ENTITY	RUNNABLES/RUNNABLE-ENTITY
SHARED-PARAMETERS/PARAMETER-DATA-PROTOTYPE	SHARED-CALPRMS/CALPRM-ELEMENT-PROTOTYPE
SUPPORTS-MULTIPLE-INSTANTIATION	SUPPORTS-MULTIPLE-INSTANTIATION
AR-TYPED-PER-INSTANCE-MEMORYS/VARIABLE-DATA-PROTOTYPE	No source model parameter is available.
CONSTANT-MEMORYS/PARAMETER-DATA-PROTOTYPE	No source model parameter is available.
CONSTANT-VALUE-MAPPING-REFS/CONSTANT-VALUE-MAPPING-REF	No source model parameter is available.
INCLUDED-DATA-TYPE-SETS/INCLUDED-DATA-TYPE-SET	No source model parameter is available.
INSTANTIATION-DATA-DEF-PROPSS/INSTANTIATION-DATA-DEF-PROPS	No source model parameter is available.
STATIC-MEMORYS/VARIABLE-DATA-PROTOTYPE	No source model parameter is available.



8.1.1.8.37.1. Creation of target model PARAMETER-DATA-PROTOTYPE/INIT-VALUE for INIT-VALUES/LOCAL-PARAMETER-INIT-VALUE-ASSIGNMENT

Note that for every source model INIT-VALUES/LOCAL-PARAMETER-INIT-VALUE-ASSIGNMENT one PARAMETER-DATA-PROTOTYPE/INIT-VALUE element is created for the target model counterpart of the referenced CALPRM-ELEMENT-PROTOTYPE.

8.1.1.8.37.2. Upgrade of source model INTER-RUNNABLE-VARIABLES/INTER-RUNNABLE-VARIABLE

If the source model value of INTER-RUNNABLE-VARIABLES/INTER-RUNNABLE-VARIABLE/COMMUNICATION-APPROACH is set to EXPLICIT, the content of INTER-RUNNABLE-VARIABLES/INTER-RUNNABLE-VARIABLE is upgraded to the target model parameter EXPLICIT-INTER-RUNNABLE-VARIABLES/VARIABLE-DATA-PROTOTYPE, in all other cases to IMPLICIT-INTER-RUNNABLE-VARIABLES/VARIABLE-DATA-PROTOTYPE. If the source model provides the parameter INTER-RUNNABLE-VARIABLES/INTER-RUNNABLE-VARIABLE/INIT-VALUE-REF, a corresponding parameter INTER-RUNNABLE-VARIABLES/INTER-RUNNABLE-VARIABLE/INIT-VALUE/CONSTANT-REF is set up in the target model.

8.1.1.8.37.3. Upgrade of source model SERVICE-NEEDSS

For every SERVICE-NEEDS instance below SERVICE-NEEDSS in the source model, one SWC-SERVICE-DEPENDENCY is created below SERVICE-DEPENDENCIES in the target model which in turn will contain the upgraded SERVICE-NEEDS instance.

8.1.1.8.37.4. Creation of target model DATA-TYPE-MAPPING-REFS/DATA-TYPE-MAPPING-REF

For every source model MODE-SWITCH-POINT aggregated by any RUNNABLE-ENTITY of the INTERNAL-BEHAVIOR, one DATA-TYPE-MAPPING-SET is created, which the target model INTERNAL-BEHAVIOR references in DATA-TYPE-MAPPING-REFS/DATA-TYPE-MAPPING-REF. See also [Section 8.1.1.8.82, "MODE-ACCESS-POINT"](#).

MODE-REQUEST-TYPE-MAP/IMPLEMENTATION-DATA-TYPE-REF references an IMPLEMENTATION-DATA-TYPE representing an unsigned eight bit data type. MODE-REQUEST-TYPE-MAP/MODE-GROUP-REF references the target model counterpart of the source model MODE-DECLARATION-GROUP element referenced by the MODE-DECLARATION-GROUP-PROTOTYPE, which in turn is referenced by the source model MODE-SWITCH-POINT.

8.1.1.8.38. PORT-API-OPTION

Source model entity: PORT-API-OPTION



Target model entity: PORT-API-OPTION

Target model property	Source model property
ENABLE-TAKE-ADDRESS	ENABLE-TAKE-ADDRESS
INDIRECT-API	INDIRECT-API
PORT-ARG-VALUES	PORT-ARG-VALUES
PORT-REF	PORT-REF

8.1.1.8.38.1. Upgrade of PORT-ARG-VALUES

For every source model element in PORT-ARG-VALUES, one PORT-DEFINED-ARGUMENT-VALUE is created in the target model. The source model elements are then aggregated within their target model PORT-DEFINED-ARGUMENT-VALUE elements. See [Section 8.1.1.8.49, “NUMERICAL-VALUE-SPECIFICATION”](#) and the following chapters for details.

8.1.1.8.39. DATA-SEND-COMPLETED-EVENT

Source model entity: DATA-SEND-COMPLETED-EVENT

Target model entity: DATA-SEND-COMPLETED-EVENT

Target model property	Source model property
EVENT-SOURCE-REF	EVENT-SOURCE-REF
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF

8.1.1.8.40. DATA-RECEIVED-EVENT

Source model entity: DATA-RECEIVED-EVENT

Target model entity: DATA-RECEIVED-EVENT

Target model property	Source model property
DATA-IREF	DATA-IREF
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF



8.1.1.8.41. DATA-RECEIVE-ERROR-EVENT

Source model entity: DATA-RECEIVE-ERROR-EVENT

Target model entity: DATA-RECEIVE-ERROR-EVENT

Target model property	Source model property
DATA-IREF	DATA-IREF
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF

8.1.1.8.42. ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT

Source model entity: ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT

Target model entity: ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT

Target model property	Source model property
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF

8.1.1.8.42.1. Upgrade of source model EVENT-SOURCE-REF

Note that in the target model one ASYNCHRONOUS-SERVER-CALL-RESULT-POINT element is created for each ASYNCHRONOUS-SERVER-CALL-POINT referenced by the source model EVENT-SOURCE-REF. The target model EVENT-SOURCE-REF references the ASYNCHRONOUS-SERVER-CALL-RESULT-POINT.

8.1.1.8.43. TIMING-EVENT

Source model entity: TIMING-EVENT

Target model entity: TIMING-EVENT

Target model property	Source model property
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF
PERIOD	PERIOD



8.1.1.8.44. OPERATION-INVOKED-EVENT

Source model entity: OPERATION-INVOKED-EVENT

Target model entity: OPERATION-INVOKED-EVENT

Target model property	Source model property
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF
OPERATION-IREF	OPERATION-IREF

8.1.1.8.45. SWC-MODE-SWITCH-EVENT

Source model entity: MODE-SWITCH-EVENT

Target model entity: SWC-MODE-SWITCH-EVENT

Target model property	Source model property
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF
ACTIVATION	ACTIVATION
MODE-IREFS/MODE-IREF/CONTEXT-MODE-DECLARATION-GROUP-PROTOTYPE-REF	MODE-IREF/MODE-DECLARATION-GROUP-PROTOTYPE-REF
MODE-IREFS/MODE-IREF/TARGET-MODE-DECLARATION-REF	MODE-IREF/MODE-DECLARATION-REF
MODE-IREFS/MODE-IREF/CONTEXT-PORT-REF	MODE-IREF/R-PORT-PROTOTYPE-REF

8.1.1.8.46. MODE-SWITCHED-ACK-EVENT

Source model entity: MODE-SWITCHED-ACK-EVENT

Target model entity: MODE-SWITCHED-ACK-EVENT

Target model property	Source model property
DISABLED-MODE-IREFS/DISABLED-MODE-IREF	MODE-DEPENDENCY/DEPENDENT-ON-MODE-IREFS/DEPENDENT-ON-MODE-IREF
START-ON-EVENT-REF	START-ON-EVENT-REF
EVENT-SOURCE-REF	EVENT-SOURCE-REF



8.1.1.8.47. ASYNCHRONOUS-SERVER-CALL-POINT

Source model entity: ASYNCHRONOUS-SERVER-CALL-POINT

Target model entity: ASYNCHRONOUS-SERVER-CALL-POINT

Target model property	Source model property
OPERATION-IREF	OPERATION-IREFS/OPERATION-IREF
TIMEOUT	TIMEOUT

8.1.1.8.47.1. Upgrade of source model OPERATION-IREFS/OPERATION-IREF

Note that only the first source model entity in OPERATION-IREFS/OPERATION-IREF is upgraded to the target model OPERATION-IREF since the target model allows at most one OPERATION-IREF element.

8.1.1.8.48. SYNCHRONOUS-SERVER-CALL-POINT

Source model entity: SYNCHRONOUS-SERVER-CALL-POINT

Target model entity: SYNCHRONOUS-SERVER-CALL-POINT

Target model property	Source model property
OPERATION-IREF	OPERATION-IREFS/OPERATION-IREF
TIMEOUT	TIMEOUT

8.1.1.8.48.1. Upgrade of source model OPERATION-IREFS/OPERATION-IREF

Note that only the first source model entity in OPERATION-IREFS/OPERATION-IREF is upgraded to the target model OPERATION-IREF since the target model allows at most one OPERATION-IREF element.

8.1.1.8.49. NUMERICAL-VALUE-SPECIFICATION

Source model entities: INTEGER-LITERAL, REAL-LITERAL, BOOLEAN-LITERAL, OPAQUE-LITERAL, CHAR-LITERAL

Target model entity: NUMERICAL-VALUE-SPECIFICATION

Target model property	Source model property
SHORT-LABEL	SHORT-NAME
No target model parameter is available.	TYPE-TREF



Target model property	Source model property
VALUE	VALUE

8.1.1.8.49.1. Upgrade of source model VALUE

Note that depending on the source model entity, the content of the source model `VALUE` is upgraded differently into the target model `VALUE`:

- ▶ **INTEGER-LITERALS:** No conversion is required.
- ▶ **REAL-LITERALS:** No conversion is required.
- ▶ **BOOLEAN-LITERALS:** `false` is converted to 0, `true` is converted to 1.
- ▶ **OPAQUE-LITERALS:** The opaque literal byte array is converted into an integral number. The first byte lies in the target model value MSB, the last byte in the target model LSB. Supported opaque literals are at most eight bytes long.
- ▶ **CHAR-LITERALS:** The value is converted directly. Note that the model upgrade assumes that the source model value is provided as code which represents the character, it is not provided as character itself.

8.1.1.8.50. TEXT-VALUE-SPECIFICATION

Source model entity: STRING-LITERAL

Target model entity: TEXT-VALUE-SPECIFICATION

Target model property	Source model property
SHORT-LABEL	SHORT-NAME
No target model parameter is available.	TYPE-TREF
VALUE	VALUE

8.1.1.8.51. ARRAY-VALUE-SPECIFICATION

Source model entity: ARRAY-SPECIFICATION

Target model entity: ARRAY-VALUE-SPECIFICATION

Target model property	Source model property
SHORT-LABEL	SHORT-NAME
No target model parameter is available.	TYPE-TREF
ELEMENTS	ELEMENTS



8.1.1.8.52. RECORD-VALUE-SPECIFICATION

Source model entity: RECORD-SPECIFICATION

Target model entity: RECORD-VALUE-SPECIFICATION

Target model property	Source model property
SHORT-LABEL	SHORT-NAME
No target model parameter is available.	TYPE-TREF
FIELDS	ELEMENTS

8.1.1.8.53. CONSTANT-SPECIFICATION

Source model entity: CONSTANT-SPECIFICATION

Target model entity: CONSTANT-SPECIFICATION

Target model property	Source model property
SHORT-NAME	SHORT-NAME
VALUE-SPEC	VALUE

8.1.1.8.54. CONSTANT-REFERENCE

Source model entity: CONSTANT-REFERENCE

Target model entity: CONSTANT-REFERENCE

Target model property	Source model property
TYPE-TREF	TYPE-TREF
CONSTANT-REF	CONSTANT-REF

8.1.1.8.55. UNIT

Source model entity: UNIT

Target model entity: UNIT

Target model property	Source model property
FACTOR-SI-TO-UNIT	FACTOR-SI-TO-UNIT
OFFSET-SI-TO-UNIT	OFFSET-SI-TO-UNIT



8.1.1.8.55.1. No upgrade of source model DISPLAY-NAME and PHYSICAL-DIMENSION-REF

Due to internal data model restrictions, the source model parameters **DISPLAY-NAME** and **PHYSICAL-DIMENSION-REF** are not upgraded to the target model.

8.1.1.8.56. UNIT-GROUP

Due to internal data model restrictions, there is no upgrade path for **UNIT-GROUP** elements.

8.1.1.8.57. SW-BASE-TYPE

Source model entity: **SW-BASE-TYPE**

Target model entity: **SW-BASE-TYPE**

Target model property	Source model property
MEM-ALIGNMENT	MEM-ALIGNMENT
BASE-TYPE-SIZE	BASE-TYPE-SIZE

8.1.1.8.57.1. Upgrade of source model BASE-TYPE-ENCODING

If the source model **BASE-TYPE-ENCODING** contains one of the values

- ▶ 1C
- ▶ 2C
- ▶ BCD-P
- ▶ BCD-UP
- ▶ DSP-FRACTIONAL
- ▶ SM
- ▶ IEEE754
- ▶ ISO-8859-1
- ▶ ISO-8859-2
- ▶ WINDOWS-1252
- ▶ UTF-8
- ▶ UCS-2
- ▶ NONE



the value is directly copied into the target model parameter `BASE-TYPE-ENCODING`. For any other value in the source model, `NONE` will be written into the target model parameter.

8.1.1.8.57.2. Upgrade of source model `CATEGORY`

If the source model `CATEGORY` value is either `SW_FIXED_LENGTH` or `FIXED_LENGTH`, the target model `CATEGORY` parameter will contain the value `FIXED_LENGTH`. If the source model parameter value contains `SW_VARIABLE_LENGTH` or `VARIABLE_LENGTH`, the target model `CATEGORY` parameter value will contain `VARIABLE_LENGTH`.

8.1.1.8.57.3. No upgrade of source model `MAX-BASE-TYPE-SIZE` and `BASE-TYPE-REF`

Due to internal data model restrictions, the source model parameters `MAX-BASE-TYPE-SIZE` and `BASE-TYPE-REF` are not upgraded into the target model.

8.1.1.8.57.4. No upgrade of source model `BYTE-ORDER`

The source model parameter `BYTE-ORDER` is not upgraded into the target model because `BYTE-ORDER` is not a property of a (base) data type.

8.1.1.8.58. SW-DATA-DEF-PROPS

Source model entity: `SW-DATA-DEF-PROPS`

Target model entity: `SW-DATA-DEF-PROPS`

Target model property	Source model property
<code>SW-DATA-DEF-PROPS-CONDITIONAL/BASE-TYPE-REF</code>	<code>BASE-TYPE-REF</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/COMPU-METHOD-REF</code>	<code>COMPU-METHOD-REF</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/INVALID-VALUE</code>	<code>INVALID-VALUE</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/SW-CALIBRATION-ACCESS</code>	<code>SW-CALIBRATION-ACCESS</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/SW-IMPL-POLICY</code>	<code>SW-IMPL-POLICY</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/UNIT-REF</code>	<code>UNIT-REF</code>



Target model property	Source model property
SW-DATA-DEF-PROPS-CONDITIONAL/IN-VALID-VALUE	INVALID-VALUE

8.1.1.8.58.1. No upgrade of source model parameters DATA-CONSTR-REF, DISPLAY-FORMAT, SW-CALPRM-AXIS-SET, SW-CLASS-ATTR-IMPL-REF, SW-CLASS-REF, SW-CODE-SYNTAX-REF, SW-DATA-DEPENDENCY, SW-HOST-VARIABLE, SW-POINTER, SW-RECORD-LAYOUT-REF, SW-TEXT-PROPS, SW-VALUE-BLOCK-SIZE, SW-VARIABLE-ACCESS-IMPL-POLICY

The source model parameters DATA-CONSTR-REF, DISPLAY-FORMAT, SW-CALPRM-AXIS-SET, SW-CLASS-ATTR-IMPL-REF, SW-CLASS-REF, SW-CODE-SYNTAX-REF, SW-DATA-DEPENDENCY, SW-HOST-VARIABLE, SW-POINTER, SW-RECORD-LAYOUT-REF, SW-TEXT-PROPS, SW-VALUE-BLOCK-SIZE, SW-VARIABLE-ACCESS-IMPL-POLICY are not upgraded to the target model.

8.1.1.8.58.2. No configuration of target model parameters SW-ALIGNMENT, SW-COMPARISON-VARIABLES, SW-DATA-DEPENDENCY, SW-HOST-VARIABLE, SW-INTENDED-RESOLUTION, SW-INTERPOLATION-METHOD, SW-IS-VIRTUAL, SW-POINTER-TARGET-PROPS, SW-REFRESH-TIMING, SW-TEXT-PROPS, VALUE-AXIS-DATA-TYPE-REF

The target model parameters SW-ALIGNMENT, SW-COMPARISON-VARIABLES, SW-DATA-DEPENDENCY, SW-HOST-VARIABLE, SW-INTENDED-RESOLUTION, SW-INTERPOLATION-METHOD, SW-IS-VIRTUAL, SW-POINTER-TARGET-PROPS, SW-REFRESH-TIMING, SW-TEXT-PROPS, VALUE-AXIS-DATA-TYPE-REF are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.58.3. Configuration of SW-DATA-DEF-PROPS-CONDITIONAL/SW-IMPL-POLICY

If the source model SW-DATA-DEF-PROPS element is aggregated within a DATA-ELEMENT-PROTOTYPE which has its IS-QUEUED value set to true, the upgraded SW-DATA-DEF-PROPS-CONDITIONAL/SW-IMPL-POLICY contains the value QUEUED, else it contains the SW-IMPL-POLICY value of the source model SW-DATA-DEF-PROPS counterpart. If the source model SW-DATA-DEF-PROPS value is MESSAGE, the parameter is not configured at all in the target model since it does not support this value.

8.1.1.8.58.4. Creation of SW-DATA-DEF-PROPS elements in the target model

For every target model ARGUMENT-DATA-PROTOTYPE, PARAMETER-DATA-PROTOTYPE, or VARIABLE-DATA-PROTOTYPE which has a source model counterpart which has its IS-QUEUED parameter value set to true but does not aggregate an SW-DATA-DEF-PROPS element, a dedicated SW-DATA-DEF-PROPS is created in the target model which has its SW-IMPL-POLICY value set to QUEUED.



8.1.1.8.59. IMPLEMENTATION-DATA-TYPE

Source model entities: ARRAY-TYPE, BOOLEAN-TYPE, CHAR-TYPE, INTEGER-TYPE, OPAQUE-TYPE, REAL-TYPE, RECORD-TYPE, STRING-TYPE

Target model entity: IMPLEMENTATION-DATA-TYPE

8.1.1.8.59.1. Creation of AUTOSAR 4.0 platform data types

Before any source model data type is upgraded into the target model, the AUTOSAR 4.0 platform data types are created, see [Section 8.1.4, “AUTOSAR 4.0 platform data types”](#) for details.

8.1.1.8.59.2. Configuration of target model IMPLEMENTATION-DATA-TYPE/CATEGORY platform data types

If the source model element is an ARRAY-TYPE or a STRING-TYPE, CATEGORY is set to ARRAY. For RECORD-TYPES, CATEGORY is set to STRUCTURE. In all other cases, CATEGORY is set to TYPE_REFERENCE.

8.1.1.8.59.3. Configuration of target model DATA-CONSTR

If the source model data type element contains a LOWER-LIMIT or an UPPER-LIMIT parameter, one DATA-CONSTR element is created in the target model which contains the limits. The target model IMPLEMENTATION-DATA-TYPE element references the DATA-CONSTR via SW-DATA-DEF-PROPS/SW-DATA-DEF-PROPS-VARIANTS/SW-DATA-DEF-PROPS-CONDITIONAL/DATA-CONSTR-REF.

If the source model data type is an integral data type that additionally contains an INVALID-VALUE that does not lie within the interval set up by LOWER-LIMIT and UPPER-LIMIT, the corresponding target model limits are adapted. Thus the target model value range includes the INVALID-VALUE.

8.1.1.8.59.4. Assignment of AUTOSAR 4.0 platform data type to target model IMPLEMENTATION-DATA-TYPE

If the source model data type element is one of BOOLEAN-TYPE, CHAR-TYPE, INTEGER-TYPE, OPAQUE-TYPE, or REAL-TYPE, a matching AUTOSAR 4.0 platform data type is determined according to the source model data type properties LOWER-LIMIT, UPPER-LIMIT (extensions of the limits as described in [Section 8.1.1.8.59.3, “Configuration of target model DATA-CONSTR”](#) are taken into account), ENCODING. The platform data type is referenced via SW-DATA-DEF-PROPS/SW-DATA-DEF-PROPS-VARIANTS/SW-DATA-DEF-PROPS-CONDITIONAL/IMPLEMENTATION-DATA-TYPE-REF



8.1.1.8.59.5. ARRAY-TYPE

If the source model data type is an ARRAY-TYPE, the following additional mapping rules apply:

Target model property	Source model property
SUB-ELEMENTS/IMPLEMENTATION-DATA-TYPE-ELEMENT/SHORT-NAME	ELEMENT/SHORT-NAME
SUB-ELEMENTS/IMPLEMENTATION-DATA-TYPE-ELEMENT/ARRAY-SIZE	ELEMENT/MAX-NUMBER-OF-ELEMENTS

The parameter SUB-ELEMENTS/IMPLEMENTATION-DATA-TYPE-ELEMENT/ARRAY-SIZE-SEMANTICS is set to FIXED-SIZE.

8.1.1.8.59.6. RECORD-TYPE

If the source model data type is a RECORD-TYPE, the following additional mapping rules apply:

Target model property	Source model property
SUB-ELEMENTS/IMPLEMENTATION-DATA-TYPE-ELEMENT	ELEMENTS/RECORD-ELEMENT

8.1.1.8.60. COMPU-METHOD

Source model entity: COMPU-METHOD

Target model entity: COMPU-METHOD

Target model property	Source model property
COMPU-INTERNAL-TO-PHYS/COMPU-SCALES/COMPU-SCALE	COMPU-INTERNAL-TO-PHYS/COMPU-SCALES/COMPU-SCALE
COMPU-PHYS-TO-INTERNAL/COMPU-SCALES/COMPU-SCALE	COMPU-PHYS-TO-INTERNAL/COMPU-SCALES/COMPU-SCALE
DISPLAY-FORMAT	DISPLAY-FORMAT
UNIT-REF	UNIT-REF

8.1.1.8.61. COMPU-SCALE

Source model entity: COMPU-SCALE



Target model entity: COMPU-SCALE

Target model property	Source model property
LOWER-LIMIT	LOWER-LIMIT
UPPER-LIMIT	UPPER-LIMIT
COMPU-CONST	COMPU-CONST
COMPU-INVERSE-VALUE	No source model parameter is available.
COMPU-RATIONAL-COEFFS / COMPU-NUMERATOR	COMPU-RATIONAL-COEFFS / COMPU-NUMERATOR
COMPU-RATIONAL-COEFFS / COMPU-DENOMINATOR	COMPU-RATIONAL-COEFFS / COMPU-DENOMINATOR
MASK	No source model parameter is available.
SHORT-LABEL	SHORT-LABEL
SYMBOL	No source model parameter is available.

8.1.1.8.62. EXCLUSIVE-AREA

Source model entity: EXCLUSIVE-AREA

Target model entity: EXCLUSIVE-AREA

Apart from the standard parameters SHORT-NAME, CATEGORY, and UUID, no further parameters are upgraded.

8.1.1.8.63. COM-MGR-USER-NEEDS

Source model entities: COM-MGR-USER-NEEDS, SWC-COM-MGR-USER-NEEDS

Target model entity: COM-MGR-USER-NEEDS

Target model property	Source model property
MAX-COMM-MODE	MAX-COMM-MODE

8.1.1.8.63.1. No upgrade of source model SWC-COM-MGR-USER-NEEDS/CALLBACK-PORTS and SWC-COM-MGR-USER-NEEDS/SERVICE-CALL-PORTS

The source model parameters SWC-COM-MGR-USER-NEEDS/CALLBACK-PORTS and SWC-COM-MGR-USER-NEEDS/SERVICE-CALL-PORTS are not upgraded into the target model because there are no corresponding parameters.



8.1.1.8.64. DIAGNOSTIC-COMMUNICATION-MANAGER-NEEDS

Source model entities: DIAGNOSTIC-COMMUNICATION-NEEDS, SWC-DIAGNOSTIC-COMMUNICATION-NEEDS

Target model entity: DIAGNOSTIC-COMMUNICATION-MANAGER-NEEDS

Apart from the standard parameters SHORT-NAME, CATEGORY, and UUID, no further parameters are upgraded.

8.1.1.8.64.1. No upgrade of source model SWC-DIAGNOSTIC-COMMUNICATION-NEEDS/CALLBACK-PORTS and SWC-DIAGNOSTIC-COMMUNICATION-NEEDS/SERVICE-CALL-PORT

The source model parameters SWC-DIAGNOSTIC-COMMUNICATION-NEEDS/CALLBACK-PORTS and SWC-DIAGNOSTIC-COMMUNICATION-NEEDS/SERVICE-CALL-PORT are not upgraded into the target model because there are no corresponding parameters.

8.1.1.8.65. DIAGNOSTIC-EVENT-NEEDS

Source model entities: DIAGNOSTIC-EVENT-NEEDS, SWC-DIAGNOSTIC-EVENT-NEEDS

Target model entity: DIAGNOSTIC-EVENT-NEEDS

Apart from the standard parameters SHORT-NAME, CATEGORY, and UUID, no further parameters are upgraded.

8.1.1.8.65.1. No upgrade of source model SWC-DIAGNOSTIC-EVENT-NEEDS/CALLBACK-PORTS and SWC-DIAGNOSTIC-EVENT-NEEDS/SERVICE-CALL-PORTS

The source model parameters SWC-DIAGNOSTIC-EVENT-NEEDS/CALLBACK-PORTS and SWC-DIAGNOSTIC-EVENT-NEEDS/SERVICE-CALL-PORTS are not upgraded into the target model because there are no corresponding parameters.

8.1.1.8.65.2. No configuration of target model parameters CONSIDER-PTO-STATUS, DIAG-EVENT-DEBOUNCE-ALGORITHM, DIAG-REQUIREMENT, DTC-KIND, DTC-NUMBER, INHIBITING-FID-REF

The target model parameters CONSIDER-PTO-STATUS, DIAG-EVENT-DEBOUNCE-ALGORITHM, DIAG-REQUIREMENT, DTC-KIND, DTC-NUMBER, INHIBITING-FID-REF are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.66. ECU-STATE-MGR-USER-NEEDS

Source model entities: ECU-STATE-MGR-USER-NEEDS, SWC-ECU-STATE-MGR-USER-NEEDS



Target model entity: ECU-STATE-MGR-USER-NEEDS

Apart from the standard parameters SHORT-NAME, CATEGORY, and UUID, no further parameters are upgraded.

8.1.1.8.66.1. No upgrade of source model SWC-ECU-STATE-MGR-USER-NEEDS/SERVICE-CALL-PORTS

The source model parameter SWC-ECU-STATE-MGR-USER-NEEDS/SERVICE-CALL-PORTS is not upgraded into the target model because there is no corresponding parameter.

8.1.1.8.66.2. No configuration of target model parameters CONSIDER-PTO-STATUS, DIAG-EVENT-DEBOUNCE-ALGORITHM, DIAG-REQUIREMENT, DTC-KIND, DTC-NUMBER, INHIBITING-FID-REF

The target model parameters CONSIDER-PTO-STATUS, DIAG-EVENT-DEBOUNCE-ALGORITHM, DIAG-REQUIREMENT, DTC-KIND, DTC-NUMBER, INHIBITING-FID-REF are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.67. FUNCTION-INHIBITION-NEEDS

Source model entities: FUNCTION-INHIBITION-NEEDS, SWC-FUNCTION-INHIBITION-NEEDS

Target model entity: FUNCTION-INHIBITION-NEEDS

Apart from the standard parameters SHORT-NAME, CATEGORY, and UUID, no further parameters are upgraded.

8.1.1.8.67.1. No upgrade of source model SWC-FUNCTION-INHIBITION-NEEDS/SERVICE-CALL-PORTS

The source model parameter SWC-FUNCTION-INHIBITION-NEEDS/SERVICE-CALL-PORTS is not upgraded into the target model because there is no corresponding parameter.

8.1.1.8.68. NV-BLOCK-NEEDS

Source model entities: BLOCK-NEEDS, SWC-NV-BLOCK-NEEDS

Target model entity: NV-BLOCK-NEEDS

Target model property	Source model property
N-DATA-SETS	(SWC-) NV-BLOCK-NEEDS/N-DATA-SETS
READONLY	(SWC-) NV-BLOCK-NEEDS/READONLY



Target model property	Source model property
RESISTANT-TO-CHANGED-SW	(SWC-) NV-BLOCK-NEEDS/RESISTANT-TO-CHANGED-SW
RESTORE-AT-START	(SWC-) NV-BLOCK-NEEDS/RESTORE-AT-START
WRITE-ONLY-ONCE	(SWC-) NV-BLOCK-NEEDS/WRITE-ONLY-ONCE
WRITING-FREQUENCY	(SWC-) NV-BLOCK-NEEDS/WRITING-FREQUENCY
WRITING-PRIORITY	(SWC-) NV-BLOCK-NEEDS/WRITING-PRIORITY

8.1.1.8.68.1. No upgrade of source model SWC-NV-BLOCK-NEEDS/CALL-BACK-PORTS, SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF, SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF, SWC-NV-BLOCK-NEEDS/SERVICE-CALL-PORTS

The source model parameters SWC-NV-BLOCK-NEEDS/CALL-BACK-PORTS, SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF, SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF, SWC-NV-BLOCK-NEEDS/SERVICE-CALL-PORTS are not upgraded into the target model because there are no corresponding parameters.

8.1.1.8.68.2. No configuration of target model parameters CALC-RAM-BLOCK-CRC, CHECK-STATIC-BLOCK-ID, N-ROM-BLOCKS, STORE-AT-SHUTDOWN, WRITE-VERIFICATION

The target model parameters CALC-RAM-BLOCK-CRC, CHECK-STATIC-BLOCK-ID, N-ROM-BLOCKS, STORE-AT-SHUTDOWN, WRITE-VERIFICATION are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.68.3. No configuration of target model parameter RELIABILITY

The source model parameter RELIABILITY is not upgraded to the target model parameter RELIABILITY because the parameter values are not related.

8.1.1.8.68.4. Additional configuration of ASSIGNED-DATAS/ROLE-BASED-DATA-ASSIGNMENT for SWC-NV-BLOCK-NEEDS elements

For every SWC-NV-BLOCK-NEEDS element in the source model which either contains a valid DEFAULT-BLOCK_REF or a valid MIRROR-BLOCK-REF, one ASSIGNED-DATAS/ROLE-BASED-DATA-ASSIGNMENT element is created in the target model. The ROLE-BASED-DATA-ASSIGNMENT element is aggregated within the target model SWC-SERVICE-DEPENDENCY element which also aggregates the target model SWC-NV-BLOCK-NEEDS element. ROLE-BASED-DATA-ASSIGNMENT/ROLE is set to ramMirror. ROLE-BASED-DATA-ASSIGNMENT/USED-PIM-REF references the target model counterpart of the source model PER-INS-



TANCE-MEMORY which is referenced by SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF. ROLE-BASED-DATA-ASSIGNMENT/USED-PARAMETER-ELEMENT aggregates a AUTOSAR-PARAMETER-REF element which in turn references in LOCAL-PARAMETER-REF the target model counterpart of the source model CALPRM-ELEMENT-PROTOTYPE which is referenced by SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF.

8.1.1.8.69. OBD-CONTROL-SERVICE-NEEDS

Source model entities: OBD-CONTROL-SERVICE-NEEDS, SWC-OBD-CONTROL-SERVICE-NEEDS

Target model entity: OBD-CONTROL-SERVICE-NEEDS

Target model property	Source model property
TEST-ID	OBD-CONTROL-SERVICE-NEEDS/TEST-ID

8.1.1.8.69.1. No upgrade of source model SWC-OBD-CONTROL-SERVICE-NEEDS/CALLBACK-PORT

The source model parameter SWC-OBD-CONTROL-SERVICE-NEEDS/CALLBACK-PORT is not upgraded into the target model because there is no corresponding parameter.

8.1.1.8.69.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL

The target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.70. OBD-INFO-SERVICE-NEEDS

Source model entities: OBD-INFO-SERVICE-NEEDS, SWC-OBD-INFO-SERVICE-NEEDS

Target model entity: OBD-INFO-SERVICE-NEEDS

Target model property	Source model property
DATA-LENGTH	OBD-INFO-SERVICE-NEEDS/DATA-LENGTH
INFO-TYPE	OBD-INFO-SERVICE-NEEDS/INFO-TYPE

8.1.1.8.70.1. No upgrade of source model SWC-OBD-INFO-SERVICE-NEEDS/CALLBACK-PORT

The source model parameter SWC-OBD-INFO-SERVICE-NEEDS/CALLBACK-PORT is not upgraded into the target model because there is no corresponding parameter.



8.1.1.8.70.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL

The target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.71. OBD-PID-SERVICE-NEEDS

Source model entities: OBD-PID-SERVICE-NEEDS, SWC-OBD-PID-SERVICE-NEEDS

Target model entity: OBD-PID-SERVICE-NEEDS

Target model property	Source model property
DATA-LENGTH	OBD-PID-SERVICE-NEEDS/DATA-LENGTH
PARAMETER-ID	OBD-PID-SERVICE-NEEDS/PARAMETER-ID
STANDARD	OBD-PID-SERVICE-NEEDS/STANDARD

8.1.1.8.71.1. No upgrade of source model SWC-OBD-PID-SERVICE-NEEDS/CALLBACK-PORT

The source model parameter SWC-OBD-PID-SERVICE-NEEDS/CALLBACK-PORT is not upgraded into the target model because there is no corresponding parameter.

8.1.1.8.71.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL

The target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.72. OBD-RATIO-SERVICE-NEEDS

Source model entities: OBD-RATIO-SERVICE-NEEDS, SWC-OBD-RATIO-SERVICE-NEEDS

Target model entity: OBD-RATIO-SERVICE-NEEDS

Target model property	Source model property
CONNECTION-TYPE	OBD-RATIO-SERVICE-NEEDS/CONNECTION-TYPE
IUMPR-GROUP	OBD-RATIO-SERVICE-NEEDS/IUMPR-GROUP
RATE-BASED-MONITORED-EVENT-REF	OBD-RATIO-SERVICE-NEEDS/RATE-BASED-MONITORED-EVENT-REF



Target model property	Source model property
USED-FID-REF	OBD-RATIO-SERVICE-NEEDS/USED-FID-REF

8.1.1.8.72.1. No upgrade of source model SWC-OBD-RATIO-SERVICE-NEEDS/SERVICE-CALL-PORT

The source model parameter **SWC-OBD-RATIO-SERVICE-NEEDS/SERVICE-CALL-PORT** is not upgraded into the target model because there is no corresponding parameter.

8.1.1.8.72.2. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL

The target model parameters **AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL** are not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.73. SUPERVISED-ENTITY-NEEDS

Source model entities: **SUPERVISED-ENTITY-NEEDS, SWC-SUPERVISED-ENTITY-NEEDS**

Target model entity: **SUPERVISED-ENTITY-NEEDS**

Target model property	Source model property
ACTIVATE-AT-START	SUPERVISED-ENTITY-NEEDS/ACTIVATE-AT-START
ENABLE-DEACTIVATION	SUPERVISED-ENTITY-NEEDS/ENABLE-DEACTIVATION
EXPECTED-ALIVE-CYCLE	SUPERVISED-ENTITY-NEEDS/EXPECTED-ALIVE-CYCLE
MAX-ALIVE-CYCLE	SUPERVISED-ENTITY-NEEDS/MAX-ALIVE-CYCLE
MIN-ALIVE-CYCLE	SUPERVISED-ENTITY-NEEDS/MIN-ALIVE-CYCLE
TOLERATED-FAILED-CYCLES	SUPERVISED-ENTITY-NEEDS/TOLERATED-FAILED-CYCLES

8.1.1.8.73.1. No configuration of target model parameters AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL

The target model parameters **AUDIENCES, DIAG-REQUIREMENT, SECURITY-ACCESS-LEVEL** are not configured during the model upgrade because there is no source model counterpart.



8.1.1.8.74. PER-INSTANCE-MEMORY

Source model entity: PER-INSTANCE-MEMORY

Target model entity: PER-INSTANCE-MEMORY

Target model property	Source model property
TYPE	TYPE
TYPE-DEFINITION	TYPE-DEFINITION

8.1.1.8.74.1. Configuration of target model parameter INIT-VALUE

The target model parameter INIT-VALUE is not configured during the model upgrade because there is no source model counterpart.

8.1.1.8.75. PER-INSTANCE-MEMORY-SIZE

Source model entity: PER-INSTANCE-MEMORY-SIZE

Target model entity: PER-INSTANCE-MEMORY-SIZE

Target model property	Source model property
ALIGNMENT	ALIGNMENT
PER-INSTANCE-MEMORY-REF	PER-INSTANCE-MEMORY-REF
SIZE	SIZE

8.1.1.8.76. SWC-IMPLEMENTATION

Source model entity: SWC-IMPLEMENTATION

Target model entity: SWC-IMPLEMENTATION

Target model property	Source model property
BEHAVIOR-REF	BEHAVIOR-REF
PER-INSTANCE-MEMORY-SIZES / PER-INSTANCE-MEMORY-SIZE	PER-INSTANCE-MEMORY-SIZES / PER-INSTANCE-MEMORY-SIZE
REQUIRED-RTE-VENDOR	REQUIRED-RTE-VENDOR



Target model property	Source model property
CODE-DESCRIPTORS/CODE	CODE-DESCRIPTORS/CODE
USED-CODE-GENERATOR	CODE-GENERATOR
PROGRAMMING-LANGUAGE	PROGRAMMING-LANGUAGE
SW-VERSION	SW-MAJOR-VERSION.SW-MINOR-VERSION.SW-PATCH-VERSION
VENDOR-ID	VENDOR-ID

8.1.1.8.76.1. No upgrade of source model parameters COMPILERS/COMPILER, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-FILE, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-LIBRARY, LINKERS/LINKER, PROCESSOR-REFS/PROCESSOR-REF, RESOURCE-CONSUMPTION

The source model parameters COMPILERS/COMPILER, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-FILE, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-LIBRARY, LINKERS/LINKER, PROCESSOR-REFS/PROCESSOR-REF, RESOURCE-CONSUMPTION are not upgraded into the target model.

8.1.1.8.76.2. Upgrade of source model parameter PROGRAMMING-LANGUAGE

If the source model parameter PROGRAMMING-LANGUAGE is not defined, C is configured as default value in the target model.

8.1.1.8.76.3. Upgrade of source model parameter VENDOR-ID

If the source model parameter VENDOR-ID is not defined, 1 is configured as default value in the target model.

8.1.1.8.76.4. Creation of target model RESOURCE-CONSUMPTION

For each SWC-IMPLEMENTATION in the target model, one RESOURCE-CONSUMPTION sub element is set up.

8.1.1.8.77. CODE

Source model entity: CODE

Target model entity: CODE



Apart from the standard parameter `SHORT-NAME`, no further parameter is directly upgraded into the target model entity.

8.1.1.8.77.1. Configuration of source model parameter `TYPE`

Every target model `CODE` element aggregates an `ARTIFACT-DESCRIPTORS/AUTOSAR-ENGINEERING-OBJECT` sub element. Its `SHORT-NAME` is set to `EngObject`. The source model parameter `TYPE` is upgraded into `ARTIFACT-DESCRIPTORS/AUTOSAR-ENGINEERING-OBJECT/CATEGORY`. If the source model value is `SRC`, the target model value is set to `SWSRC`. If the source model value is `OBJ`, the target model value is set to `SWOBJ`.

8.1.1.8.78. MODE-DECLARATION

Source model entity: `MODE-DECLARATION`

Target model entity: `MODE-DECLARATION`

Apart from the standard parameters `SHORT-NAME`, `CATEGORY`, and `UUID`, no further parameters are upgraded.

8.1.1.8.79. MODE-DECLARATION-GROUP

Source model entity: `MODE-DECLARATION-GROUP`

Target model entity: `MODE-DECLARATION-GROUP`

Target model property	Source model property
<code>INITIAL-MODE-REF</code>	<code>INITIAL-MODE-REF</code>
<code>MODE-DECLARATIONS/MODE-DECLARATION</code>	<code>MODE-DECLARATIONS/MODE-DECLARATION</code>

8.1.1.8.80. DISABLED-MODE-IREF

Source model entity: `DEPENDENT-ON-MODE-IREF`

Target model entity: `DISABLED-MODE-IREF`

Target model property	Source model property
<code>CONTEXT-PORT-REF</code>	<code>R-PORT-PROTOTYPE-REF</code>
<code>CONTEXT-MODE-DECLARATION-GROUP-PROTOTYPE-REF</code>	<code>MODE-DECLARATION-GROUP-PROTOTYPE-REF</code>
<code>MODE-DECLARATION-REF</code>	<code>MODE-DECLARATION-REF</code>



8.1.1.8.81. MODE-SWITCHED-ACK

Source model entity: MODE-SWITCHED-ACK

Target model entity: MODE-SWITCHED-ACK

Target model property	Source model property
TIMEOUT	TIMEOUT

8.1.1.8.82. MODE-ACCESS-POINT

Target model MODE-ACCESS-POINT elements are created out of source models in two scenarios.

8.1.1.8.82.1. Configuration MODE-ACCESS-POINT out of source model R-PORT-PROTOTYPE elements

For every R-PORT-PROTOTYPE referencing a SENDER-RECEIVER-INTERFACE which aggregates a MODE-DECLARATION-GROUP-PROTOTYPE element, one MODE-ACCESS-POINT element is created in every target model RUNNABLE-ENTITY element related to the target model ATOMIC-SW-COMPONENT which owns the target model counterpart of the source model R-PORT-PROTOTYPE.

In this case, MODE-ACCESS-POINT/MODE-GROUP-IREF/R-MODE-GROUP-IN-ATOMIC-SWC-INSTANCE-REF/CONTEXT-R-PORT-REF contains a reference to the target model counterpart of the source model R-PORT-PROTOTYPE and MODE-ACCESS-POINT/MODE-GROUP-IREF/R-MODE-GROUP-IN-ATOMIC-SWC-INSTANCE-REF/TARGET-MODE-GROUP-REF references the target model counterpart of the source model MODE-DECLARATION-GROUP-PROTOTYPE.

8.1.1.8.82.2. Configuration MODE-ACCESS-POINT out of source model MODE-SWITCH-POINT elements

For every source model MODE-SWITCH-POINT aggregated in a RUNNABLE-ENTITY, one MODE-ACCESS-POINT is created in the target model. The target model MODE-ACCESS-POINT is aggregated in the target model counterpart of the source model RUNNABLE-ENTITY aggregating the source model RUNNABLE-ENTITY.

In this case, MODE-ACCESS-POINT/MODE-GROUP-IREF/P-MODE-GROUP-IN-ATOMIC-SWC-INSTANCE-REF/CONTEXT-P-PORT-REF contains a reference to the target model counterpart of the source model P-PORT-PROTOTYPE referenced by the source model MODE-SWITCH-POINT. MODE-ACCESS-POINT/MODE-GROUP-IREF/P-MODE-GROUP-IN-ATOMIC-SWC-INSTANCE-REF/TARGET-MODE-GROUP-REF references the target model counterpart of the source model MODE-DECLARATION-GROUP-PROTOTYPE referenced by the source model MODE-SWITCH-POINT.



8.1.1.8.83. MODE-SWITCH-POINT

Source model entity: MODE-SWITCH-POINT

Target model entity: MODE-SWITCH-POINT

Target model property	Source model property
MODE-GROUP-IREF/CONTEXT-P-PORT-REF	MODE-GROUP-IREF/P-PORT-PROTOTYPE-REF
MODE-GROUP-IREF/TARGET-MODE-GROUP-REF	MODE-GROUP-IREF/MODE-DECLARATION-GROUP-PROTOTYPE-REF

8.1.1.8.84. SW-ADDR-METHOD

Source model entity: SW-ADDR-METHOD

Target model entity: SW-ADDR-METHOD

Apart from the standard parameters SHORT-NAME, CATEGORY, and UUID, no further parameters are upgraded.

8.1.1.8.85. MEMORY-SECTION

Source model entity: MEMORY-SECTION

Target model entity: MEMORY-SECTION

Target model property	Source model property
ALIGNMENT	ALIGNMENT
SIZE	SIZE

8.1.1.8.85.1. Configuration of source model parameters SW-ADDR-METHOD-REFS/SW-ADDR-METHOD-REF

Since the target model MEMORY-SECTION only supports one single SW-ADDRMETHOD-REF, only the first source model SW-ADDR-METHOD-REFS/SW-ADDR-METHOD-REF is upgraded in the target model.

8.1.1.8.86. SYSTEM-MAPPING

Source model entity: SYSTEM-MAPPING

Target model entity: SYSTEM-MAPPING



Target model property	Source model property
DATA-MAPPINGS/CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING	DATA-MAPPINGS/CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING
DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING	DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING
DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-MAPPING	DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-MAPPING
SW-MAPPINGS/SWC-TO-ECU-MAPPING	SW-MAPPINGS/SWC-TO-ECU-MAPPING
SW-IMPL-MAPPINGS/SWC-TO-IMPL-MAPPING	SW-IMPL-MAPPINGS/SWC-TO-IMPL-MAPPING

8.1.1.8.86.1. No upgrade of source model parameters ECU-RESOURCE-MAPPINGS , MAPPING-CONSTRAINTS , RESOURCE-ESTIMATIONS and SIGNAL-PATH-CONSTRAINTS

Due to internal data model restrictions, the source model parameters ECU-RESOURCE-MAPPINGS , MAPPING-CONSTRAINTS , RESOURCE-ESTIMATIONS and SIGNAL-PATH-CONSTRAINTS are not upgraded to their target model counterparts.

8.1.1.8.87. SWC-TO-ECU-MAPPING

Source model entity: SWC-TO-ECU-MAPPING

Target model entity: SWC-TO-ECU-MAPPING

Target model property	Source model property
COMPONENT-IREFS/COMPONENT-IREF/CONTEXT-COMPOSITION-REF	COMPONENT-IREFS/COMPONENT-IREF/SOFT-WARE-COMPOSITION-REF
COMPONENT-IREFS/COMPONENT-IREF/CONTEXT-COMPONENT-REF	COMPONENT-IREFS/COMPONENT-IREF/COMPONENT-PROTOTYPE-REF
COMPONENT-IREFS/COMPONENT-IREF/TARGET-COMPONENT-REF	COMPONENT-IREFS/COMPONENT-IREF/TARGET-COMPONENT-PROTOTYPE-REF
ECU-INSTANCE-REF	ECU-INSTANCE-REF

8.1.1.8.87.1. No configuration of target model parameters PARTITION-REF and PROCESSING-UNIT-REF

The target model parameters PARTITION-REF and PROCESSING-UNIT-REF are not configured because there are no source model counterparts.



8.1.1.8.88. SWC-TO-IMPL-MAPPING

Source model entity: SWC-TO-IMPL-MAPPING

Target model entity: SWC-TO-IMPL-MAPPING

Target model property	Source model property
COMPONENT-IREFS/COMPONENT-IREF/CONTEXT-COMPOSITION-REF	COMPONENT-IREFS/COMPONENT-IREF/SOFT-WARE-COMPOSITION-REF
COMPONENT-IREFS/COMPONENT-IREF/CONTEXT-COMPONENT-REF	COMPONENT-IREFS/COMPONENT-IREF/COMPONENT-PROTOTYPE-REF
COMPONENT-IREFS/COMPONENT-IREF/TARGET-COMPONENT-REF	COMPONENT-IREFS/COMPONENT-IREF/TARGET-COMPONENT-PROTOTYPE-REF
COMPONENT-IMPLEMENTATION-REF	COMPONENT-IMPLEMENTATION-REF

8.1.1.8.89. SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING

Source model entity: SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING

Target model entity: SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING

Target model property	Source model property
DATA-ELEMENT-IREF/CONTEXT-POSITION-REF	DATA-ELEMENT-IREF/SOFTWARE-POSITION-REF
DATA-ELEMENT-IREF/CONTEXT-COMPONENT-REF	DATA-ELEMENT-IREF/COMPONENT-PROTOTYPE-REF
DATA-ELEMENT-IREF/CONTEXT-PORT-REF	DATA-ELEMENT-IREF/PORT-PROTOTYPE-REF
DATA-ELEMENT-IREF/TARGET-DATA-PROTOTYPE-REF	DATA-ELEMENT-IREF/DATA-ELEMENT-REF
SIGNAL-GROUP-REF	SIGNAL-GROUP-REF
TYPE-MAPPING/SENDER-REC-RECORD-TYPE-MAPPING	TYPE-MAPPING/SENDER-REC-RECORD-TYPE-MAPPING
TYPE-MAPPING/SENDER-REC-ARRAY-TYPE-MAPPING	TYPE-MAPPING/SENDER-REC-ARRAY-TYPE-MAPPING

8.1.1.8.90. SENDER-RECEIVER-TO-SIGNAL-MAPPING

Source model entity: SENDER-RECEIVER-TO-SIGNAL-MAPPING



Target model entity: SENDER-RECEIVER-TO-SIGNAL-MAPPING

Target model property	Source model property
DATA-ELEMENT-IREF/CONTEXT-COM-POSITION-REF	DATA-ELEMENT-IREF/SOFTWARE-COM-POSITION-REF
DATA-ELEMENT-IREF/CONTEXT-COMPONENT-REF	DATA-ELEMENT-IREF/COMPONENT-PROTO-TYPE-REF
DATA-ELEMENT-IREF/CONTEXT-PORT-REF	DATA-ELEMENT-IREF/PORT-PROTOTYPE-REF
DATA-ELEMENT-IREF/TARGET-DATA-PROTO-TYPE-REF	DATA-ELEMENT-IREF/DATA-ELEMENT-REF
SYSTEM-SIGNAL-REF	SIGNAL-REF

8.1.1.8.91. CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING

Source model entity: CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING

Target model entity: CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING

Target model property	Source model property
APPLICATION-ERROR/SYSTEM-SIGNAL-REF	APPLICATION-ERRORS/APPLICATION-ER-ROR-MAPPING/SYSTEM-SIGNAL-REF
CLIENT-ID	CLIENT-ID
COMPOSITE-TYPE-MAPPINGS/CLIENT-SERVER-ARRAY-TYPE-MAPPING	COMPOSITE-TYPE-MAPPINGS/CLIENT-SERVER-ARRAY-TYPE-MAPPING
COMPOSITE-TYPE-MAPPINGS/CLIENT-SERVER-RECORD-TYPE-MAPPING	COMPOSITE-TYPE-MAPPINGS/CLIENT-SERVER-RECORD-TYPE-MAPPING
EMPTY-SIGNAL	EMPTY-SIGNAL
MAPPED-OPERATION-IREF/CONTEXT-COMPONENT-REF	MAPPED-OPERATION-IREF/COMPONENT-PROTO-TYPE-REF
MAPPED-OPERATION-IREF/CONTEXT-COM-POSITION-REF	MAPPED-OPERATION-IREF/COMPOSITION-PROTOTYPE-REF
MAPPED-OPERATION-IREF/TARGET-OPERATION-REF	MAPPED-OPERATION-IREF/OPERATION-REF
MAPPED-OPERATION-IREF/CONTEXT-PORT-REF	MAPPED-OPERATION-IREF/PORT-PROTOTYPE-REF
PRIMITIVE-TYPE-MAPPINGS/CLIENT-SERVER-PRIMITIVE-TYPE-MAPPING/ARGUMENT-REF	PRIMITIVE-TYPE-MAPPINGS/CLIENT-SERVER-PRIMITIVE-TYPE-MAPPING/ARGUMENT-REF



Target model property	Source model property
PRIMITIVE-TYPE-MAPPINGS/CLIENT-SERVER-PRIMITIVE-TYPE-MAPPING/SYSTEM-SIGNAL-REF	PRIMITIVE-TYPE-MAPPINGS/CLIENT-SERVER-PRIMITIVE-TYPE-MAPPING/SYSTEM-SIGNAL-REF
REQUEST-GROUP-REF	REQUEST-GROUP-REF
RESPONSE-GROUP-REF	RESPONSE-GROUP-REF
SEQUENCE-COUNTER/SYSTEM-SIGNAL-REF	SEQUENCE-COUNTER/SYSTEM-SIGNAL-REF

8.1.1.8.92. SENDER-REC-RECORD-TYPE-MAPPING

Source model entity: SENDER-REC-RECORD-TYPE-MAPPING

Target model entity: SENDER-REC-RECORD-TYPE-MAPPING

Target model property	Source model property
SENDER-REC-RECORD-TYPE-MAPPING/RECORD-ELEMENT-MAPPINGS/SENDER-REC-RECORD-ELEMENT-MAPPING	SENDER-REC-RECORD-TYPE-MAPPING/RECORD-ELEMENT-MAPPINGS/SENDER-REC-RECORD-ELEMENT-MAPPING

8.1.1.8.93. SENDER-REC-RECORD-ELEMENT-MAPPING

Source model entity: SENDER-REC-RECORD-ELEMENT-MAPPING

Target model entity: SENDER-REC-RECORD-ELEMENT-MAPPING

Target model property	Source model property
COMPLEX-TYPE-MAPPING/SENDER-REC-ARRAY-TYPE-MAPPING	COMPLEX-TYPE-MAPPING/SENDER-REC-ARRAY-TYPE-MAPPING
COMPLEX-TYPE-MAPPING/SENDER-REC-RECORD-TYPE-MAPPING	COMPLEX-TYPE-MAPPING/SENDER-REC-RECORD-TYPE-MAPPING
IMPLEMENTATION-RECORD-ELEMENT-REF	RECORD-ELEMENT-REF
SYSTEM-SIGNAL-REF	SIGNAL-REF

8.1.1.8.94. SENDER-REC-ARRAY-TYPE-MAPPING

Source model entity: SENDER-REC-ARRAY-TYPE-MAPPING



Target model entity: SENDER-REC-ARRAY-TYPE-MAPPING

Target model property	Source model property
ARRAY-ELEMENT-MAPPINGS / SENDER-REC-ARRAY-ELEMENT-MAPPING	ARRAY-ELEMENT-MAPPINGS / SENDER-REC-ARRAY-ELEMENT-MAPPING

8.1.1.8.95. SENDER-REC-ARRAY-ELEMENT-MAPPING

Source model entity: SENDER-REC-ARRAY-ELEMENT-MAPPING

Target model entity: SENDER-REC-ARRAY-ELEMENT-MAPPING

Target model property	Source model property
COMPLEX-TYPE-MAPPING / SENDER-REC-ARRAY-TYPE-MAPPING	COMPLEX-TYPE-MAPPING / SENDER-REC-ARRAY-TYPE-MAPPING
COMPLEX-TYPE-MAPPING / SENDER-REC-RECORD-TYPE-MAPPING	COMPLEX-TYPE-MAPPING / SENDER-REC-RECORD-TYPE-MAPPING
INDEXED-ARRAY-ELEMENT	INDEXED-ARRAY-ELEMENT
SYSTEM-SIGNAL-REF	SIGNAL-REF

8.1.1.8.96. END-TO-END-PROTECTION-SET

Source model entity: END-TO-END-PROTECTION-SET

Target model entity: END-TO-END-PROTECTION-SET

Target model property	Source model property
END-TO-END-PROTECTIONS / END-TO-END-PROTECTION	END-TO-END-PROTECTIONS / END-TO-END-PROTECTION

NOTE

End-to-End Protection is not part of the AUTOSAR 3.1.x standard



Since End-to-End Protection is not part of the AUTOSAR 3.1.x standard, the END-TO-END-PROTECTION-SET and its sub elements are only migrated in customer-specific models which extend AUTOSAR 3.1.x in such a way that End-to-End Protection is available.

8.1.1.8.97. END-TO-END-PROTECTION

Source model entity: END-TO-END-PROTECTION



Target model entity: END-TO-END-PROTECTION

Target model property	Source model property
END-TO-END-PROFILE/COUNTER-OFFSET	END-TO-END-PROFILE/COUNTER-OFFSET
END-TO-END-PROFILE/CRC-OFFSET	END-TO-END-PROFILE/CRC-OFFSET
END-TO-END-PROFILE/DATA-ID-MODE	END-TO-END-PROFILE/DATA-ID-MODE
END-TO-END-PROFILE/DATA-IDS/DATA-ID	END-TO-END-PROFILE/DATA-IDS/DATA-ID
END-TO-END-PROFILE/DATA-LENGTH	END-TO-END-PROFILE/DATA-LENGTH
END-TO-END-PROFILE/MAX-DELTA-COUNTER-INIT	END-TO-END-PROFILE/MAX-DELTA-COUNTER-INIT
END-TO-END-PROTECTION-I-SIGNAL-I-PDUS/ END-TO-END-PROTECTION-I-SIGNAL-I-PDU	END-TO-END-PROTECTION-I-SIGNAL-I-PDUS/ END-TO-END-PROTECTION-I-SIGNAL-I-PDU
END-TO-END-PROTECTION-VARIABLE-PROTOTYPES/END-TO-END-PROTECTION-VARIABLE-PROTOTYPE	END-TO-END-PROTECTION-DATA-ELEMENT-PROTOTYPES/END-TO-END-PROTECTION-DATA-ELEMENT-PROTOTYPE

8.1.1.8.98. END-TO-END-PROTECTION-I-SIGNAL-I-PDU

Source model entity: END-TO-END-PROTECTION-I-SIGNAL-I-PDU

Target model entity: END-TO-END-PROTECTION-I-SIGNAL-I-PDU

Target model property	Source model property
DATA-OFFSET	DATA-OFFSET
I-SIGNAL-GROUP-REF	I-SIGNAL-GROUP-REF
I-SIGNAL-I-PDU-REF	I-SIGNAL-I-PDU-REF

8.1.1.8.99. END-TO-END-PROTECTION-VARIABLE-PROTOTYPE

Source model entity: END-TO-END-PROTECTION-DATA-ELEMENT-PROTOTYPE

Target model entity: END-TO-END-PROTECTION-VARIABLE-PROTOTYPE

Target model property	Source model property
RECEIVER-IREFS/RECEIVER-IREF/CONTEXT-COMPONENT-REF	RECEIVER-IREFS/RECEIVER-IREF/COMPONENT-PROTOTYPE-REF



Target model property	Source model property
RECEIVER-IREFS/RECEIVER-IREF/CONTEXT-PORT-REF	RECEIVER-IREFS/RECEIVER-IREF/PORT-PROTOTYPE-REF
RECEIVER-IREFS/RECEIVER-IREF/TARGET-DATA-PROTOTYPE-REF	RECEIVER-IREFS/RECEIVER-IREF/TARGET-DATA-PROTOTYPE-REF
SENDER-IREF/CONTEXT-COMPONENT-REF	SENDER-IREF/COMPONENT-PROTOTYPE-REF
SENDER-IREF/CONTEXT-PORT-REF	SENDER-IREF/PORT-PROTOTYPE-REF
SENDER-IREF/TARGET-DATA-PROTOTYPE-REF	SENDER-IREF/TARGET-DATA-PROTOTYPE-REF

8.1.2. Upgrading a system model from AUTOSAR 3.2.x to AUTOSAR 4.0.3

This section describes how an AUTOSAR 3.2.x system model is upgraded to AUTOSAR 4.0.3.

Since the AUTOSAR 3.1.x and 3.2.x source models and the destination system models 4.0.2 and 4.0.3 are similar, this chapter only describes those parts of the upgrade that differ from the 3.1.x to 4.0.2 upgrade. For all other parts, see [Section 8.1.1, “Upgrading a system model from AUTOSAR 3.1.x to AUTOSAR 4.0.2”](#).

If not stated explicitly, all entities in the 4.0.x target model are located in the same package as they were in the 3.1.x source model.

If not specified explicitly otherwise, the parameters `SHORT-NAME`, `CATEGORY`, and `UUID` of Identifiable entities are upgraded directly into their counterparts in the target model.

8.1.2.1. Topology

8.1.2.1.1. SYSTEM

Source model entity: `SYSTEM`

Target model entity: `SYSTEM`

Target model property	Source model property
<code>SYSTEM-VERSION</code>	<code>SYSTEM-VERSION</code>
<code>ECU-EXTRACT-VERSION</code>	<code>ECU-EXTRACT-VERSION</code>



Target model property	Source model property
FIBEX-ELEMENT-REFS	FIBEX-ELEMENT-REFS
PNC-VECTOR-LENGTH	PNC-VECTOR-LENGTH
PNC-VECTOR-OFFSET	PNC-VECTOR-OFFSET
MAPPINGS/SYSTEM-MAPPING	MAPPING
SYSTEM/ROOT-SOFTWARE-COMPOSITIONS/ROOT-SW-COMPOSITION-PROTOTYPE	SYSTEM/SOFTWARE-COMPOSITION

8.1.2.1.1.1. Changed or moved parameters

FIBEX-ELEMENT-REFS

In FIBEX-ELEMENT-REFS, references to the following types of source model entities are upgraded to their target model counterparts: CAN-CLUSTER, COMMUNICATION-CLUSTER, DCM-I-PDU, ECU-INSTANCE, FLEXRAY-CLUSTER, FRAME, GATEWAY, I-PDU-GROUP, I-SIGNAL, LIN-CLUSTER, MULTIPLEXED-I-PDU, N-PDU, NM-PDU, PDUR-I-PDU-GROUP, SIGNAL-I-PDU, SUBSTITUTION-FRAME, USER-DEFINED-I-PDU, USER-DEFINED-PDU.

8.1.2.1.2. ECU-INSTANCE

Source model entity: ECU-INSTANCE

Target model entity: ECU-INSTANCE

Target model property	Source model property
ASSOCIATED-COM-I-PDU-GROUP-REFS/ASSOCIATED-COM-I-PDU-GROUP-REF	ASSOCIATED-I-PDU-GROUP-REFS/ASSOCIATED-I-PDU-GROUP-REF
ASSOCIATED-PDUR-I-PDU-GROUP-REFS/ASSOCIATED-PDUR-I-PDU-GROUP-REF	ASSOCIATED-PDUR-I-PDU-GROUP-REFS/ASSOCIATED-PDUR-I-PDU-GROUP-REF
COM-CONFIGURATION-GW-TIME-BASE	COM-PROCESSING-PERIOD-GW
COM-CONFIGURATION-RX-TIME-BASE	COM-PROCESSING-PERIOD-RX
COM-CONFIGURATION-TX-TIME-BASE	COM-PROCESSING-PERIOD
COMM-CONTROLLERS	COMM-CONTROLLERS
CONNECTORS	CONNECTORS
DIAGNOSTIC-ADDRESS	DIAGNOSTIC-ADDRESS
SLEEP-MODE-SUPPORTED	SLEEP-MODE-SUPPORTED
WAKE-UP-OVER-BUS-SUPPORTED	WAKE-UP-OVER-BUS-SUPPORTED



Target model property	Source model property
No target model parameter is available.	COM-CONFIGURATION-ID
No target model parameter is available.	PDU-R-CONFIGURATION-ID
Not upgraded.	RESPONSE-ADDRESSS/RESPONSE-ADDRESS

8.1.2.1.2.1. Reason for not upgraded source model parameter

RESPONSE-ADDRESSS/RESPONSE-ADDRESS

The parameter RESPONSE-ADDRESSS/RESPONSE-ADDRESS is not upgraded to the target model since the contained diagnostic addresses cannot be uniquely assigned to concrete interfaces.

8.1.2.1.3. CAN-CLUSTER

Source model entity: CAN-CLUSTER

Target model entity: CAN-CLUSTER

Target model property	Source model property
CAN-CLUSTER-CONDITIONAL/PHYSICAL-CHAN-NELS	PHYSICAL-CHANNELS
CAN-CLUSTER-CONDITIONAL/PROTOCOL-NAME	PROTOCOL-NAME
CAN-CLUSTER-CONDITIONAL/PROTOCOL-VERSION	PROTOCOL-VERSION
CAN-CLUSTER-CONDITIONAL/SPEED	SPEED/1000
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT

8.1.2.1.3.1. Changed or moved parameters

NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED

The source model parameters NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED are configured in the target model entity NM-ECU, see [Section 8.1.2.7.2, "NM-ECU"](#).

SPEED

In the target model, the parameter SPEED is given in kbit/s, whereas the source model provides it in bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.



NM-BUS-LOAD-REDUCTION-ACTIVE, NM-WAIT-BUS-SLEEP-TIME, NM-IMMEDIATE-NM-CYCLE-TIME, NM-IMMEDIATE-NM-TRANSMISSIONS, NM-MSG-CYCLE-TIME, and NM-REMOTE-SLEEP-INDICATION-TIME

The parameters NM-BUS-LOAD-REDUCTION-ACTIVE, NM-WAIT-BUS-SLEEP-TIME, NM-IMMEDIATE-NM-CYCLE-TIME, NM-IMMEDIATE-NM-TRANSMISSIONS, NM-MSG-CYCLE-TIME, and NM-REMOTE-SLEEP-INDICATION-TIME are configured as NM-BUSLOAD-REDUCTION-ACTIVE, NM-WAIT-BUS-SLEEP-TIME, NM-IMMEDIATE-NM-CYCLE-TIME, NM-IMMEDIATE-NM-TRANSMISSIONS, NM-MSG-CYCLE-TIME, and NM-REMOTE-SLEEP-INDICATION-TIME in the target model entity CAN-NM-CLUSTER, see [Section 8.1.2.7.3, "CAN-NM-CLUSTER"](#).

NM-BUS-LOAD-REDUCTION-ENABLED

The parameter NM-BUS-LOAD-REDUCTION-ENABLED is configured as NM-BUSLOAD-REDUCTION-ENABLED in the target model entity CAN-NM-CLUSTER-COUPLING, see [Section 8.1.1.7.8, "FLEXRAY-NM-CLUSTER-COUPLING"](#).

NM-LOWER-CAN-ID and NM-UPPER-CAN-ID

The parameters NM-LOWER-CAN-ID and NM-UPPER-CAN-ID are configured in the target model entity CAN-NM-RANGE-CONFIG parameters LOWER-CAN-ID and UPPER-CAN-ID, which is aggregated in CAN-NM-NODE, see [Section 8.1.1.7.4, "CAN-NM-NODE"](#).

8.1.2.1.4. CAN-COMMUNICATION-CONTROLLER

Source model entity: CAN-COMMUNICATION-CONTROLLER

Target model entity: CAN-COMMUNICATION-CONTROLLER

Target model property	Source model property
CATEGORY	CATEGORY
CAN-COMMUNICATION-CONTROLLER-CONDITION-AL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/SYNC-JUMP-WIDTH	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/SYNC-JUMP-WIDTH
CAN-COMMUNICATION-CONTROLLER-CONDITION-AL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-1	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-1
CAN-COMMUNICATION-CONTROLLER-CONDITION-AL/CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-2	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/TIME-SEG-2
No target model parameter is available.	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/NUMBER-OF-SAMPLES
No target model parameter is available.	CAN-CONTROLLER-ATTRIBUTES/CAN-CONTROLLER-CONFIGURATION/PROPAGATION-DELAY



Target model property	Source model property
CAN-COMMUNICATION-CONTROLLER/CAN-CONTROLLER-CONFIGURATION-REQUIREMENTS	No source model parameter is available.

8.1.2.1.4.1. Changed or moved parameters

NM-MSG-CYCLE-OFFSET and NM-MSG-REDUCED-TIME

The source model parameters NM-MSG-CYCLE-OFFSET and NM-MSG-REDUCED-TIME are configured in the target model entity NM-NODE, see [Section 8.1.2.7.2, "NM-ECU"](#).

NM-USER-DATA-ENABLED

The source model parameter NM-USER-DATA-ENABLED is configured in the target model entity NM-ECU, see [Section 8.1.2.7.2, "NM-ECU"](#).

8.1.2.1.5. CAN-COMMUNICATION-CONNECTOR

Source model entity: CAN-COMMUNICATION-CONNECTOR

Target model entity: CAN-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
PNC-WAKEUP-CAN-ID	PNC-WAKEUP-CAN-ID
PNC-WAKEUP-CAN-ID-EXTENDED	PNC-WAKEUP-CAN-ID-EXTENDED
PNC-WAKEUP-CAN-ID-MASK	PNC-WAKEUP-CAN-ID-MASK
PNC-WAKEUP-DATA-MASK	PNC-WAKEUP-DATA-MASK
PNC-WAKEUP-DLC	PNC-WAKEUP-DLC
PNC-GATEWAY-TYPE	PNC-GATEWAY-TYPE

8.1.2.1.5.1. Changed or moved parameters

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity CAN-PHYSICAL-CHANNEL.



NM-ADDRESS

The source model parameter NM-ADDRESS is configured in the target model entity CAN-NM-NODE, see [Section 8.1.2.7.4, “CAN-NM-NODE”](#).

8.1.2.1.6. CAN-PHYSICAL-CHANNEL

Source model entities: PHYSICAL-CHANNEL, COMMUNICATION-CONNECTOR

Target model entity: CAN-PHYSICAL-CHANNEL

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-TRIGGERINGS	FRAME-TRIGGERINGSS
PDU-TRIGGERINGS	I-PDU-TRIGGERINGS
I-SIGNAL-TRIGGERINGS	I-SIGNAL-TRIGGERINGS

8.1.2.1.6.1. Changed or moved parameters

COMM-CONNECTORS

The target model parameter COMM-CONNECTORS is configured by retrieving all target model COMMUNICATION-CONNECTOR entities that meet the following condition: The source model counterpart of the COMMUNICATION-CONNECTOR references the source model counterpart of the upgraded CAN-PHYSICAL-CHANNEL via CHANNEL-REF.

CAN-TP-CONNECTION-CHANNEL

The CAN-TP-CONNECTION-CHANNEL entities aggregated in TP-CHANNELS are upgraded to CAN-TP-CONNECTION and CAN-TP-CHANNEL entities in the target model, aggregated in CAN-TP-CONFIG, see [Section 8.1.2.4.1, “CAN-TP-CONFIG”](#).

TP-ADDRESS

The TP-ADDRESS entities aggregated in TP-ADDRESSSS are upgraded to CAN-TP-ADDRESS entities in the target model, aggregated in CAN-TP-CONFIG, see [Section 8.1.2.4.1, “CAN-TP-CONFIG”](#).

CAN-TP-NODE

The CAN-TP-NODE entities aggregated in TP-NODES are upgraded to CAN-TP-NODE entities in the target model, which are aggregated in CAN-TP-CONFIG, see [Section 8.1.2.4.1, “CAN-TP-CONFIG”](#).

8.1.2.1.7. FLEXRAY-CLUSTER

Source model entity: FLEXRAY-CLUSTER



Target model entity: FLEXRAY-CLUSTER/FLEXRAY-CLUSTER-CONDITIONAL

Target model property	Source model property
PHYSICAL-CHANNELS	PHYSICAL-CHANNELS
PROTOCOL-NAME	PROTOCOL-NAME
PROTOCOL-VERSION	PROTOCOL-VERSION
SPEED	SPEED/1000
ACTION-POINT-OFFSET	ACTION-POINT-OFFSET
BIT	BIT
CAS-RX-LOW-MAX	CAS-RX-LOW-MAX
COLD-START-ATTEMPTS	COLD-START-ATTEMPTS
CYCLE	CYCLE
CYCLE-COUNT-MAX	CYCLE-COUNT-MAX
DYNAMIC-SLOT-IDLE-PHASE	DYNAMIC-SLOT-IDLE-PHASE
LISTEN-NOISE	LISTEN-NOISE
MACRO-PER-CYCLE	MACRO-PER-CYCLE
MACROTICK-DURATION	MACROTICK-DURATION
MAX-WITHOUT-CLOCK-CORRECTION-FATAL	MAX-WITHOUT-CLOCK-CORRECTION-FATAL
MAX-WITHOUT-CLOCK-CORRECTION-PASSIVE	MAX-WITHOUT-CLOCK-CORRECTION-PASSIVE
MINISLOT-ACTION-POINT-OFFSET	MINISLOT-ACTION-POINT-OFFSET
MINISLOT-DURATION	MINISLOT-DURATION
NETWORK-IDLE-TIME	NETWORK-IDLE-TIME
NETWORK-MANAGEMENT-VECTOR-LENGTH	NETWORK-MANAGEMENT-VECTOR-LENGTH
NUMBER-OF-MINISLOTS	NUMBER-OF-MINISLOTS
NUMBER-OF-STATIC-SLOTS	NUMBER-OF-STATIC-SLOTS
OFFSET-CORRECTION-START	OFFSET-CORRECTION-START
PAYLOAD-LENGTH-STATIC	PAYLOAD-LENGTH-STATIC
SAMPLE-CLOCK-PERIOD	SAMPLE-CLOCK-PERIOD
STATIC-SLOT-DURATION	STATIC-SLOT-DURATION
SYMBOL-WINDOW	SYMBOL-WINDOW
SYMBOL-WINDOW-ACTION-POINT-OFFSET	SYMBOL-WINDOW-ACTION-POINT-OFFSET
SYNC-FRAME-ID-COUNT-MAX	SYNC-FRAME-ID-COUNT-MAX
TRANSMISSION-START-SEQUENCE-DURATION	TRANSMISSION-START-SEQUENCE-DURATION



Target model property	Source model property
WAKEUP-RX-IDLE	WAKEUP-RX-IDLE
WAKEUP-RX-LOW	WAKEUP-RX-LOW
WAKEUP-RX-WINDOW	WAKEUP-RX-WINDOW
WAKEUP-TX-ACTIVE	WAKEUP-TX-ACTIVE
WAKEUP-TX-IDLE	WAKEUP-TX-IDLE
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT
No target model parameter is available.	BUS-GUARDIAN-ENABLE-PART
No target model parameter is available.	CAS-RX-LOW-MIN
No target model parameter is available.	MACRO-INITIAL-OFFSET
No target model parameter is available.	MAX-INITIALISATION-ERROR
No target model parameter is available.	MAX-PROPAGATION-DELAY
No target model parameter is available.	MIN-PROPAGATION-DELAY
No target model parameter is available.	OFFSET-CORRECTION-MAX
DETECT-NIT-ERROR	No source model parameter is available.
IGNORE-AFTER-TX	IGNORE-AFTER-TX
SAFETY-MARGIN	No source model parameter is available.

8.1.2.1.7.1. Changed or moved parameters

NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED

The source model parameters NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED are configured in the target model entity NM-ECU, see [Section 8.1.2.7.2, “NM-ECU”](#).

SPEED

In the target model, the parameter SPEED is given in kBit/s, whereas the source model provides it in Bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.

NM-DATA-CYCLE, NM-READY-SLEEP-COUNT, NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPETAT-MESSAGE-TIME, NM-REPETITION-CYCLE, NM-VOTING-CYCLE

The source model parameters NM-DATA-CYCLE, NM-READY-SLEEP-COUNT, NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPETAT-MESSAGE-TIME, NM-REPETITION-CYCLE, and NM-VOTING-CYCLE are configured in the target model entity FLEXRAY-NM-CLUSTER, see [Section 8.1.2.7.6, “FLEXRAY-NM-CLUSTER”](#).



8.1.2.1.7.2. Reason for not upgraded source model parameter

MAX-FRAME-LENGTH, BUS-GUARDIAN-ENABLE-PART, CAS-RX-LOW-MIN, CAS-RX-LOW-MIN, MACRO-INITIAL-OFFSET, MAX-INITIALISATION-ERROR, MAX-PROPAGATION-DELAY, MIN-PROPAGATION-DELAY, OFFSET-CORRECTION-MAX, NM-REPEAT-MESSAGE-SUPPORT

The parameters MAX-FRAME-LENGTH, BUS-GUARDIAN-ENABLE-PART, CAS-RX-LOW-MIN, CAS-RX-LOW-MIN, MACRO-INITIAL-OFFSET, MAX-INITIALISATION-ERROR, MAX-PROPAGATION-DELAY, MIN-PROPAGATION-DELAY, OFFSET-CORRECTION-MAX, NM-REPEAT-MESSAGE-SUPPORT are not upgraded to the target model, because they do not contribute any data required for configuring an ECU. Therefore, they have been removed from the target model.

8.1.2.1.8. FLEXRAY-COMMUNICATION-CONTROLLER

Source model entity: FLEXRAY-COMMUNICATION-CONTROLLER

Target model entity: FLEXRAY-COMMUNICATION-CONTROLLER/FLEXRAY-COMMUNICATION-CONTROLLER-CONDITIONAL

Target model property	Source model property
CATEGORY	CATEGORY
ACCEPTED-STARTUP-RANGE	ACCEPTED-STARTUP-RANGE
ALLOW-HALT-DUE-TO-CLOCK	ALLOW-HALT-DUE-TO-CLOCK
ALLOW-PASSIVE-TO-ACTIVE	ALLOW-PASSIVE-TO-ACTIVE
CLUSTER-DRIFT-DAMPING	CLUSTER-DRIFT-DAMPING
DECODING-CORRECTION	DECODING-CORRECTION
DELAY-COMPENSATION-A	DELAY-COMPENSATION-A
DELAY-COMPENSATION-B	DELAY-COMPENSATION-B
EXTERN-OFFSET-CORRECTION	EXTERN-OFFSET-CORRECTION
EXTERN-RATE-CORRECTION	EXTERN-RATE-CORRECTION
KEY-SLOT-ID	KEY-SLOT-ID
KEY-SLOT-ONLY-ENABLED	KEY-SLOT-ONLY-ENABLED
KEY-SLOT-USED-FOR-START-UP	KEY-SLOT-USED-FOR-START-UP
KEY-SLOT-USED-FOR-SYNC	KEY-SLOT-USED-FOR-SYNC
LATEST-TX	LATEST-TX
LISTEN-TIMEOUT	LISTEN-TIMEOUT
MACRO-INITIAL-OFFSET-A	MACRO-INITIAL-OFFSET-A
MACRO-INITIAL-OFFSET-B	MACRO-INITIAL-OFFSET-B



Target model property	Source model property
MAXIMUM-DYNAMIC-PAYLOAD-LENGTH	MAXIMUM-DYNAMIC-PAYLOAD-LENGTH
MICRO-INITIAL-OFFSET-A	MICRO-INITIAL-OFFSET-A
MICRO-INITIAL-OFFSET-B	MICRO-INITIAL-OFFSET-B
MICRO-PER-CYCLE	MICRO-PER-CYCLE
MICROTICK-DURATION	MICROTICK-DURATION
OFFSET-CORRECTION-OUT	OFFSET-CORRECTION-OUT
RATE-CORRECTION-OUT	RATE-CORRECTION-OUT
SAMPLES-PER-MICROTICK	SAMPLES-PER-MICROTICK
WAKE-UP-PATTERN	WAKE-UP-PATTERN
No target model parameter is available.	DYNAMIC-SEGMENT-ENABLE
No target model parameter is available.	MICRO-PER-MACRO-NOM
No target model parameter is available.	START-UP-NODE
No target model parameter is available.	SYNC-SLOT
EXTERNAL-SYNC	EXTERNAL-SYNC
FALL-BACK-INTERNAL	FALL-BACK-INTERNAL
FLEXRAY-FIFOS/FLEXRAY-FIFO-CONFIGURATION	FLEXRAY-FIFOS/FLEXRAY-FIFO-CONFIGURATION
NM-VECTOR-EARLY-UPDATE	NM-VECTOR-EARLY-UPDATE
SECOND-KEY-SLOT-ID	SECOND-KEY-SLOT-ID
TWO-KEY-SLOT-MODE	TWO-KEY-SLOT-MODE
No target model parameter is available.	NM-SOURCE-NODE-IDENTIFIER-ENABLED

8.1.2.1.8.1. Changed or moved parameters

NM-CONTROL-BIT-VECTOR-ENABLED

The source model parameter NM-CONTROL-BIT-VECTOR-ENABLED is configured in the target model entity FLEXRAY-NM-CLUSTER, see [Section 8.1.2.7.6, “FLEXRAY-NM-CLUSTER”](#).

NM-NODE-DETECTION-ENABLED, NM-USER-DATA-ENABLED

The source model parameters NM-NODE-DETECTION-ENABLED and NM-USER-DATA-ENABLED are configured in the target model entity NM-ECU, see [Section 8.1.2.7.2, “NM-ECU”](#).



8.1.2.1.8.2. Reason for not upgraded source model parameter

DYNAMIC SEGMENT-ENABLE, MICRO-PER-MACRO-NOM, START-UP-NODE, SYNC-SLOT

The parameters DYNAMIC SEGMENT-ENABLE, MICRO-PER-MACRO-NOM, START-UP-NODE, SYNC-SLOT are not upgraded to the target model, because they do not contribute any data required for configuring an ECU or are redundant with respect to other parameters. Therefore, they have been removed from the target model.

8.1.2.1.9. FLEXRAY-FIFO-CONFIGURATION

Source model entity: FLEXRAY-FIFO-CONFIGURATION

Target model entity: FLEXRAY-FIFO-CONFIGURATION

Target model property	Source model property
ADMIT-WITHOUT-MESSAGE-ID	ADMIT-WITHOUT-MESSAGE-ID
BASE-CYCLE	BASE-CYCLE
CHANNEL-REF	CHANNEL-REFS/CHANNEL-REF
CYCLE-REPETITION	CYCLE-REPETITION
FIFO-DEPTH	FIFO-DEPTH
FIFO-RANGES/FLEXRAY-FIFO-RANGE	FIFO-RANGES/FLEXRAY-FIFO-RANGE
MSG-ID-MASK	MSG-ID-MASK
MSG-ID-MATCH	MSG-ID-MATCH

8.1.2.1.9.1. Changed or moved parameters

CHANNEL-REFS/CHANNEL-REF

If multiple CHANNEL-REFS/CHANNEL-REF exist in the source model, a copy of FLEXRAY-FIFO-CONFIGURATION is created for each CHANNEL-REFS/CHANNEL-REF.

8.1.2.1.10. FLEXRAY-FIFO-RANGE

Source model entity: FLEXRAY-FIFO-RANGE

Target model entity: FLEXRAY-FIFO-RANGE

Target model property	Source model property
RANGE-MAX	RANGE-MAX
RANGE-MIN	RANGE-MIN



8.1.2.1.11. FLEXRAY-COMMUNICATION-CONNECTOR

Source model entity: FLEX-RAY-COMMUNICATION-CONNECTOR

Target model entity: FLEXRAY-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
WAKE-UP-CHANNEL	WAKE-UP-CHANNEL
PNC-GATEWAY-TYPE	PNC-GATEWAY-TYPE

8.1.2.1.11.1. Changed or moved parameters

ECU-COMM-PORT-INSTANCES

In ECU-COMM-PORT-INSTANCES, the aggregated source model entities are upgraded to their target model counterparts: FRAME-PORT, I-PDU-PORT, SIGNAL-PORT.

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity FLEXRAY-PHYSICAL-CHANNEL.

NM-ADDRESS

The source model parameter NM-ADDRESS is configured in the target model entity FLEXRAY-NM-NODE, see [Section 8.1.2.7.7, "FLEXRAY-NM-NODE"](#).

8.1.2.1.12. FLEXRAY-PHYSICAL-CHANNEL

Source model entities: PHYSICAL-CHANNEL, FLEX-RAY-COMMUNICATION-CONNECTOR

Target model entity: FLEXRAY-PHYSICAL-CHANNEL

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-TRIGGERINGS	FRAME-TRIGGERINGSS
PDU-TRIGGERINGS	I-PDU-TRIGGERINGS
I-SIGNAL-TRIGGERINGS	I-SIGNAL-TRIGGERINGS
CHANNEL-NAME	CHANNEL-NAME



8.1.2.1.12.1. Changed or moved parameters

COMM-CONNECTORS

The target model parameter COMM-CONNECTORS is configured by retrieving all target model FLEXRAY-COMMUNICATION-CONNECTOR entities that meet the following condition: The source model counterpart of the FLEXRAY-COMMUNICATION-CONNECTOR references the source model counterpart of the upgraded FLEXRAY-PHYSICAL-CHANNEL via CHANNEL-REF.

TP-ADDRESS

The TP-ADDRESS entities aggregated in TP-ADDRESSSS are upgraded to TP-ADDRESS entities, aggregated in the target model parameter FLEXRAY-TP-CONFIG/TP-ADDRESSSS, see [Section 8.1.2.5.1, "FLEXRAY-TP-CONFIG"](#).

FLEXRAY-ISO-TP-CONNECTION

The FLEXRAY-ISO-TP-CONNECTION entities aggregated in TP-CHANNELS are upgraded to FLEXRAY-TP-NODE entities, which are aggregated in the target model parameter FLEXRAY-TP-CONFIG/TP-NODES, see [Section 8.1.2.5.1, "FLEXRAY-TP-CONFIG"](#).

FLEXRAY-ISO-TP-NODE

The FLEXRAY-ISO-TP-NODE entities aggregated in TP-NODES are upgraded to FLEXRAY-TP-CONNECTION entities, aggregated in the target model parameter FLEXRAY-TP-CONFIG, see [Section 8.1.2.5.1, "FLEXRAY-TP-CONFIG"](#).

FLEXRAY-ISO-TP-PDU-POOL

The FLEXRAY-ISO-TP-PDU-POOL entities aggregated in PDU-POOLS are upgraded to FLEXRAY-TP-PDU-POOL entities, aggregated in the target model parameter FLEXRAY-TP-CONFIG, see [Section 8.1.2.5.1, "FLEXRAY-TP-CONFIG"](#).

FLEXRAY-ISO-TP-CONNECTION-CONTROL

The FLEXRAY-ISO-TP-CONNECTION-CONTROL entities aggregated in TP-CONNECTION-CONTROLS are upgraded to FLEXRAY-TP-CONNECTION-CONTROL entities, aggregated in the target model parameter FLEXRAY-TP-CONFIG/TP-CONNECTION-CONTROLS, see [Section 8.1.2.5.1, "FLEXRAY-TP-CONFIG"](#).

FLEXRAY-ISO-TP-ECU

The FLEXRAY-ISO-TP-ECU entities aggregated in TP-ECUSS are upgraded to FLEXRAY-TP-ECU entities, aggregated in the target model parameter FLEXRAY-TP-CONFIG/TP-ECUS, see [Section 8.1.2.5.1, "FLEXRAY-TP-CONFIG"](#).

8.1.2.1.13. LIN-CLUSTER

The LIN-CLUSTER is upgraded as described in [Section 8.1.1.11, "LIN-CLUSTER"](#).

8.1.2.1.14. LIN-MASTER

The LIN-MASTER is upgraded as described in [Section 8.1.1.12, "LIN-MASTER"](#).



8.1.2.1.15. LIN-SLAVE

The LIN-SLAVE is upgraded as described in [Section 8.1.1.13, “LIN-SLAVE”](#).

8.1.2.1.16. LIN-ERROR-RESPONSE

The LIN-ERROR-RESPONSE is upgraded as described in [Section 8.1.1.14, “LIN-ERROR-RESPONSE”](#).

8.1.2.1.17. LIN-COMMUNICATION-CONNECTOR

Source model entity: COMMUNICATION-CONNECTOR

Target model entity: LIN-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
LIN-CONFIGURABLE-FRAMES/LIN-CON-FIGURABLE-FRAME	No source model parameter is available.
LIN-ORDERED-CONFIGURABLE-FRAMES/LIN-OR-DERED-CONFIGURABLE-FRAME	No source model parameter is available.
PNC-GATEWAY-TYPE	PNC-GATEWAY-TYPE

8.1.2.1.17.1. Changed or moved parameters

ECU-COMM-PORT-INSTANCES

In ECU-COMM-PORT-INSTANCES, the aggregated source model entities are upgraded into their target model counterparts: FRAME-PORT, I-PDU-PORT, SIGNAL-PORT.

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity LIN-PHYSICAL-CHANNEL.

INITIAL-NAD

The target model parameter INITIAL-NAD is configured by taking the CONFIGURED-NAD parameter value of the LIN-SLAVE which is referenced via COMM-CONTROLLER-REF.



8.1.2.1.18. LIN-PHYSICAL-CHANNEL

Source model entities: PHYSICAL-CHANNEL, COMMUNICATION-CONNECTOR

Target model entity: LIN-PHYSICAL-CHANNEL

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-TRIGGERINGS	FRAME-TRIGGERINGSS
PDU-TRIGGERINGS	I-PDU-TRIGGERINGS
I-SIGNAL-TRIGGERINGS	I-SIGNAL-TRIGGERINGS

8.1.2.1.18.1. Changed or moved parameters

COMM-CONNECTORS

The target model parameter COMM-CONNECTORS is configured by retrieving all target model LIN-COMMUNICATION-CONNECTOR entities that meet the following condition: The source model counterpart of the LIN-COMMUNICATION-CONNECTOR references the source model counterpart of the upgraded LIN-PHYSICAL-CHANNEL via CHANNEL-REF.

TP-ADDRESS

The TP-ADDRESS entities aggregated in TP-ADDRESSSS are upgraded to TP-ADDRESS entities, aggregated in the target model parameter LIN-TP-CONFIG, see [Section 8.1.2.6.1, “LIN-TP-CONFIG”](#).

LIN-TP-CHANNEL entities aggregated in TP-CHANNELS

The LIN-TP-CHANNEL entities aggregated in TP-CHANNELS are upgraded to LIN-TP-CONNECTION entities, which are aggregated in the target model parameter LIN-TP-CONFIG, see [Section 8.1.2.6.1, “LIN-TP-CONFIG”](#).

LIN-TP-NODE

The LIN-TP-NODE entities aggregated in TP-NODES are upgraded to LIN-TP-NODE entities, aggregated in the target model parameter LIN-TP-CONFIG, see [Section 8.1.2.6.1, “LIN-TP-CONFIG”](#).

LIN-PHYSICAL-CHANNEL

The target model entity LIN-PHYSICAL-CHANNEL aggregates in its parameter SCHEDULE-TABLES all LIN-SCHEDULE-TABLE entities the source model entity LIN-CLUSTER aggregates in its corresponding parameter.

8.1.2.2. Communication

8.1.2.2.1. CAN-FRAME-TRIGGERING

Source model entity: CAN-FRAME-TRIGGERING



Target model entity: CAN-FRAME-TRIGGERING

Target model property	Source model property
CAN-ADDRESSING-MODE	CAN-ADDRESSING-MODE
CATEGORY	CATEGORY
IDENTIFIER	IDENTIFIER
FRAME-PORT-REFS/FRAME-PORT-REF	FRAME-PORT-REFS/FRAME-PORT-REF
FRAME-REF	FRAME-REF
PDU-TRIGGERINGS/PDU-TRIGGERING-REF-CONDITIONAL	I-PDU-TRIGGERING-REFS/I-PDU-TRIGGERING-REF
RX-IDENTIFIER-RANGE/LOWER-CAN-ID	RX-IDENTIFIER-RANGE/LOWER-CAN-ID
RX-IDENTIFIER-RANGE/UPPER-CAN-ID	RX-IDENTIFIER-RANGE/UPPER-CAN-ID

8.1.2.2. FLEXRAY-FRAME-TRIGGERING

The FLEXRAY-FRAME-TRIGGERING is upgraded as described in [Section 8.1.1.2.2, “FLEXRAY-FRAME-TRIGGERING”](#).

8.1.2.3. FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING

The FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING is upgraded as described in [Section 8.1.1.2.3, “FLEXRAY-ABSOLUTELY-SCHEDULED-TIMING”](#).

8.1.2.4. LIN-FRAME-TRIGGERING

Source model entity: LIN-FRAME-TRIGGERING

Target model entity: LIN-FRAME-TRIGGERING

Target model property	Source model property
CATEGORY	CATEGORY
FRAME-PORT-REFS/FRAME-PORT-REF	FRAME-PORT-REFS/FRAME-PORT-REF
FRAME-REF	FRAME-REF
IDENTIFIER	IDENTIFIER
LIN-CHECKSUM	LIN-CHECKSUM



Target model property	Source model property
PDU-TRIGGERINGS/PDU-TRIGGERING-REF-CONDITIONAL	I-PDU-TRIGGERING-REFS/I-PDU-TRIGGERING-REF

8.1.2.2.4.1. Changed or moved parameters

RELATIVELY-SCHEDULED-TIMING

The source model RELATIVELY-SCHEDULED-TIMING entities aggregated in LIN-FRAME-TRIGGERING/RELATIVELY-SCHEDULED-TIMINGS are now aggregated in the target model entity LIN-SCHEDULE-TABLE, see [Section 8.1.2.2.10, “LIN-SCHEDULE-TABLE”](#).

8.1.2.2.5. CAN-FRAME

CAN-FRAME is upgraded as described in [Section 8.1.1.2.5, “CAN-FRAME”](#).

8.1.2.2.6. FLEXRAY-FRAME

FLEXRAY-FRAME is upgraded as described in [Section 8.1.1.2.6, “FLEXRAY-FRAME”](#).

8.1.2.2.7. LIN-UNCONDITIONAL-FRAME

LIN-UNCONDITIONAL-FRAME is upgraded as described in [Section 8.1.1.2.7, “LIN-UNCONDITIONAL-FRAME”](#).

8.1.2.2.8. LIN-SPORADIC-FRAME

LIN-SPORADIC-FRAME is upgraded as described in [Section 8.1.1.2.8, “LIN-SPORADIC-FRAME”](#).

8.1.2.2.9. LIN-EVENT-TRIGGERED-FRAME

LIN-EVENT-TRIGGERED-FRAME is upgraded as described in [Section 8.1.1.2.9, “LIN-EVENT-TRIGGERED-FRAME”](#).

8.1.2.2.10. LIN-SCHEDULE-TABLE

LIN-SCHEDULE-TABLE is upgraded as described in [Section 8.1.1.2.10, “LIN-SCHEDULE-TABLE”](#).



8.1.2.2.11. ASSIGN-FRAME-ID

ASSIGN-FRAME-ID is upgraded as described in [Section 8.1.1.2.11, “ASSIGN-FRAME-ID”](#).

8.1.2.2.12. UNASSIGN-FRAME-ID

UNASSIGN-FRAME-ID is upgraded as described in [Section 8.1.1.2.12, “UNASSIGN-FRAME-ID”](#).

8.1.2.2.13. ASSIGN-NAD

ASSIGN-NAD is upgraded as described in [Section 8.1.1.2.13, “ASSIGN-NAD”](#).

8.1.2.2.14. FREE-FORMAT

FREE-FORMAT is upgraded as described in [Section 8.1.1.2.14, “FREE-FORMAT”](#).

8.1.2.2.15. PDU-TRIGGERING

PDU-TRIGGERING is upgraded as described in [Section 8.1.1.2.19, “PDU-TRIGGERING”](#).

8.1.2.2.16. I-SIGNAL-TRIGGERING

I-SIGNAL-TRIGGERING is upgraded as described in [Section 8.1.1.2.20, “I-SIGNAL-TRIGGERING”](#).

8.1.2.2.17. DCM-I-PDU

DCM-I-PDU is upgraded as described in [Section 8.1.1.2.21, “DCM-I-PDU”](#).

8.1.2.2.18. I-SIGNAL-I-PDU

I-SIGNAL-I-PDU is upgraded as described in [Section 8.1.1.2.22, “I-SIGNAL-I-PDU”](#).

8.1.2.2.19. MULTIPLEXED-I-PDU

Source model entity: MULTIPLEXED-I-PDU



Target model entity: MULTIPLEXED-I-PDU

Target model property	Source model property
CATEGORY	CATEGORY
DYNAMIC-PARTS/DYNAMIC-PART	DYNAMIC-PART
DYNAMIC-PART-ALTERNATIVES/DYNAMIC-PART-ALTERNATIVE	DYNAMIC-PART-ALTERNATIVES/DYNAMIC-PART-ALTERNATIVE
LENGTH	LENGTH
SELECTOR-FIELD-BYTE-ORDER	SELECTOR-FIELD-BYTE-ORDER
SELECTOR-FIELD-LENGTH	SELECTOR-FIELD-LENGTH
SELECTOR-FIELD-START-POSITION	SELECTOR-FIELD-START-POSITION
STATIC-PARTS/STATIC-PART	STATIC-PART
TRIGGER-MODE	TRIGGER-MODE
UNUSED-BIT-PATTERN	UNUSED-BIT-PATTERN

8.1.2.2.19.1. Changed or moved parameters

LENGTH

During the upgrade, the parameter LENGTH's value given in units of bits in the source model is converted to bytes as required by the target model.

8.1.2.2.20. DYNAMIC-PART

Source model entity: DYNAMIC-PART

Target model entity: DYNAMIC-PART

Target model property	Source model property
DYNAMIC-PART-ALTERNATIVES/DYNAMIC-PART-ALTERNATIVE	DYNAMIC-PART-ALTERNATIVES/DYNAMIC-PART-ALTERNATIVE
SEGMENT-POSITIONS/SEGMENT-POSITION	SEGMENT-POSITIONS/SEGMENT-POSITION

8.1.2.2.21. DYNAMIC-PART-ALTERNATIVE

Source model entity: DYNAMIC-PART-ALTERNATIVE



Target model entity: DYNAMIC-PART-ALTERNATIVE

Target model property	Source model property
I-PDU-REF	I-PDU-REF
SELECTOR-FIELD-CODE	SELECTOR-FIELD-CODE
INITIAL-DYNAMIC-PART	INITIAL-DYNAMIC-PART

8.1.2.22. STATIC-PART

Source model entity: STATIC-PART

Target model entity: STATIC-PART

Target model property	Source model property
I-PDU-REF	I-PDU-REF
SEGMENT-POSITIONS/SEGMENT-POSITION	SEGMENT-POSITIONS/SEGMENT-POSITION

8.1.2.23. SEGMENT-POSITION

Source model entity: SEGMENT-POSITION

Target model entity: SEGMENT-POSITION

Target model property	Source model property
SEGMENT-BYTE-ORDER	SEGMENT-BYTE-ORDER
SEGMENT-LENGTH	SEGMENT-LENGTH
SEGMENT-POSITION	SEGMENT-POSITION

8.1.2.24. N-PDU

Source model entity: N-PDU

Target model entity: N-PDU

Target model property	Source model property
CATEGORY	CATEGORY
LENGTH	LENGTH



8.1.2.2.24.1. Changed or moved parameters

LENGTH

During the upgrade process, the parameter LENGTH's value given in units of bits in the source model is converted to bytes as required by the target model.

8.1.2.2.25. NM-PDU

Source model entity: NM-PDU

Target model entity: NM-PDU

Target model property	Source model property
CATEGORY	CATEGORY
LENGTH	LENGTH
I-SIGNAL-TO-I-PDU-MAPPINGS/I-SIGNAL-TO-I-PDU-MAPPING	I-SIGNAL-TO-I-PDU-MAPPINGS/I-SIGNAL-TO-I-PDU-MAPPING
NM-VOTE-INFORMATION	No source model parameter is available.

8.1.2.2.25.1. Changed or moved parameters

LENGTH

During the upgrade process, the parameter LENGTH's value given in units of bits in the source model is converted to bytes as required by the target model.

NM-CBV-POSITION, NM-NID-POSITION, and NM-USER-DATA-LENGTH

The source model parameters NM-CBV-POSITION, NM-NID-POSITION, and NM-USER-DATA-LENGTH are used to configure parameter values of the target model entity CAN-NM-CLUSTER, see [Section 8.1.2.7.3, "CAN-NM-CLUSTER"](#).

NM-DATA-INFORMATION

The target model parameter NM-DATA-INFORMATION is set to true if the source model parameter NM-USER-DATA-LENGTH is greater than zero or at least one of the optional parameters NM-CBV-POSITION, NM-NID-POSITION is present. Otherwise NM-DATA-INFORMATION is set to false.

8.1.2.2.26. USER-DEFINED-PDU

Source model entity: USER-DEFINED-PDU

Target model entity: USER-DEFINED-PDU



Target model property	Source model property
CDD-TYPE	CDD-TYPE

8.1.2.27. USER-DEFINED-I-PDU

Source model entity: USER-DEFINED-I-PDU

Target model entity: USER-DEFINED-I-PDU

Target model property	Source model property
CDD-TYPE	CDD-TYPE
LENGTH	LENGTH

8.1.2.27.1. Changed or moved parameters

LENGTH

During the upgrade process, the parameter LENGTH's value given in units of bits in the source model is converted to bytes as required by the target model.

8.1.2.28. PDU-TO-FRAME-MAPPING

PDU-TO-FRAME-MAPPING is upgraded as described in [Section 8.1.1.2.30, “PDU-TO-FRAME-MAPPING”](#).

8.1.2.29. I-PDU-TIMING

Source model entity: I-PDU-TIMING-SPECIFICATION

Target model entity: I-PDU-TIMING

Target model property	Source model property
MINIMUM-DELAY	MINIMUM-DELAY
TRANSMISSION-MODE-DECLARATION	TRANSMISSION-MODE-DECLARATION
No target model parameter is available.	REQUEST-CONTROLLED-TIMING

8.1.2.30. I-SIGNAL-I-PDU-GROUP

I-SIGNAL-I-PDU-GROUP is upgraded as described in [Section 8.1.1.2.32, “I-SIGNAL-I-PDU-GROUP”](#).



8.1.2.2.31. PDUR-I-PDU-GROUP

Source model entity: PDUR-I-PDU-GROUP

Target model entity: PDUR-I-PDU-GROUP

Target model property	Source model property
COMMUNICATION-MODE	COMMUNICATION-MODE
I-PDUS/PDU-TRIGGERING-REF-CONDITION-AL/PDU-TRIGGERING-REF	I-PDU-REFS/I-PDU-REF

8.1.2.2.32. I-SIGNAL-TO-I-PDU-MAPPING

I-SIGNAL-TO-I-PDU-MAPPING is upgraded as described in [Section 8.1.1.2.33, “I-SIGNAL-TO-I-PDU-MAPPING”](#).

8.1.2.2.33. I-SIGNAL

I-SIGNAL is upgraded as described in [Section 8.1.1.2.34, “I-SIGNAL”](#).

8.1.2.2.34. I-SIGNAL-GROUP

I-SIGNAL-GROUP is upgraded as described in [Section 8.1.1.2.35, “I-SIGNAL-GROUP”](#).

8.1.2.2.35. SYSTEM-SIGNAL

SYSTEM-SIGNAL is upgraded as described in [Section 8.1.1.2.36, “SYSTEM-SIGNAL”](#).

8.1.2.2.36. I-SIGNAL-PORT

Source model entity: SIGNAL-PORT

Target model entity: I-SIGNAL-PORT

Target model property	Source model property
CATEGORY	CATEGORY
COMMUNICATION-DIRECTION	COMMUNICATION-DIRECTION
TIMEOUT	TIMEOUT
DATA-FILTER	DATA-FILTER



8.1.2.2.37. FRAME-PORT

FRAME-PORT is upgraded as described in [Section 8.1.1.2.38, “FRAME-PORT”](#).

8.1.2.2.38. I-PDU-PORT

I-PDU-PORT is upgraded as described in [Section 8.1.1.2.39, “I-PDU-PORT”](#).

8.1.2.2.39. TRANSMISSION-MODE-DECLARATION

Source model entity: TRANSMISSION-MODE-DECLARATION

Target model entity: TRANSMISSION-MODE-DECLARATION

Target model property	Source model property
TRANSMISSION-MODE-CONDITIONS / TRANSMISSION-MODE-CONDITION	TRANSMISSION-MODE-CONDITIONS / TRANSMISSION-MODE-CONDITION
TRANSMISSION-MODE-FALSE-TIMING	TRANSMISSION-MODE-FALSE-TIMING

8.1.2.2.39.1. Changed or moved parameters

TRANSMISSION-MODE-TRUE-TIMING

All source model timing elements residing in I-PDU-TIMING-SPECIFICATION of the source model SIGNAL-I-PDU are moved into the target model TRANSMISSION-MODE-TRUE-TIMING element.

8.1.2.2.40. TRANSMISSION-MODE-CONDITION

TRANSMISSION-MODE-CONDITION is upgraded as described in [Section 8.1.1.2.41, “TRANSMISSION-MODE-CONDITION”](#).

8.1.2.2.41. TRANSMISSION-MODE-FALSE-TIMING

TRANSMISSION-MODE-FALSE-TIMING is upgraded as described in [Section 8.1.1.2.42, “TRANSMISSION-MODE-FALSE-TIMING”](#).

8.1.2.2.42. EVENT-CONTROLLED-TIMING

EVENT-CONTROLLED-TIMING is upgraded as described in [Section 8.1.1.2.43, “EVENT-CONTROLLED-TIMING”](#).



8.1.2.2.43. CYCLIC-TIMING

CYCLIC-TIMING is upgraded as described in [Section 8.1.1.2.44, “CYCLIC-TIMING”](#).

8.1.2.2.44. TIME-RANGE-TYPE

TIME-RANGE-TYPE is upgraded as described in [Section 8.1.1.2.45, “TIME-RANGE-TYPE”](#).

8.1.2.2.45. ABSOLUTE-TOLERANCE

ABSOLUTE-TOLERANCE is upgraded as described in [Section 8.1.1.2.46, “ABSOLUTE-TOLERANCE”](#).

8.1.2.2.46. RELATIVE-TOLERANCE

RELATIVE-TOLERANCE is upgraded as described in [Section 8.1.1.2.47, “RELATIVE-TOLERANCE”](#).

8.1.2.3. Gateway

8.1.2.3.1. FRAME-MAPPING

FRAME-MAPPING is upgraded as described in [Section 8.1.1.3.1, “FRAME-MAPPING”](#).

8.1.2.3.2. I-PDU-MAPPING

Source model entity: I-PDU-MAPPING

Target model entity: I-PDU-MAPPING

Target model property	Source model property
PDUR-TP-CHUNK-SIZE	PDUR-TP-CHUNK-SIZE
SOURCE-I-PDU-REF	SOURCE-I-PDU/SOURCE-I-PDU-REF
TARGET-I-PDU/TARGET-I-PDU-REF	TARGET-I-PDU/TARGET-I-PDU-REF
No target model parameter is available.	SOURCE-I-PDU-REF/PDU-MAPPINGS-DE-FAULT-VALUE



8.1.2.3.3. TARGET-I-PDU-REF

TARGET-I-PDU-REF is upgraded as described in [Section 8.1.1.3.3, “TARGET-I-PDU-REF”](#).

8.1.2.3.4. PDU-MAPPING-DEFAULT-VALUE

PDU-MAPPING-DEFAULT-VALUE is upgraded as described in [Section 8.1.1.3.4, “PDU-MAPPING-DEFAULT-VALUE”](#).

8.1.2.3.5. DEFAULT-VALUE-ELEMENT

DEFAULT-VALUE-ELEMENT is upgraded as described in [Section 8.1.1.3.5, “DEFAULT-VALUE-ELEMENT”](#).

8.1.2.3.6. I-SIGNAL-MAPPING

I-SIGNAL-MAPPING is upgraded as described in [Section 8.1.1.3.6, “I-SIGNAL-MAPPING”](#).

8.1.2.4. CAN Transport Layer

8.1.2.4.1. CAN-TP-CONFIG

For each CAN-CLUSTER in the source model that has at least one CAN-TP-CONNECTION-CHANNEL aggregated in its PHYSICAL-CHANNEL, one CAN-TP-CONFIG element is created in the target model. The target model element CAN-TP-CONFIG resides in the same AR-PACKAGE as the related CAN-CLUSTER.

8.1.2.4.1.1. New parameters in the target model

CAN-TP-ECU

For each ECU-INSTANCE in the source model that is attached to the PHYSICAL-CHANNEL, one CAN-TP-ECU element is created in the target model. The target model element CAN-TP-ECU references the target model counterpart of the source model ECU-INSTANCE in ECU-INSTANCE-REF. All created CAN-TP-ECU elements are aggregated in CAN-TP-CONFIG/TP-ECUS.

8.1.2.4.1.2. Moved parameters in the target model

CAN-TP-NODE

For each source model CAN-TP-NODE aggregated in a PHYSICAL-CHANNEL of a CAN-CLUSTER for which a CAN-TP-CONFIG has been created in the target model, one CAN-TP-NODE is added. CAN-TP-NODE is



aggregated in CAN-TP-CONFIG/TP-NODES. [Section 8.1.2.4.2, “CAN-TP-NODE”](#) describes the upgrade in detail.

CAN-TP-ADDRESS

For each source model TP-ADDRESS aggregated in a PHYSICAL-CHANNEL of a CAN-CLUSTER for which a CAN-TP-CONFIG has been created in the target model, one CAN-TP-ADDRESS is added. CAN-TP-ADDRESS is aggregated in CAN-TP-CONFIG/TP-ADDRESSSS. [Section 8.1.2.4.3, “CAN-TP-ADDRESS”](#) describes the upgrade in detail.

CAN-TP-CONNECTION

For each source model CAN-TP-CONNECTION-CHANNEL aggregated in a PHYSICAL-CHANNEL of a CAN-CLUSTER for which a CAN-TP-CONFIG has been created in the target model, one CAN-TP-CHANNEL is created and aggregated in CAN-TP-CONFIG/TP-CHANNELS. Depending on the CHANNEL-MODE-TYPE of the CAN-TP-CONNECTION, one or two CAN-TP-CONNECTION-CHANNELS are created in the target model and aggregated in CAN-TP-CONFIG/TP-CONNECTIONS. If the CHANNEL-MODE-TYPE of the source model CAN-TP-CONNECTION is HALF-DUPLEX-MODE, one CAN-TP-CONNECTION-CHANNEL is created in the target model. If the CHANNEL-MODE-TYPE of the source model CAN-TP-CONNECTION is FULL-DUPLEX-MODE, two CAN-TP-CONNECTION-CHANNELS are created. [Section 8.1.2.4.5, “CAN-TP-CONNECTION”](#) and [Section 8.1.2.4.4, “CAN-TP-CHANNEL”](#) describes the upgrade in detail.

8.1.2.4.2. CAN-TP-NODE

Source model entity: CAN-TP-NODE

Target model entity: CAN-TP-NODE

Target model property	Source model property
CATEGORY	CATEGORY
CONNECTOR-REF	CONNECTOR-REFS/CONNECTOR-REF
TP-ADDRESS-REF	TP-ADDRESS-REF
MAX-FC-WAIT	RX-WFT-MAX
TIMEOUT-AR	TIMEOUT-AR

8.1.2.4.2.1. New parameters in the target model

TIMEOUT-AS

The target model parameter TIMEOUT-AS is set to the minimum TIMEOUT-AS value of all source model CAN-TP-CONNECTION-CHANNELS that reference the CAN-TP-NODE as sending node.

ST-MIN

The target model parameter ST-MIN is set to the maximum MINIMUM-SEPARATION-TIME value of all source model CAN-TP-CONNECTION-CHANNELS that reference the CAN-TP-NODE as receiving node.



8.1.2.4.3. CAN-TP-ADDRESS

Source model entity: TP-ADDRESS

Target model entity: CAN-TP-ADDRESS

Target model property	Source model property
CATEGORY	CATEGORY
TP-ADDRESS	TP-ADDRESS

8.1.2.4.4. CAN-TP-CHANNEL

Source model entity: CAN-TP-CONNECTION-CHANNEL

Target model entity: CAN-TP-CHANNEL

Target model property	Source model property
CHANNEL-MODE	CHANNEL-MODE-TYPE

8.1.2.4.5. CAN-TP-CONNECTION

Source model entity: CAN-TP-CONNECTION-CHANNEL

Target model entity: CAN-TP-CONNECTION

Target model property	Source model property
ADDRESSING-FORMAT	ADDRESSING-FORMAT
MAX-BLOCK-SIZE	BLOCK-SIZE
DATA-PDU-REF	DATA-PDU-REF
FLOW-CONTROL-PDU-REF	FLOW-CONTROL-PDU-REF
MAX-BLOCK-SIZE	BLOCK-SIZE
PADDING-ACTIVATION	PADDING-ACTIVATION
TRANSMITTER-REF	SOURCE-REF
TA-TYPE	TA-TYPE
RECEIVER-REFS/RECEIVER-REF	TARGET-REFS/TARGET-REF
TIMEOUT-BR	TIMEOUT-BR



Target model property	Source model property
TIMEOUT-BS	TIMEOUT-BS
TIMEOUT-CR	TIMEOUT-CR
TIMEOUT-CS	TIMEOUT-CS
TP-SDU-REF	TP-SDU-REF
No target model parameter is available.	MULTICAST-ADDRESSING
MULTICAST-REF	MULTICAST-REF

8.1.2.4.5.1. New parameters in the target model

CAN-TP-CHANNEL-REF

Every created CAN-TP-CONNECTION references the CAN-TP-CHANNEL it belongs to in CAN-TP-CHAN-NEL-REF.

8.1.2.4.5.2. Changed or moved parameters

MINIMUM-SEPARATION-TIME

The parameter MINIMUM-SEPARATION-TIME is upgraded to the receiver CAN-TP-NODE, see [Section 8.1.2.4.2, “CAN-TP-NODE”](#).

TIMEOUT-AS

The parameter TIMEOUT-AS is upgraded to the sender CAN-TP-NODE, see [Section 8.1.2.4.2, “CAN-TP-NODE”](#).

8.1.2.4.6. Second CAN-TP-CONNECTION for FULL-DUPLEX-MODE

Source model entity: CAN-TP-CONNECTION-CHANNEL

Target model entity: CAN-TP-CONNECTION

The second target model parameter CAN-TP-CONNECTION is only created for CAN-TP-CONNECTION-CHAN-NELS in FULL-DUPLEX-MODE.

Target model property	Source model property
ADDRESSING-FORMAT	ADDRESSING-FORMAT
MAX-BLOCK-SIZE	BLOCK-SIZE



Target model property	Source model property
FLOW-CONTROL-PDU-REF	DATA-PDU-REF
DATA-PDU-REF	FLOW-CONTROL-PDU-REF
MAX-BLOCK-SIZE	BLOCK-SIZE
PADDING-ACTIVATION	PADDING-ACTIVATION
RECEIVER-REFS/RECEIVER-REF	SOURCE-REF
TA-TYPE	TA-TYPE
TRANSMITTER-REF	TARGET-REFS/TARGET-REF
TIMEOUT-BR	TIMEOUT-BR
TIMEOUT-BS	TIMEOUT-BS
TIMEOUT-CR	TIMEOUT-CR
TIMEOUT-CS	TIMEOUT-CS
No target model parameter is available.	MULTICAST-ADDRESSING
MULTICAST-REF	MULTICAST-REF

8.1.2.5. FlexRay ISO Transport Layer

NOTE



Non-ISO FlexRay Transport Layer upgrade is not supported

The system model upgrade only supports the upgrade of the FlexRay ISO Transport Layer.
The upgrade of the Non-ISO FlexRay Transport Layer variant is *not* supported.

8.1.2.5.1. FLEXRAY-TP-CONFIG

For each FLEXRAY-CLUSTER in the source model that contains at least one FLEXRAY-ISO-TP-CONNECTION in one of its PHYSICAL-CHANNELS, one FLEXRAY-TP-CONFIG element is created in the target model. The target model element FLEXRAY-TP-CONFIG resides in the same AR-PACKAGE as the related FLEXRAY-CLUSTER.

8.1.2.5.1.1. New parameters in the target model

FLEXRAY-TP-ECU

For each ECU-INSTANCE in the source model that is attached to one of the PHYSICAL-CHANNELS, one FLEXRAY-TP-ECU element is created in the target model. In ECU-INSTANCE-REF the FLEXRAY-TP-



ECU element references the target model counterpart of the source model **ECU-INSTANCE**. All created **FLEXRAY-TP-ECU** elements are aggregated in **FLEXRAY-TP-CONFIG/TP-ECUS**. [Section 8.1.2.5.2, “FLEXRAY-TP-ECU”](#) describes the upgrade in detail.

8.1.2.5.1.2. Moved parameters in the target model

FLEXRAY-TP-NODE

For each source model **FLEXRAY-ISO-TP-NODE** that is attached to any **PHYSICAL-CHANNEL** of a **FLEXRAY-CLUSTER**, and for which a **FLEXRAY-TP-CONFIG** has been created in the target model, one **FLEXRAY-TP-NODE** is added. **FLEXRAY-TP-NODE** is aggregated in **FLEXRAY-TP-CONFIG/TP-NODES**. [Section 8.1.2.5.3, “FLEXRAY-TP-NODE”](#) describes the upgrade in detail.

TP-ADDRESS

For each source model **TP-ADDRESS** attached to any **PHYSICAL-CHANNEL** of a **FLEXRAY-CLUSTER** for which a **FLEXRAY-TP-CONFIG** has been created in the target model, one **FLEXRAY-TP-NODE** is added. **TP-ADDRESS** is aggregated in **FLEXRAY-TP-CONFIG/TP-ADDRESS**. [Section 8.1.2.5.4, “TP-ADDRESS”](#) describes the upgrade in detail.

FLEXRAY-TP-CONNECTION

For each source model **FLEXRAY-ISO-TP-CONNECTION** attached to any **PHYSICAL-CHANNEL** of a **FLEXRAY-CLUSTER** for which a **FLEXRAY-TP-CONFIG** has been created in the target model, one **FLEXRAY-TP-CONNECTION** is added. **FLEXRAY-TP-CONNECTION** is aggregated in **FLEXRAY-TP-CONFIG/TP-CONNECTIONS**. [Section 8.1.2.5.5, “FLEXRAY-TP-CONNECTION”](#) describes the upgrade in detail.

FLEXRAY-TP-CONNECTION-CONTROL

For each source model **FLEXRAY-ISO-TP-CONNECTION-CONTROL** attached to any **PHYSICAL-CHANNEL** of a **FLEXRAY-CLUSTER** for which a **FLEXRAY-TP-CONFIG** has been created in the target model, one **FLEXRAY-TP-CONNECTION-CONTROL** is added. **FLEXRAY-TP-CONNECTION-CONTROL** is aggregated in **FLEXRAY-TP-CONFIG/TP-CONNECTION-CONTROLS**. [Section 8.1.2.5.6, “FLEXRAY-TP-CONNECTION-CONTROL”](#) describes the upgrade in detail.

FLEXRAY-TP-PDU-POOL

For each source model **FLEXRAY-ISO-TP-PDU-POOL** attached to any **PHYSICAL-CHANNEL** of a **FLEXRAY-CLUSTER** for which a **FLEXRAY-TP-CONFIG** has been created in the target model, one **FLEXRAY-TP-PDU-POOL** is added. **FLEXRAY-TP-PDU-POOL** is aggregated in **FLEXRAY-TP-CONFIG/PDU-POOLS**. [Section 8.1.2.5.7, “FLEXRAY-TP-PDU-POOL”](#) describes the upgrade in detail.

8.1.2.5.2. FLEXRAY-TP-ECU

Source model entity: **ECU-INSTANCE**

Target model entity: **FLEXRAY-TP-ECU**



Target model property	Source model property
CYCLE-TIME-MAIN-FUNCTION	CYCLE-TIME-MAIN-FUNCTION
FULL-DUPLEX-ENABLED	FULL-DUPLEX-ENABLED

8.1.2.5.3. FLEXRAY-TP-NODE

Source model entity: FLEXRAY-ISO-TP-NODE

Target model entity: FLEXRAY-TP-NODE

Target model property	Source model property
CONNECTOR-REFS/CONNECTOR-REF	CONNECTOR-REFS/CONNECTOR-REF
TP-ADDRESS-REF	TP-ADDRESS-REF

8.1.2.5.4. TP-ADDRESS

Source model entity: TP-ADDRESS

Target model entity: TP-ADDRESS

Target model property	Source model property
TP-ADDRESS	TP-ADDRESS

8.1.2.5.5. FLEXRAY-TP-CONNECTION

Source model entity: FLEXRAY-ISO-TP-CONNECTION

Target model entity: FLEXRAY-TP-CONNECTION

Target model property	Source model property
BANDWIDTH-LIMITATION	BANDWIDTH-LIMITATION
DIRECT-TP-SDU-REF	DIRECT-TP-SDU-REF
MULTICAST-REF	MULTICAST-REF
RECEIVER-REFS/RECEIVER-REF	RECEIVER-REFS/RECEIVER-REF
REVERSED-TP-SDU-REF	REVERSED-TP-SDU-REF
RX-PDU-POOL-REF	RX-PDU-POOL-REF
TP-CONNECTION-CONTROL-REF	TP-CONNECTION-CONTROL-REF



Target model property	Source model property
TRANSMITTER-REF	TRANSMITTER-REF
TX-PDU-POOL-REF	TX-PDU-POOL-REF

8.1.2.5.6. FLEXRAY-TP-CONNECTION-CONTROL

Source model entity: FLEXRAY-ISO-TP-CONNECTION-CONTROL

Target model entity: FLEXRAY-TP-CONNECTION-CONTROL

Target model property	Source model property
ACK-TYPE	ACK-TYPE
MAX-AR	MAX-AR
MAX-AS	MAX-AS
MAX-BUFFER-SIZE	MAX-BUFFER-SIZE
MAX-FC-WAIT	MAX-FC-WAIT
MAX-FR-IF	MAX-FR-IF
MAX-NUMBER-OF-NPDU-PER-CYCLE	MAX-NUMBER-OF-NPDU-PER-CYCLE
MAX-RETRIES	MAX-RETRIES
SEPARATION-CYCLE-EXPONENT	SEPARATION-CYCLE-EXPONENT
TIME-BR	TIME-BR
TIME-BUFFER	TIME-BUFFER
TIME-FR-IF	TIME-FR-IF
TIMEOUT-AR	TIMEOUT-AR
TIMEOUT-AS	TIMEOUT-AS
TIMEOUT-BS	TIMEOUT-BS
TIMEOUT-CR	TIMEOUT-CR

The source model ACK-TYPE parameter value ACK-WITHOUT-RT is not supported in the target model. If this source model parameter value is detected during model upgrade, the corresponding destination parameter is left undefined.

8.1.2.5.7. FLEXRAY-TP-PDU-POOL

Source model entity: FLEXRAY-ISO-TP-PDU-POOL



Target model entity: FLEXRAY-TP-PDU-POOL

Target model property	Source model property
N-PDU-REFS/N-PDU-REF	N-PDU-REFS/N-PDU-REF

8.1.2.6. LIN Transport Layer

8.1.2.6.1. LIN-TP-CONFIG

For each LIN-CLUSTER in the source model that contains at least one LIN-TP-CHANNEL in its PHYSICAL-CHANNEL, one LIN-TP-CONFIG element is created in the target model. LIN-TP-CONFIG resides in the same AR-PACKAGE as the related LIN-CLUSTER.

8.1.2.6.1.1. Moved parameters in the target model

LIN-TP-NODE

For each source model LIN-TP-NODE attached to the PHYSICAL-CHANNEL of a LIN-CLUSTER for which a LIN-TP-CONFIG has been created in the target model, one LIN-TP-NODE is added. LIN-TP-NODE is aggregated in LIN-TP-CONFIG/TP-NODES. [Section 8.1.2.6.1.2, “LIN-TP-NODE”](#) describes the upgrade in detail.

TP-ADDRESS

For each source model TP-ADDRESS attached to the PHYSICAL-CHANNEL of a LIN-CLUSTER for which a LIN-TP-CONFIG has been created in the target model, one TP-ADDRESS is added. TP-ADDRESS is aggregated in LIN-TP-CONFIG/TP-ADDRESSES. [Section 8.1.2.6.1.3, “TP-ADDRESS”](#) describes the upgrade in detail.

LIN-TP-CONNECTION

For each source model LIN-TP-CHANNEL attached to the PHYSICAL-CHANNEL of a LIN-CLUSTER for which a LIN-TP-CONFIG has been created in the target model, one LIN-TP-CONNECTION is added. LIN-TP-CONNECTION is aggregated in LIN-TP-CONFIG/TP-CONNECTIONS. [Section 8.1.2.6.1.4, “LIN-TP-CONNECTION”](#) describes the upgrade in detail.

8.1.2.6.1.2. LIN-TP-NODE

Source model entity: LIN-TP-NODE

Target model entity: LIN-TP-NODE



Target model property	Source model property
CONNECTOR-REF	CONNECTOR-REFS/CONNECTOR-REF
TP-ADDRESS-REF	TP-ADDRESS-REF
MAX-NUMBER-OF-RESP-PENDING-FRAMES	No source model parameter is available.
P-2-MAX	No source model parameter is available.
P-2-TIMING	No source model parameter is available.

8.1.2.6.1.3. TP-ADDRESS

Source model entity: TP-ADDRESS

Target model entity: TP-ADDRESS

Target model property	Source model property
TP-ADDRESS	TP-ADDRESS

8.1.2.6.1.4. LIN-TP-CONNECTION

Source model entity: LIN-TP-CHANNEL

Target model entity: LIN-TP-CONNECTION

Target model property	Source model property
DATA-PDU-REF	DATA-PDU-REF
DROP-NOT-REQUESTED-NAD	DROP-NOT-REQUESTED-NAD
FLOW-CONTROL-REF	FLOW-CONTROL-REF
LIN-TP-N-SDU-REF	LIN-TP-N-SDU-REF
MULTICAST-REF	MULTICAST-REF
TRANSMITTER-REF	SOURCE-REF
RECEIVER-REFS/RECEIVER-REF	TARGET-REFS/TARGET-REF

8.1.2.7. Network Management

8.1.2.7.1. NM-CONFIG

NM-CONFIG is upgraded as described in [Section 8.1.1.7.1, “NM-CONFIG”](#).



8.1.2.7.2. NM-ECU

For every ECU-INSTANCE in the source model, one NM-ECU element is created in the target model. The related ECU-INSTANCE is referenced in ECU-INSTANCE-REF. The NM-ECU is aggregated in NM-CONFIG/NM-IF-ECUS

8.1.2.7.2.1. New parameters in the target model

NM-NODE-DETECTION-ENABLED

NM-NODE-DETECTION-ENABLED is configured by taking the maximum value of NM-NODE-DETECTION-ENABLED of any FLEXRAY-COMMUNICATION-CONTROLLER of the ECU-INSTANCE. If none of these FLEXRAY-COMMUNICATION-CONTROLLER elements has the value set, NM-NODE-DETECTION-ENABLED's maximum value is taken of all COMMUNICATION-CLUSTER elements to which the ECU-INSTANCE is connected.

NM-NODE-ID-ENABLED

NM-NODE-ID-ENABLED is determined by taking NM-NODE-ID-ENABLED's maximum value of all COMMUNICATION-CLUSTER elements to which the ECU-INSTANCE is connected.

NM-USER-DATA-ENABLED

NM-USER-DATA-ENABLED is configured by taking NM-USER-DATA-ENABLED's maximum value of all CAN-COMMUNICATION-CONTROLLERS and FLEXRAY-COMMUNICATION-CONTROLLERS to which the ECU-INSTANCE is connected.

8.1.2.7.3. CAN-NM-CLUSTER

Source model entity: CAN-CLUSTER

Target model entity: CAN-NM-CLUSTER

Target model property	Source model property
NM-BUSLOAD-REDUCTION-ACTIVE	NM-BUS-LOAD-REDUCTION-ACTIVE
NM-NETWORK-TIMEOUT	NM-TIMEOUT-TIME
NM-REPEAT-MESSAGE-TIME	NM-REPEAT-MESSAGE-STATE-TIME
NM-REMOTE-SLEEP-INDICATION-TIME	NM-REMOTE-SLEEP-INDICATION-TIME
NM-WAIT-BUS-SLEEP-TIME	NM-WAIT-BUS-SLEEP-TIME
NM-CHANNEL-SLEEP-MASTER	No source model parameter is available.
NM-IMMEDIATE-NM-CYCLE-TIME	NM-IMMEDIATE-NM-CYCLE-TIME
NM-MSG-CYCLE-TIME	NM-MSG-CYCLE-TIME



Target model property	Source model property
NM-REMOTE-SLEEP-INDICATION-TIME	NM-REMOTE-SLEEP-INDICATION-TIME

The created CAN-NM-CLUSTER is aggregated in NM-CONFIG/NM-CLUSTERS. It references the related CAN-CLUSTER in COMMUNICATION-CLUSTER-REF.

8.1.2.7.3.1. New parameters in the target model

NM-CBV-POSITION, NM-NID-POSITION

NM-CBV-POSITION, NM-NID-POSITION are calculated by taking NM-CBV-POSITION, NM-NID-POSITION's maximum value of any NM-PDU transmitted over the CAN-CLUSTER in the source model, and then dividing the result by eight.

NM-USER-DATA-LENGTH

NM-USER-DATA-LENGTH is calculated by taking NM-USER-DATA-LENGTH's maximum value of any NM-PDU transmitted over the CAN-CLUSTER.

8.1.2.7.4. CAN-NM-NODE

For every CAN-COMMUNICATION-CONTROLLER connected to a CAN-CLUSTER in the source model, one CAN-NM-NODE entity is created. CAN-NM-NODE is aggregated in CAN-NM-CLUSTER/NM-NODES.

8.1.2.7.4.1. New parameters in the target model

NM-RANGE-CONFIG

If the source model COMMUNICATION-CLUSTER to which the CAN-COMMUNICATION-CONTROLLER is connected defines values for both NM-LOWER-CAN-ID and NM-UPPER-CAN-ID, one NM-RANGE-CONFIG element is added to the CAN-NM-NODE. NM-RANGE-CONFIG/LOWER-CAN-ID is set to the value of NM-LOWER-CAN-ID, and NM-RANGE-CONFIG/UPPER-CAN-ID is set to the value of NM-UPPER-CAN-ID.

CONTROLLER-REF

CONTROLLER-REF is configured to reference the target model counterpart of the CAN-COMMUNICATION-CONTROLLER.

NM-NODE-ID

The parameter NM-NODE-ID is derived from the NM-ADDRESS parameter value of COMMUNICATION-CONNECTOR that references the source model counterpart of the CAN-COMMUNICATION-CONTROLLER.

NM-IF-ECU-REF

NM-IF-ECU-REF is configured to reference the NM-ECU element, which has been created for the source model ECU-INSTANCE that aggregates the CAN-COMMUNICATION-CONTROLLER this CAN-NM-NODE has been created for.

**RX-NM-PDU-REFS/RX-NM-PDU-REF**

RX-NM-PDU-REFS/RX-NM-PDU-REF is configured to reference all NM-PDUS the CAN-COMMUNICATION-CONTROLLER is receiving.

TX-NM-PDU-REFS/TX-NM-PDU-REF

TX-NM-PDU-REFS/TX-NM-PDU-REF is configured to reference all NM-PDUS the CAN-COMMUNICATION-CONTROLLER is sending.

NM-MSG-REDUCED-TIME

NM-MSG-REDUCED-TIME is configured by CAN-COMMUNICATION-CONTROLLER/NM-MSG-REDUCED-TIME.

NM-MSG-CYCLE-OFFSET

NM-MSG-CYCLE-OFFSET is configured by CAN-COMMUNICATION-CONTROLLER/NM-MSG-CYCLE-OFFSET.

8.1.2.7.5. CAN-NM-CLUSTER-COUPING

CAN-NM-CLUSTER-COUPING defines parameters which have to be set consistently among two coupled CAN-NM-CLUSTER elements. [Section 8.1.1.7.5, “CAN-NM-CLUSTER-COUPING”](#) describes the peculiarities of the CAN-NM-CLUSTER-COUPING.

For each of the coupled sets of CAN-NM-CLUSTER elements, and for each CAN-NM-CLUSTER that is not coupled to any other CAN-NM-CLUSTER, one element is created, aggregated in CAN-NM-CONFIG/NM-CLUSTER-COUPLINGS.

8.1.2.7.5.1. New parameters in the target model

COUPLED-CLUSTER-REFS/COUPLED-CLUSTER-REF

COUPLED-CLUSTER-REFS/COUPLED-CLUSTER-REF is configured to reference all CAN-NM-CLUSTERS that are coupled.

NM-BUSLOAD-REDUCTION-ENABLED

If the NM-BUS-LOAD-REDUCTION-ENABLED parameters of the CAN-CLUSTER counterparts of the coupled CAN-NM-CLUSTERS yield a consistent value, the NM-BUSLOAD-REDUCTION-ENABLED of the CAN-NM-CLUSTER-COUPING is configured accordingly.

8.1.2.7.6. FLEXRAY-NM-CLUSTER

One FLEXRAY-NM-CLUSTER is created in the target model for each FLEXRAY-CLUSTER in the source model. FLEXRAY-NM-CLUSTER is aggregated in NM-CONFIG/NM-CLUSTERS and references the related FLEXRAY-CLUSTER in COMMUNICATION-CLUSTER-REF.



8.1.2.7.6.1. New parameters in the target model

NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPEAT-MESSAGE-TIME, NM-REPETITION-CYCLE, NM-VOTING-CYCLE, NM-DATA-CYCLE, NM-READY-SLEEP-COUNT

NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPEAT-MESSAGE-TIME, NM-REPETITION-CYCLE, NM-VOTING-CYCLE, NM-DATA-CYCLE, NM-READY-SLEEP-COUNT are defined by taking the respective parameter values of the source model FLEXRAY-CLUSTER.

NM-CONTROL-BIT-VECTOR-ACTIVE

NM-CONTROL-BIT-VECTOR-ACTIVE is defined by taking the maximum NM-CONTROL-BIT-VECTOR-ENABLED parameter value of all source model FLEXRAY-COMMUNICATION-CONTROLLERS that are connected to the FLEXRAY-CLUSTER.

8.1.2.7.7. FLEXRAY-NM-NODE

For every FLEXRAY-COMMUNICATION-CONTROLLER connected to a FLEXRAY-CLUSTER in the source model, one FLEXRAY-NM-NODE entity is created. FLEXRAY-NM-NODE is aggregated in FLEXRAY-NM-CLUSTER/NM-NODES.

8.1.2.7.7.1. New parameters in the target model

CONTROLLER-REF

CONTROLLER-REF is configured to reference the target model counterpart of the FLEXRAY-COMMUNICATION-CONTROLLER.

NM-NODE-ID

The parameter NM-NODE-ID is derived from NM-ADDRESS's maximum value of all COMMUNICATION-CONNECTORS that reference the source model counterpart of the FLEXRAY-COMMUNICATION-CONTROLLER.

NM-IF-ECU-REF

NM-IF-ECU-REF references the target model NM-ECU element that has been created for the source model ECU-INSTANCE to which the source model FLEXRAY-COMMUNICATION-CONTROLLER belongs. A source model FLEXRAY-COMMUNICATION-CONTROLLER belongs to a source model ECU-INSTANCE, if the ECU-INSTANCE aggregates the FLEXRAY-COMMUNICATION-CONTROLLER in COMM-CONTROLLERS.

RX-NM-PDU-REFS/RX-NM-PDU-REF

RX-NM-PDU-REFS/RX-NM-PDU-REF is configured to reference all NM-PDUS the FLEXRAY-COMMUNICATION-CONTROLLER is receiving.

TX-NM-PDU-REFS/TX-NM-PDU-REF

TX-NM-PDU-REFS/TX-NM-PDU-REF is configured to reference all NM-PDUS the FLEXRAY-COMMUNICATION-CONTROLLER is sending.



8.1.2.7.8. FLEXRAY-NM-CLUSTER-COUPING

FLEXRAY-NM-CLUSTER-COUPING defines parameters which have to be set consistently among two FLEXRAY-NM-CLUSTER elements which are coupled. [Section 8.1.1.7.5, “CAN-NM-CLUSTER-COUPING”](#) describes the peculiarities of the CAN cluster analogy CAN-NM-CLUSTER-COUPING.

For each of the coupled sets of FLEXRAY-NM-CLUSTER elements, and for each FLEXRAY-NM-CLUSTER not coupled to any other FLEXRAY-NM-CLUSTER, one FLEXRAY-NM-CLUSTER-COUPING element is created, aggregated in FLEXRAY-NM-CONFIG/NM-CLUSTER-COPLINGS.

8.1.2.8. Software Component Description

8.1.2.8.1. COMPOSITION-SW-COMPONENT-TYPE

The COMPOSITION-SW-COMPONENT-TYPE is upgraded as described in [Section 8.1.1.8.1, “COMPOSITION-SW-COMPONENT-TYPE”](#).

8.1.2.8.2. APPLICATION-SW-COMPONENT-TYPE

Source model entities: APPLICATION-SOFTWARE-COMPONENT-TYPE, SENSOR-ACTUATOR-SOFT-WARE-COMPONENT-TYPE

Target model entity: APPLICATION-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP

8.1.2.8.2.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.2.8.3. SERVICE-SW-COMPONENT-TYPE

Source model entity: SERVICE-COMPONENT-TYPE



Target model entity: SERVICE-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
No upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MODULE-DESCRIPTION-REF

8.1.2.8.3.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.2.8.4. COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE

Source model entity: COMPLEX-DEVICE-DRIVER-COMPONENT-TYPE

Target model entity: COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
No upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MODULE-DESCRIPTION-REF
No upgrade is currently implemented.	HARDWARE-ELEMENT-REFS/HARDWARE-ELEMENT-REF

8.1.2.8.4.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.



8.1.2.8.5. ECU-ABSTRACTION-SW-COMPONENT-TYPE

Source model entity: ECU-ABSTRACTION-COMPONENT-TYPE

Target model entity: ECU-ABSTRACTION-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
No upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MODULE-DESCRIPTION-REF
No upgrade is currently implemented.	HARDWARE-ELEMENT-REFS/HARDWARE-ELEMENT-REF

8.1.2.8.5.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.2.8.6. PARAMETER-SW-COMPONENT-TYPE

Source model entity: CALPRM-COMPONENT-TYPE

Target model entity: PARAMETER-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
CONSTANT-MAPPING-REFS/CONSTANT-MAPPING-REF	No source model parameter is available.
DATA-TYPE-MAPPING-REFS/DATA-TYPE-MAPPING-REF	Refer to Section 8.1.1.8.37.4, “Creation of target model DATA-TYPE-MAPPING-REFS/DATA-TYPE-MAPPING-REF” .
INSTANTIATION-DATA-DEF-PROPS/INSTANTIATION-DATA-DEF-PROPS	No source model parameter is available.



8.1.2.8.6.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.2.8.7. R-PORT-PROTOTYPE

R-PORT-PROTOTYPE is upgraded as described in [Section 8.1.1.8.7, “R-PORT-PROTOTYPE”](#).

8.1.2.8.8. P-PORT-PROTOTYPE

P-PORT-PROTOTYPE is upgraded as described in [Section 8.1.1.8.8, “P-PORT-PROTOTYPE”](#).

8.1.2.8.9. CLIENT-SERVER-INTERFACE

CLIENT-SERVER-INTERFACE is upgraded as described in [Section 8.1.1.8.9, “CLIENT-SERVER-INTERFACE”](#).

8.1.2.8.10. DELEGATION-SW-CONNECTOR

DELEGATION-SW-CONNECTOR is upgraded as described in [Section 8.1.1.8.11, “DELEGATION-SW-CONNECTOR”](#).

8.1.2.8.11. ASSEMBLY-SW-CONNECTOR

ASSEMBLY-SW-CONNECTOR is upgraded as described in [Section 8.1.1.8.10, “ASSEMBLY-SW-CONNECTOR”](#).

8.1.2.8.12. SERVICE-CONNECTOR-PROTOTYPE

AUTOSAR 3.2.x SERVICE-CONNECTOR-PROTOTYPE elements are not upgraded because no counterpart element exists in AUTOSAR 4.0. If such SENDER-RECEIVER-INTERFACE elements are encountered, a warning is issued.

8.1.2.8.13. PARAMETER-INTERFACE

PARAMETER-INTERFACE is upgraded as described in [Section 8.1.1.8.13, “PARAMETER-INTERFACE”](#).



8.1.2.8.14. MODE-SWITCH-INTERFACE

MODE-SWITCH-INTERFACE is upgraded as described in [Section 8.1.1.8.14, “MODE-SWITCH-INTERFACE”](#).

8.1.2.8.15. MODE-GROUP

Source model entity: MODE-GROUP

Target model entity: MODE-GROUP

Target model property	Source model property
TYPE-TREF	TYPE-TREF
SW-CALIBRATION-ACCESS	No source model parameter is available.

8.1.2.8.16. SW-COMPONENT-PROTOTYPE

SW-COMPONENT-PROTOTYPE is upgraded as described in [Section 8.1.1.8.16, “SW-COMPONENT-PROTOTYPE”](#).

8.1.2.8.17. VARIABLE-DATA-PROTOTYPE

VARIABLE-DATA-PROTOTYPE is upgraded as described in [Section 8.1.1.8.17, “VARIABLE-DATA-PROTOTYPE”](#).

8.1.2.8.18. ARGUMENT-DATA-PROTOTYPE

ARGUMENT-DATA-PROTOTYPE is upgraded as described in [Section 8.1.1.8.18, “ARGUMENT-DATA-PROTOTYPE”](#).

8.1.2.8.19. PARAMETER-DATA-PROTOTYPE

PARAMETER-DATA-PROTOTYPE is upgraded as described in [Section 8.1.1.8.19, “PARAMETER-DATA-PROTOTYPE”](#).

8.1.2.8.20. CLIENT-SERVER-OPERATION

CLIENT-SERVER-OPERATION is upgraded as described in [Section 8.1.1.8.20, “CLIENT-SERVER-OPERATION”](#).



8.1.2.8.21. APPLICATION-ERROR

APPLICATION-ERROR is upgraded as described in [Section 8.1.1.8.21, “APPLICATION-ERROR”](#).

8.1.2.8.22. NONQUEUED-RECEIVER-COM-SPEC

Source model entity: UNQUEUED-RECEIVER-COM-SPEC

Target model entity: NONQUEUED-RECEIVER-COM-SPEC

Target model property	Source model property
ALIVE-TIMEOUT	ALIVE-TIMEOUT
No target model parameter is available.	HANDLE-INVALID
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF
No target model parameter is available.	RESYNC-TIME
DATA-ELEMENT-REF	DATA-ELEMENT-REF
FILTER	FILTER
HANDLE-NEVER-RECEIVED	HANDLE-NEVER-RECEIVED if the source model parameter value is available, else false.
ENABLE-UPDATE	No source model parameter is available, the target model parameter is set to false.
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.
HANDLE-TIMEOUT-TYPE	No source model parameter is available, the target model parameter is set to NONE.
NETWORK-REPRESENTATION	No source model parameter is available.
USES-END-TO-END-PROTECTION	USES-END-TO-END-PROTECTION
EXTERNAL-REPLACEMENT-REF	No source model parameter is available.
MAX-DELTA-COUNTER-INIT	No source model parameter is available.

8.1.2.8.22.1. Processing of source model parameter HANDLE-INVALID

If the parameter HANDLE-INVALID is provided in the source model, a INVALIDATION-POLICY element is created in the related target model SENDER-RECEIVER-INTERFACE. The INVALIDATION-POLICY parameter HANDLE-INVALID is configured according to the source model UNQUEUED-RECEIVER-COM-SPEC parameter HANDLE-INVALID if all UNQUEUED-RECEIVER-COM-SPEC elements belonging to the source model SENDER-RECEIVER-INTERFACE and DATA-ELEMENT are providing the same HANDLE-INVALID value.



8.1.2.8.23. QUEUED-RECEIVER-COM-SPEC

Source model entity: QUEUED-RECEIVER-COM-SPEC

Target model entity: QUEUED-RECEIVER-COM-SPEC

Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
No target model parameter is available.	FILTER
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.
HANDLE-OUT-OF-RANGE-STATUS	No source model parameter is available.
NETWORK-REPRESENTATION	No source model parameter is available.
QUEUE-LENGTH	QUEUE-LENGTH
USES-END-TO-END-PROTECTION	No source model parameter is available, the target model parameter is set to false.
MAX-DELTA-COUNTER-INIT	No source model parameter is available.

8.1.2.8.24. CLIENT-COM-SPEC

CLIENT-COM-SPEC is upgraded as described in [Section 8.1.1.8.24, “CLIENT-COM-SPEC”](#).

8.1.2.8.25. PARAMETER-REQUIRE-COM-SPEC

PARAMETER-REQUIRE-COM-SPEC is upgraded as described in [Section 8.1.1.8.25, “PARAMETER-REQUIRE-COM-SPEC”](#).

8.1.2.8.26. QUEUED-SENDER-COM-SPEC

Source model entity: QUEUED-SENDER-COM-SPEC

Target model entity: QUEUED-SENDER-COM-SPEC

Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
TRANSMISSION-ACKNOWLEDGE	TRANSMISSION-ACKNOWLEDGE
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.



Target model property	Source model property
NETWORK-REPRESENTATION	No source model parameter is available.
USES-END-TO-END-PROTECTION	USES-END-TO-END-PROTECTION

8.1.2.8.27. NONQUEUED-SENDER-COM-SPEC

Source model entity: UNQUEUED-SENDER-COM-SPEC

Target model entity: NONQUEUED-SENDER-COM-SPEC

Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
TRANSMISSION-ACKNOWLEDGE	TRANSMISSION-ACKNOWLEDGE
No target model parameter is available.	CAN-INVALIDATE
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.
NETWORK-REPRESENTATION	No source model parameter is available.
USES-END-TO-END-PROTECTION	USES-END-TO-END-PROTECTION

8.1.2.8.28. SERVER-COM-SPEC

SERVER-COM-SPEC is upgraded as described in [Section 8.1.1.8.28, “SERVER-COM-SPEC”](#).

8.1.2.8.29. MODE-SWITCH-SENDER-COM-SPEC

Source model entity: MODE-SWITCH-COM-SPEC

Target model entity: MODE-SWITCH-SENDER-COM-SPEC

Target model property	Source model property
MODE-GROUP-REF	MODE-GROUP-REF
MODE-SWITCHED-ACK	MODE-SWITCHED-ACK
QUEUE-LENGTH	QUEUE-LENGTH
ENHANCED-MODE-API	No source model parameter is available.



8.1.2.8.30. PARAMETER-PROVIDE-COM-SPEC

PARAMETER-PROVIDE-COM-SPEC is upgraded as described in [Section 8.1.1.8.30, “PARAMETER-PROVIDE-COM-SPEC”](#).

8.1.2.8.31. TRANSMISSION-ACKNOWLEDGE

TRANSMISSION-ACKNOWLEDGE is upgraded as described in [Section 8.1.1.8.31, “TRANSMISSION-ACKNOWLEDGE”](#).

8.1.2.8.32. DATA-FILTER

Source model entities: SIGNAL-PORT/DATA-FILTER and TRANSMISSION-MODE-CONDITION/DATA-FILTER

Target model entity: DATA-FILTER

Target model property	Source model property
MASK	MASK
MAX	MAX
MIN	MIN
OFFSET	OFFSET
PERIOD	PERIOD
X	X

8.1.2.8.32.1. Configuration of target model parameter DATA-FILTER-TYPE

DATA-FILTER-TYPE is configured depending on the source model element below SIGNAL-PORT/DATA-FILTER and TRANSMISSION-MODE-CONDITION/DATA-FILTER. The following values are converted:

- ▶ ALWAYS
- ▶ MASKED-NEW-DIFFERS-MASKED-OLD
- ▶ MASKED-NEW-DIFFERS-X
- ▶ MASKED-NEW-EQUALS-X
- ▶ NEVER
- ▶ NEW-IS-OUTSIDE



- ▶ NEW-IS-WITHIN
- ▶ ONE-EVERY-N

8.1.2.8.33. VARIABLE-ACCESS

VARIABLE-ACCESS is upgraded as described in [Section 8.1.1.8.33, “VARIABLE-ACCESS”](#).

8.1.2.8.34. PARAMETER-ACCESS

PARAMETER-ACCESS is upgraded as described in [Section 8.1.1.8.34, “PARAMETER-ACCESS”](#).

8.1.2.8.35. RUNNABLE-ENTITY

Source model entity: RUNNABLE-ENTITY

Target model entity: RUNNABLE-ENTITY

Target model property	Source model property
ARGUMENTS/RUNNABLE-ENTITY-ARGUMENT	No source model parameter is available.
Currently no upgrade is available.	BSW-ENTITY-REF
PARAMETER-ACCESSS/PARAMETER-ACCESS/ACCESSED-PARAMETER	CALPRM-ACCESSS/CALPRM-ACCESS
CAN-BE-INVOKED-CONCURRENTLY	CAN-BE-INVOKED-CONCURRENTLY
CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF	CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF
DATA-READ-ACCESSS/VARIABLE-ACCESS	DATA-READ-ACCESSS/DATA-READ-ACCESS
DATA-RECEIVE-POINT-BY-ARGUMENTS/VARIABLE-ACCESS	DATA-RECEIVE-POINTS/DATA-RECEIVE-POINT
DATA-SEND-POINTS/VARIABLE-ACCESS	DATA-SEND-POINTS/DATA-SEND-POINT
DATA-WRITE-ACCESSS/VARIABLE-ACCESS	DATA-WRITE-ACCESSS/DATA-WRITE-ACCESS
MINIMUM-START-INTERVAL	MINIMUM-START-INTERVAL
MODE-SWITCH-POINTS/MODE-SWITCH-POINT	MODE-SWITCH-POINTS/MODE-SWITCH-POINT
PARAMETER-ACCESSS/PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	PER-INSTANCE-CALPRM-ACCESS-REFS/PER-INSTANCE-CALPRM-ACCESS-REF
RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF	RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF



Target model property	Source model property
SERVER-CALL-POINTS/ASYNCHRONOUS-SERVER-CALL-POINT	SERVER-CALL-POINTS/ASYNCHRONOUS-SERVER-CALL-POINT
SERVER-CALL-POINTS/SYNCHRONOUS-SERVER-CALL-POINT	SERVER-CALL-POINTS/SYNCHRONOUS-SERVER-CALL-POINT
PARAMETER-ACCESSSS/PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	SHARED-CALPRM-ACCESS-REFS/SHARED-CALPRM-ACCESS-REF
SYMBOL	SYMBOL
WAIT-POINTS/WAIT-POINT	WAIT-POINTS/WAIT-POINT
WRITTEN-LOCAL-VARIABLES/VARIABLE-ACCESS/ACCESSED-VARIABLE/LOCAL-VARIABLE-REF	WRITTEN-VARIABLE-REFS/WRITTEN-VARIABLE-REF
READ-LOCAL-VARIABLES/VARIABLE-ACCESS/ACCESSED-VARIABLE/LOCAL-VARIABLE-REF	READ-VARIABLE-REFS/READ-VARIABLE-REF

8.1.2.8.35.1. Migration of source model DATA-RECEIVE-POINT elements

Note that all source model DATA-RECEIVE-POINT elements are migrated to DATA-RECEIVE-POINT-BY-ARGUMENTS/VARIABLE-ACCESS elements in the target model, i.e. there is no conversion to the target model entities DATA-RECEIVE-POINT-BY-VALUES/VARIABLE-ACCESS.

8.1.2.8.35.2. Creation of target model ASYNCHRONOUS-SERVER-CALL-RESULT-POINT elements

Note that for every source model ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT entity which either references a RUNNABLE-ENTITY via START-ON-EVENT-REF or references a SERVER-CALL-POINT of the RUNNABLE-ENTITY via EVENT-SOURCE-REF, one ASYNCHRONOUS-SERVER-CALL-RESULT-POINTS/ASYNCHRONOUS-SERVER-CALL-RESULT-POINT is created in the target model.

8.1.2.8.36. WAIT-POINT

WAIT-POINT is upgraded as described in [Section 8.1.1.8.36, “WAIT-POINT”](#).

8.1.2.8.37. SWC-INTERNAL-BEHAVIOR

SWC-INTERNAL-BEHAVIOR is upgraded as described in [Section 8.1.1.8.37, “SWC-INTERNAL-BEHAVIOR”](#).



8.1.2.8.38. PORT-API-OPTION

PORT-API-OPTION is upgraded as described in [Section 8.1.1.8.38, “PORT-API-OPTION”](#).

8.1.2.8.39. DATA-SEND-COMPLETED-EVENT

DATA-SEND-COMPLETED-EVENT is upgraded as described in [Section 8.1.1.8.39, “DATA-SEND-COMPLETED-EVENT”](#).

8.1.2.8.40. DATA-RECEIVED-EVENT

DATA-RECEIVED-EVENT is upgraded as described in [Section 8.1.1.8.40, “DATA-RECEIVED-EVENT”](#).

8.1.2.8.41. DATA-RECEIVE-ERROR-EVENT

DATA-RECEIVE-ERROR-EVENT is upgraded as described in [Section 8.1.1.8.41, “DATA-RECEIVE-ERROR-EVENT”](#).

8.1.2.8.42. ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT

ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT is upgraded as described in [Section 8.1.1.8.42, “ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT”](#).

8.1.2.8.43. TIMING-EVENT

TIMING-EVENT is upgraded as described in [Section 8.1.1.8.43, “TIMING-EVENT”](#).

8.1.2.8.44. OPERATION-INVOKED-EVENT

OPERATION-INVOKED-EVENT is upgraded as described in [Section 8.1.1.8.44, “OPERATION-INVOKED-EVENT”](#).

8.1.2.8.45. SWC-MODE-SWITCH-EVENT

SWC-MODE-SWITCH-EVENT is upgraded as described in [Section 8.1.1.8.45, “SWC-MODE-SWITCH-EVENT”](#).



8.1.2.8.46. MODE-SWITCHED-ACK-EVENT

MODE-SWITCHED-ACK-EVENT is upgraded as described in [Section 8.1.1.8.46, “MODE-SWITCHED-ACK-EVENT”](#).

8.1.2.8.47. ASYNCHRONOUS-SERVER-CALL-POINT

ASYNCHRONOUS-SERVER-CALL-POINT is upgraded as described in [Section 8.1.1.8.47, “ASYNCHRONOUS-SERVER-CALL-POINT”](#).

8.1.2.8.48. SYNCHRONOUS-SERVER-CALL-POINT

SYNCHRONOUS-SERVER-CALL-POINT is upgraded as described in [Section 8.1.1.8.48, “SYNCHRONOUS-SERVER-CALL-POINT”](#).

8.1.2.8.49. NUMERICAL-VALUE-SPECIFICATION

NUMERICAL-VALUE-SPECIFICATION is upgraded as described in [Section 8.1.1.8.49, “NUMERICAL-VALUE-SPECIFICATION”](#).

8.1.2.8.50. TEXT-VALUE-SPECIFICATION

TEXT-VALUE-SPECIFICATION is upgraded as described in [Section 8.1.1.8.50, “TEXT-VALUE-SPECIFICATION”](#).

8.1.2.8.51. ARRAY-VALUE-SPECIFICATION

ARRAY-VALUE-SPECIFICATION is upgraded as described in [Section 8.1.1.8.51, “ARRAY-VALUE-SPECIFICATION”](#).

8.1.2.8.52. RECORD-VALUE-SPECIFICATION

RECORD-VALUE-SPECIFICATION is upgraded as described in [Section 8.1.1.8.52, “RECORD-VALUE-SPECIFICATION”](#).

8.1.2.8.53. CONSTANT-SPECIFICATION

CONSTANT-SPECIFICATION is upgraded as described in [Section 8.1.1.8.53, “CONSTANT-SPECIFICATION”](#).



8.1.2.8.54. CONSTANT-REFERENCE

CONSTANT-REFERENCE is upgraded as described in [Section 8.1.1.8.54, “CONSTANT-REFERENCE”](#).

8.1.2.8.55. UNIT

Source model entity: UNIT

Target model entity: UNIT

Target model property	Source model property
FACTOR-SI-TO-UNIT	FACTOR-SI-TO-UNIT
OFFSET-SI-TO-UNIT	OFFSET-SI-TO-UNIT
PHYSICAL-DIMENSION-REF	PHYSICAL-DIMENSION-REF

8.1.2.8.55.1. No upgrade of source model DISPLAY-NAME

Due to internal data model restrictions, the source model parameter DISPLAY-NAME is not upgraded to the target model.

8.1.2.8.56. UNIT-GROUP

Due to internal data model restrictions, there is no upgrade path for UNIT-GROUP elements.

8.1.2.8.57. PHYSICAL-DIMENSION

Source model entity: PHYSICAL-DIMENSION

Target model entity: PHYSICAL-DIMENSION

Target model property	Source model property
CURRENT-EXP	CURRENT-EXP
LENGTH-EXP	LENGTH-EXP
LUMINOUS-INTENSITY-EXP	LUMINOUS-INTENSITY-EXP
MASS-EXP	MASS-EXP
MOLAR-AMOUNT-EXP	MOLAR-AMOUNT-EXP
TEMPERATURE-EXP	TEMPERATURE-EXP
TIME-EXP	TIME-EXP



8.1.2.8.58. SW-BASE-TYPE

Source model entity: SW-BASE-TYPE

Target model entity: SW-BASE-TYPE

Target model property	Source model property
MEM-ALIGNMENT	MEM-ALIGNMENT
BASE-TYPE-SIZE	BASE-TYPE-SIZE
MAX-BASE-TYPE-SIZE	MAX-BASE-TYPE-SIZE

8.1.2.8.58.1. Upgrade of source model BASE-TYPE-ENCODING.

If the source model BASE-TYPE-ENCODING contains one of the values

- ▶ 1C
- ▶ 2C
- ▶ BCD-P
- ▶ BCD-UP
- ▶ DSP-FRACTIONAL
- ▶ SM
- ▶ IEEE754
- ▶ ISO-8859-1
- ▶ ISO-8859-2
- ▶ WINDOWS-1252
- ▶ UTF-8
- ▶ UCS-2
- ▶ NONE

the value is directly copied into the target model parameter BASE-TYPE-ENCODING. For any other value in the source model, NONE will be written into the target model parameter.

8.1.2.8.58.2. Upgrade of source model CATEGORY

If the source model CATEGORY value is either SW_FIXED_LENGTH or FIXED_LENGTH, the target model CATEGORY parameter will contain the value FIXED_LENGTH. If the source model parameter value contains SW_VARIABLE_LENGTH or VARIABLE_LENGTH, the target model CATEGORY parameter value will contain VARIABLE_LENGTH.



8.1.2.8.58.3. No upgrade of source model BASE-TYPE-REF

Due to internal data model restrictions, the source model parameter `BASE-TYPE-REF` is not upgraded into the target model.

8.1.2.8.58.4. No upgrade of source model BYTE-ORDER

The source model parameter `BYTE-ORDER` is not upgraded into the target model because `BYTE-ORDER` is not a property of a (base) data type.

8.1.2.8.59. SW-DATA-DEF-PROPS

Source model entity: `SW-DATA-DEF-PROPS`

Target model entity: `SW-DATA-DEF-PROPS`

Target model property	Source model property
<code>SW-DATA-DEF-PROPS-CONDITIONAL/BASE-TYPE-REF</code>	<code>BASE-TYPE-REF</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/COMPU-METHOD-REF</code>	<code>COMPU-METHOD-REF</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/INVALID-VALUE</code>	<code>INVALID-VALUE</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/SW-CALIBRATION-ACCESS</code>	<code>SW-CALIBRATION-ACCESS</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/SW-IMPL-POLICY</code>	<code>SW-IMPL-POLICY</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/UNIT-REF</code>	<code>UNIT-REF</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/INVALID-VALUE</code>	<code>INVALID-VALUE</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/SW-VALUE-BLOCK-SIZE</code>	<code>SW-VALUE-BLOCK-SIZE/V</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/DISPLAY-FORMAT</code>	<code>DISPLAY-FORMAT</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/DATA-CONSTR-REF</code>	<code>DATA-CONSTR-REF</code>
<code>SW-DATA-DEF-PROPS-CONDITIONAL/SW-POINTER-TARGET-PROPS</code>	<code>SW-POINTER</code>



8.1.2.8.59.1. No upgrade of source model parameters SW-CALPRM-AXIS-SET, SW-CLASS-ATTR-IMPL-REF, SW-CLASS-REF, SW-CODE-SYNTAX-REF, SW-DATA-DEPENDENCY , SW-HOST-VARIABLE, SW-POINTER, SW-RECORD-LAYOUT-REF, SW-TEXT-PROPS, SW-VARIABLE-ACCESS-IMPL-POLICY

The source model parameters SW-CALPRM-AXIS-SET, SW-CLASS-ATTR-IMPL-REF, SW-CLASS-REF, SW-CODE-SYNTAX-REF, SW-DATA-DEPENDENCY , SW-HOST-VARIABLE, SW-POINTER, SW-RECORD-LAYOUT-REF, SW-TEXT-PROPS, SW-VARIABLE-ACCESS-IMPL-POLICY are not upgraded to the target model.

8.1.2.8.59.2. No configuration of target model parameters SW-ALIGNMENT, SW-COMPARISON-VARIABLES, SW-DATA-DEPENDENCY, SW-HOST-VARIABLE, SW-INTENDED-RESOLUTION, SW-INTERPOLATION-METHOD, SW-IS-VIRTUAL, SW-REFRESH-TIMING, SW-TEXT-PROPS, VALUE-AXIS-DATA-TYPE-REF

The target model parameters SW-ALIGNMENT, SW-COMPARISON-VARIABLES, SW-DATA-DEPENDENCY, SW-HOST-VARIABLE, SW-INTENDED-RESOLUTION, SW-INTERPOLATION-METHOD, SW-IS-VIRTUAL, SW-REFRESH-TIMING, SW-TEXT-PROPS, VALUE-AXIS-DATA-TYPE-REF are not configured during the model upgrade because there is no source model counterpart.

8.1.2.8.59.3. Configuration of SW-DATA-DEF-PROPS-CONDITIONAL/SW-IMPL-POLICY

If the source model SW-DATA-DEF-PROPS element is aggregated within a DATA-ELEMENT-PROTOTYPE which has its IS-QUEUED value set to true, the upgraded SW-DATA-DEF-PROPS-CONDITIONAL/SW-IMPL-POLICY contains the value QUEUED, else it contains the SW-IMPL-POLICY value of the source model SW-DATA-DEF-PROPS counterpart. If the source model SW-DATA-DEF-PROPS value is MESSAGE, the parameter is not configured at all in the target model since it does not support this value.

8.1.2.8.59.4. Creation of SW-DATA-DEF-PROPS elements in the target model

For every target model ARGUMENT-DATA-PROTOTYPE, PARAMETER-DATA-PROTOTYPE, or VARIABLE-DATA-PROTOTYPE which has a source model counterpart which has its IS-QUEUED parameter value set to true but does not aggregate an SW-DATA-DEF-PROPS element, a dedicated SW-DATA-DEF-PROPS is created in the target model which has its SW-IMPL-POLICY value set to QUEUED.

8.1.2.8.60. DATA-CONSTR

Source model entity: DATA-CONSTR

Target model entity: DATA-CONSTR



Target model property	Source model property
DATA-CONSTR-RULES / DATA-CONSTR-RULE	DATA-CONSTR-RULES / DATA-CONSTR-RULE
SHORT-NAME-PATTERN	No source model parameter is available.

8.1.2.8.61. DATA-CONSTR-RULE

Source model entity: DATA-CONSTR-RULE

Target model entity: DATA-CONSTR-RULE

Target model property	Source model property
INTERNAL-CONSTRS	INTERNAL-CONSTRS
PHYS-CONSTRS	PHYS-CONSTRS

8.1.2.8.62. PHYS-CONSTRS

Source model entity: PHYS-CONSTRS

Target model entity: PHYS-CONSTRS

Target model property	Source model property
LOWER-LIMIT	LOWER-LIMIT
UPPER-LIMIT	UPPER-LIMIT
UNIT-REF	UNIT-REF

8.1.2.8.62.1. No configuration of target model parameters MAX-DIFF, MAX-GRADIENT, MONOTONY, SCALE-CONSTRS

The target model parameters MAX-DIFF, MAX-GRADIENT, MONOTONY, SCALE-CONSTRS are not configured during the model upgrade because there is no source model counterpart.

8.1.2.8.62.2. Configuration of target model parameter UNIT-REF

If any of the source model PHYS-CONSTRS has no valid UNIT-REF defined, a dedicated UNIT is created in the target model, called NoDimensionUnit. A target model PHYS-CONSTRS is referencing this UNIT via UNIT-REF if its source model counterpart does not provide UNIT-REF.



8.1.2.8.63. INTERNAL-CONSTRS

Source model entity: INTERNAL-CONSTRS

Target model entity: INTERNAL-CONSTRS

Target model property	Source model property
LOWER-LIMIT	LOWER-LIMIT
UPPER-LIMIT	UPPER-LIMIT

8.1.2.8.63.1. No configuration of target model parameters MAX-DIFF, MAX-GRADIENT, MONOTONY, SCALE-CONSTRS

The target model parameters MAX-DIFF, MAX-GRADIENT, MONOTONY, SCALE-CONSTRS are not configured during the model upgrade because there is no source model counterpart.

8.1.2.8.64. IMPLEMENTATION-DATA-TYPE

IMPLEMENTATION-DATA-TYPE is upgraded as described in [Section 8.1.1.8.59, “IMPLEMENTATION-DATA-TYPE”](#).

8.1.2.8.65. COMPU-METHOD

COMPU-METHOD is upgraded as described in [Section 8.1.1.8.60, “COMPU-METHOD”](#).

8.1.2.8.66. COMPU-SCALE

Source model entity: COMPU-SCALE

Target model entity: COMPU-SCALE

Target model property	Source model property
LOWER-LIMIT	LOWER-LIMIT
UPPER-LIMIT	UPPER-LIMIT
COMPU-CONST	COMPU-CONST
COMPU-INVVERSE-VALUE	No source model parameter is available.
COMPU-RATIONAL-COEFFS/COMPU-NUMERATOR	COMPU-RATIONAL-COEFFS/COMPU-NUMERATOR
COMPU-RATIONAL-COEFFS/COMPU-DENOMINATOR	COMPU-RATIONAL-COEFFS/COMPU-DENOMINATOR



Target model property	Source model property
MASK	No source model parameter is available.
SHORT-LABEL	SHORT-LABEL
SYMBOL	SYMBOL

8.1.2.8.67. EXCLUSIVE-AREA

EXCLUSIVE-AREA is upgraded as described in [Section 8.1.1.8.62, “EXCLUSIVE-AREA”](#).

8.1.2.8.68. COM-MGR-USER-NEEDS

COM-MGR-USER-NEEDS is upgraded as described in [Section 8.1.1.8.63, “COM-MGR-USER-NEEDS”](#).

8.1.2.8.69. DIAGNOSTIC-COMMUNICATION-MANAGER-NEEDS

DIAGNOSTIC-COMMUNICATION-MANAGER-NEEDS is upgraded as described in [Section 8.1.1.8.64, “DIAGNOSTIC-COMMUNICATION-MANAGER-NEEDS”](#).

8.1.2.8.70. DIAGNOSTIC-EVENT-NEEDS

DIAGNOSTIC-EVENT-NEEDS is upgraded as described in [Section 8.1.1.8.65, “DIAGNOSTIC-EVENT-NEEDS”](#).

8.1.2.8.71. ECU-STATE-MGR-USER-NEEDS

ECU-STATE-MGR-USER-NEEDS is upgraded as described in [Section 8.1.1.8.66, “ECU-STATE-MGR-USER-NEEDS”](#).

8.1.2.8.72. FUNCTION-INHIBITION-NEEDS

FUNCTION-INHIBITION-NEEDS is upgraded as described in [Section 8.1.1.8.67, “FUNCTION-INHIBITION-NEEDS”](#).

8.1.2.8.73. NV-BLOCK-NEEDS

NV-BLOCK-NEEDS is upgraded as described in [Section 8.1.1.8.68, “NV-BLOCK-NEEDS”](#).



8.1.2.8.74. OBD-CONTROL-SERVICE-NEEDS

OBD-CONTROL-SERVICE-NEEDS is upgraded as described in [Section 8.1.1.8.69, “OBD-CONTROL-SERVICE-NEEDS”](#).

8.1.2.8.75. OBD-PID-SERVICE-NEEDS

OBD-PID-SERVICE-NEEDS is upgraded as described in [Section 8.1.1.8.71, “OBD-PID-SERVICE-NEEDS”](#).

8.1.2.8.76. OBD-RATIO-SERVICE-NEEDS

OBD-RATIO-SERVICE-NEEDS is upgraded as described in [Section 8.1.1.8.72, “OBD-RATIO-SERVICE-NEEDS”](#).

8.1.2.8.77. SUPERVISED-ENTITY-NEEDS

SUPERVISED-ENTITY-NEEDS is upgraded as described in [Section 8.1.1.8.73, “SUPERVISED-ENTITY-NEEDS”](#).

8.1.2.8.78. PER-INSTANCE-MEMORY

Source model entity: PER-INSTANCE-MEMORY

Target model entity: PER-INSTANCE-MEMORY

Target model property	Source model property
TYPE	TYPE
TYPE-DEFINITION	TYPE-DEFINITION

8.1.2.8.78.1. Configuration of target model parameter INIT-VALUE

The target model parameter INIT-VALUE is not configured during the model upgrade because there is no source model counterpart.

8.1.2.8.78.2. Creation of target model parameter SW-DATA-DEF-PROPS

For each PER-INSTANCE-MEMORY element created in the target model, one SW-DATA-DEF-PROPS sub element is set up.



8.1.2.8.79. PER-INSTANCE-MEMORY-SIZE

PER-INSTANCE-MEMORY-SIZE is upgraded as described in [Section 8.1.1.8.75, “PER-INSTANCE-MEMORY-SIZE”](#).

8.1.2.8.80. SWC-IMPLEMENTATION

SWC-IMPLEMENTATION is upgraded as described in [Section 8.1.1.8.76, “SWC-IMPLEMENTATION”](#), except for the SHORT-NAME of the created RESOURCE-CONSUMPTION element, see [Section 8.1.3.5.12.2, “Creation of target model RESOURCE-CONSUMPTION”](#).

8.1.2.8.81. CODE

CODE is upgraded as described in [Section 8.1.1.8.77, “CODE”](#).

8.1.2.8.82. MODE-DECLARATION

MODE-DECLARATION is upgraded as described in [Section 8.1.1.8.78, “MODE-DECLARATION”](#).

8.1.2.8.83. MODE-DECLARATION-GROUP

MODE-DECLARATION-GROUP is upgraded as described in [Section 8.1.1.8.79, “MODE-DECLARATION-GROUP”](#).

8.1.2.8.84. DISABLED-MODE-IREF

DISABLED-MODE-IREF is upgraded as described in [Section 8.1.1.8.80, “DISABLED-MODE-IREF”](#).

8.1.2.8.85. MODE-SWITCHED-ACK

MODE-SWITCHED-ACK is upgraded as described in [Section 8.1.1.8.81, “MODE-SWITCHED-ACK”](#).

8.1.2.8.86. MODE-ACCESS-POINT

MODE-ACCESS-POINT is upgraded as described in [Section 8.1.1.8.82, “MODE-ACCESS-POINT”](#).



8.1.2.8.87. MODE-SWITCH-POINT

MODE-SWITCH-POINT is upgraded as described in [Section 8.1.1.8.83, “MODE-SWITCH-POINT”](#).

8.1.2.8.88. SW-ADDR-METHOD

SW-ADDR-METHOD is upgraded as described in [Section 8.1.1.8.84, “SW-ADDR-METHOD”](#).

8.1.2.8.89. MEMORY-SECTION

MEMORY-SECTION elements are not required by the RTE and are therefore not upgraded into the target model.

8.1.2.8.90. SYSTEM-MAPPING

Source model entity: MAPPING

Target model entity: SYSTEM-MAPPING

Target model property	Source model property
DATA-MAPPINGS/CLIENT-SERVER-TO-SIGNAL-NAL-GROUP-MAPPING	DATA-MAPPINGS/CLIENT-SERVER-TO-SIGNAL-NAL-GROUP-MAPPING
DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-NAL-GROUP-MAPPING	DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-NAL-GROUP-MAPPING
DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-NAL-MAPPING	DATA-MAPPINGS/SENDER-RECEIVER-TO-SIGNAL-NAL-MAPPING
PNC-MAPPINGS/PNC-MAPPING	PNC-MAPPINGS/PNC-MAPPING
SW-MAPPINGS/SWC-TO-ECU-MAPPING	SW-MAPPINGS/SWC-TO-ECU-MAPPING
SW-IMPL-MAPPINGS/SWC-TO-IMPL-MAPPING	SW-IMPL-MAPPINGS/SWC-TO-IMPL-MAPPING

8.1.2.8.90.1. No upgrade of source model parameters ECU-RESOURCE-MAPPINGS , MAPPING-CONSTRAINTS , RESOURCE-ESTIMATIONS and SIGNAL-PATH-CONSTRAINTS

Due to internal data model restrictions, the source model parameters ECU-RESOURCE-MAPPINGS , MAPPING-CONSTRAINTS , RESOURCE-ESTIMATIONS and SIGNAL-PATH-CONSTRAINTS are not upgraded to their target model counterparts.

8.1.2.8.91. SWC-TO-ECU-MAPPING

SWC-TO-ECU-MAPPING is upgraded as described in [Section 8.1.1.8.87, “SWC-TO-ECU-MAPPING”](#).



8.1.2.8.92. SWC-TO-IMPL-MAPPING

SWC-TO-IMPL-MAPPING is upgraded as described in [Section 8.1.1.8.88, “SWC-TO-IMPL-MAPPING”](#).

8.1.2.8.93. SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING

SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING is upgraded as described in [Section 8.1.1.8.89, “SENDER-RECEIVER-TO-SIGNAL-GROUP-MAPPING”](#).

8.1.2.8.94. SENDER-RECEIVER-TO-SIGNAL-MAPPING

SENDER-RECEIVER-TO-SIGNAL-MAPPING is upgraded as described in [Section 8.1.1.8.90, “SENDER-RECEIVER-TO-SIGNAL-MAPPING”](#).

8.1.2.8.95. CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING

CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING is upgraded as described in [Section 8.1.1.8.91, “CLIENT-SERVER-TO-SIGNAL-GROUP-MAPPING”](#).

8.1.2.8.96. SENDER-REC-RECORD-TYPE-MAPPING

SENDER-REC-RECORD-TYPE-MAPPING is upgraded as described in [Section 8.1.1.8.92, “SENDER-REC-RECORD-TYPE-MAPPING”](#).

8.1.2.8.97. SENDER-REC-RECORD-ELEMENT-MAPPING

SENDER-REC-RECORD-ELEMENT-MAPPING is upgraded as described in [Section 8.1.1.8.93, “SENDER-REC-RECORD-ELEMENT-MAPPING”](#).

8.1.2.8.98. SENDER-REC-ARRAY-TYPE-MAPPING

SENDER-REC-ARRAY-TYPE-MAPPING is upgraded as described in [Section 8.1.1.8.94, “SENDER-REC-ARRAY-TYPE-MAPPING”](#).

8.1.2.8.99. SENDER-REC-ARRAY-ELEMENT-MAPPING

SENDER-REC-ARRAY-ELEMENT-MAPPING is upgraded as described in [Section 8.1.1.8.95, “SENDER-REC-ARRAY-ELEMENT-MAPPING”](#).



8.1.2.8.100. PNC-MAPPING

Source model entity: PNC-MAPPING

Target model entity: PNC-MAPPING

Target model property	Source model property
PNC-GROUP-REFS / PNC-GROUP-REF	PNC-GROUP-REFS / PNC-GROUP-REF
PNC-IDENTIFIER	PNC-IDENTIFIER
VFC-IREFS / VFC-IREF	VFC-IREFS / VFC-IREF
WAKEUP-FRAME-REFS / WAKEUP-FRAME-REF	WAKEUP-FRAME-REFS / WAKEUP-FRAME-REF

8.1.2.8.101. END-TO-END-PROTECTION-SET

END-TO-END-PROTECTION-SET is upgraded as described in [Section 8.1.1.8.96, “END-TO-END-PROTECTION-SET”](#).

8.1.2.8.102. END-TO-END-PROTECTION

END-TO-END-PROTECTION is upgraded as described in [Section 8.1.1.8.97, “END-TO-END-PROTECTION”](#).

8.1.2.8.103. END-TO-END-PROTECTION-I-SIGNAL-I-PDU

END-TO-END-PROTECTION-I-SIGNAL-I-PDU is upgraded as described in [Section 8.1.1.8.98, “END-TO-END-PROTECTION-I-SIGNAL-I-PDU”](#).

8.1.2.8.104. END-TO-END-PROTECTION-VARIABLE-PROTOTYPE

END-TO-END-PROTECTION-VARIABLE-PROTOTYPE is upgraded as described in [Section 8.1.1.8.99, “END-TO-END-PROTECTION-VARIABLE-PROTOTYPE”](#).

8.1.3. Upgrading a system model from AUTOSAR 3.2.2 to AUTOSAR 4.0.3

This section describes how an AUTOSAR 3.2.2 system model is upgraded to AUTOSAR 4.0.3.

Since the AUTOSAR 3.2.1 and 3.2.2 source models are similar, this chapter only describes those parts of the upgrade that differ from the 3.2.1 to 4.0.3 upgrade. For all other parts, see [Section 8.1.2, “Upgrading a system model from AUTOSAR 3.2.x to AUTOSAR 4.0.3”](#).



If not stated explicitly, all entities in the 4.0.3 target model are located in the same package as they were in the 3.2.2 source model.

If not specified explicitly otherwise, the parameters `SHORT-NAME`, `CATEGORY`, and `UUID` of Identifiable entities are upgraded directly into their counterparts in the target model.

8.1.3.1. Topology

8.1.3.1.1. CAN-CLUSTER

Source model entity: CAN-CLUSTER

Target model entity: CAN-CLUSTER

Target model property	Source model property
CAN-CLUSTER-CONDITIONAL/PHYSICAL-CHANNELS	PHYSICAL-CHANNELS
CAN-CLUSTER-CONDITIONAL/PROTOCOL-NAME	PROTOCOL-NAME
CAN-CLUSTER-CONDITIONAL/PROTOCOL-VERSION	PROTOCOL-VERSION
CAN-CLUSTER-CONDITIONAL/SPEED	SPEED/1000
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT
No target model parameter is available.	NM-COORDINATOR-SYNC-PRIO

8.1.3.1.1.1. Changed or moved parameters

`NM-NODE-DETECTION-ENABLED` and `NM-NODE-ID-ENABLED`

The source model parameters `NM-NODE-DETECTION-ENABLED` and `NM-NODE-ID-ENABLED` are configured in the target model entity `NM-ECU`, see [Section 8.1.2.7.2, “NM-ECU”](#).

`SPEED`

In the target model, the parameter `SPEED` is given in kbit/s, whereas the source model provides it in bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.

`NM-BUS-LOAD-REDUCTION-ACTIVE`, `NM-WAIT-BUS-SLEEP-TIME`, `NM-IMMEDIATE-NM-CYCLE-TIME`, `NM-IMMEDIATE-NM-TRANSMISSIONS`, `NM-MSG-CYCLE-TIME`, and `NM-REMOTE-SLEEP-INDICATION-TIME`

The parameters `NM-BUS-LOAD-REDUCTION-ACTIVE`, `NM-WAIT-BUS-SLEEP-TIME`, `NM-IMMEDIATE-NM-CYCLE-TIME`, `NM-IMMEDIATE-NM-TRANSMISSIONS`, `NM-MSG-CYCLE-TIME`, and `NM-RE-`



MOTE-SLEEP-INDICATION-TIME are configured as NM-BUSLOAD-REDUCTION-ACTIVE, NM-WAIT-BUS-SLEEP-TIME, NM-IMMEDIATE-NM-CYCLE-TIME, NM-IMMEDIATE-NM-TRANSMISSIONS, NM-MSG-CYCLE-TIME, and NM-REMOTE-SLEEP-INDICATION-TIME in the target model entity CAN-NM-CLUSTER, see [Section 8.1.2.7.3, “CAN-NM-CLUSTER”](#).

NM-BUS-LOAD-REDUCTION-ENABLED

The parameter NM-BUS-LOAD-REDUCTION-ENABLED is configured as NM-BUSLOAD-REDUCTION-ENABLED in the target model entity CAN-NM-CLUSTER-COUPLING, see [Section 8.1.1.7.8, “FLEXRAY-NM-CLUSTER-COUPLING”](#).

NM-LOWER-CAN-ID and NM-UPPER-CAN-ID

The parameters NM-LOWER-CAN-ID and NM-UPPER-CAN-ID are configured in the target model entity CAN-NM-RANGE-CONFIG parameters LOWER-CAN-ID and UPPER-CAN-ID, which is aggregated in CAN-NM-NODE, see [Section 8.1.1.7.4, “CAN-NM-NODE”](#).

8.1.3.1.2. CAN-COMMUNICATION-CONNECTOR

Source model entity: CAN-COMMUNICATION-CONNECTOR

Target model entity: CAN-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
PNC-WAKEUP-CAN-ID	PNC-WAKEUP-CAN-ID
PNC-WAKEUP-CAN-ID-EXTENDED	PNC-WAKEUP-CAN-ID-EXTENDED
PNC-WAKEUP-CAN-ID-MASK	PNC-WAKEUP-CAN-ID-MASK
PNC-WAKEUP-DATA-MASK	PNC-WAKEUP-DATA-MASK
PNC-WAKEUP-DLC	PNC-WAKEUP-DLC
PNC-GATEWAY-TYPE	PNC-GATEWAY-TYPE
No target model parameter is available.	NM-ENABLED

8.1.3.1.2.1. Changed or moved parameters

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity CAN-PHYSICAL-CHANNEL.



NM-ADDRESS

The source model parameter NM-ADDRESS is configured in the target model entity CAN-NM-NODE, see [Section 8.1.2.7.4, “CAN-NM-NODE”](#).

8.1.3.1.3. FLEXRAY-CLUSTER

Source model entity: FLEXRAY-CLUSTER

Target model entity: FLEXRAY-CLUSTER/FLEXRAY-CLUSTER-CONDITIONAL

Target model property	Source model property
PHYSICAL-CHANNELS	PHYSICAL-CHANNELS
PROTOCOL-NAME	PROTOCOL-NAME
PROTOCOL-VERSION	PROTOCOL-VERSION
SPEED	SPEED/1000
ACTION-POINT-OFFSET	ACTION-POINT-OFFSET
BIT	BIT
CAS-RX-LOW-MAX	CAS-RX-LOW-MAX
COLD-START-ATTEMPTS	COLD-START-ATTEMPTS
CYCLE	CYCLE
CYCLE-COUNT-MAX	CYCLE-COUNT-MAX
DYNAMIC-SLOT-IDLE-PHASE	DYNAMIC-SLOT-IDLE-PHASE
LISTEN-NOISE	LISTEN-NOISE
MACRO-PER-CYCLE	MACRO-PER-CYCLE
MACROTICK-DURATION	MACROTICK-DURATION
MAX-WITHOUT-CLOCK-CORRECTION-FATAL	MAX-WITHOUT-CLOCK-CORRECTION-FATAL
MAX-WITHOUT-CLOCK-CORRECTION-PASSIVE	MAX-WITHOUT-CLOCK-CORRECTION-PASSIVE
MINISLOT-ACTION-POINT-OFFSET	MINISLOT-ACTION-POINT-OFFSET
MINISLOT-DURATION	MINISLOT-DURATION
NETWORK-IDLE-TIME	NETWORK-IDLE-TIME
NETWORK-MANAGEMENT-VECTOR-LENGTH	NETWORK-MANAGEMENT-VECTOR-LENGTH
NUMBER-OF-MINISLOTS	NUMBER-OF-MINISLOTS
NUMBER-OF-STATIC-SLOTS	NUMBER-OF-STATIC-SLOTS
OFFSET-CORRECTION-START	OFFSET-CORRECTION-START
PAYLOAD-LENGTH-STATIC	PAYLOAD-LENGTH-STATIC



Target model property	Source model property
SAMPLE-CLOCK-PERIOD	SAMPLE-CLOCK-PERIOD
STATIC-SLOT-DURATION	STATIC-SLOT-DURATION
SYMBOL-WINDOW	SYMBOL-WINDOW
SYMBOL-WINDOW-ACTION-POINT-OFFSET	SYMBOL-WINDOW-ACTION-POINT-OFFSET
SYNC-FRAME-ID-COUNT-MAX	SYNC-FRAME-ID-COUNT-MAX
TRANSMISSION-START-SEQUENCE-DURATION	TRANSMISSION-START-SEQUENCE-DURATION
WAKEUP-RX-IDLE	WAKEUP-RX-IDLE
WAKEUP-RX-LOW	WAKEUP-RX-LOW
WAKEUP-RX-WINDOW	WAKEUP-RX-WINDOW
WAKEUP-TX-ACTIVE	WAKEUP-TX-ACTIVE
WAKEUP-TX-IDLE	WAKEUP-TX-IDLE
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT
No target model parameter is available.	BUS-GUARDIAN-ENABLE-PART
No target model parameter is available.	CAS-RX-LOW-MIN
No target model parameter is available.	MACRO-INITIAL-OFFSET
No target model parameter is available.	MAX-INITIALISATION-ERROR
No target model parameter is available.	MAX-PROPAGATION-DELAY
No target model parameter is available.	MIN-PROPAGATION-DELAY
No target model parameter is available.	OFFSET-CORRECTION-MAX
DETECT-NIT-ERROR	No source model parameter is available.
IGNORE-AFTER-TX	IGNORE-AFTER-TX
SAFETY-MARGIN	No source model parameter is available.
No target model parameter is available.	TRANSCEIVER-STANDBY-DELAY
No target model parameter is available.	NM-COORDINATOR-SYNC-PRIO

8.1.3.1.3.1. Changed or moved parameters

NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED

The source model parameters NM-NODE-DETECTION-ENABLED and NM-NODE-ID-ENABLED are configured in the target model entity NM-ECU, see [Section 8.1.2.7.2, “NM-ECU”](#).



SPEED

In the target model, the parameter SPEED is given in kbit/s, whereas the source model provides it in bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.

NM-DATA-CYCLE, NM-READY-SLEEP-COUNT, NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPEAT-MESSAGE-TIME, NM-REPETITION-CYCLE, NM-VOTING-CYCLE

The source model parameters NM-DATA-CYCLE, NM-READY-SLEEP-COUNT, NM-REMOTE-SLEEP-INDICATION-TIME, NM-REPEAT-MESSAGE-TIME, NM-REPETITION-CYCLE, and NM-VOTING-CYCLE are configured in the target model entity FLEXRAY-NM-CLUSTER, see [Section 8.1.2.7.6, "FLEXRAY-NM-CLUSTER"](#).

8.1.3.1.3.2. Reason for not upgraded source model parameter

MAX-FRAME-LENGTH, BUS-GUARDIAN-ENABLE-PART, CAS-RX-LOW-MIN, CAS-RX-LOW-MIN, MACRO-INITIAL-OFFSET, MAX-INITIALISATION-ERROR, MAX-PROPAGATION-DELAY, MIN-PROPAGATION-DELAY, OFFSET-CORRECTION-MAX, NM-REPEAT-MESSAGE-SUPPORT

The parameters MAX-FRAME-LENGTH, BUS-GUARDIAN-ENABLE-PART, CAS-RX-LOW-MIN, CAS-RX-LOW-MIN, MACRO-INITIAL-OFFSET, MAX-INITIALISATION-ERROR, MAX-PROPAGATION-DELAY, MIN-PROPAGATION-DELAY, OFFSET-CORRECTION-MAX, NM-REPEAT-MESSAGE-SUPPORT are not upgraded to the target model, because they do not contribute any data required for configuring an ECU. Therefore, they have been removed from the target model.

8.1.3.1.4. FLEXRAY-COMMUNICATION-CONNECTOR

Source model entity: FLEX-RAY-COMMUNICATION-CONNECTOR

Target model entity: FLEXRAY-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
WAKE-UP-CHANNEL	WAKE-UP-CHANNEL
PNC-GATEWAY-TYPE	PNC-GATEWAY-TYPE
No target model parameter is available.	NM-READY-SLEEP-TIME
No target model parameter is available.	NM-ENABLED



8.1.3.1.4.1. Changed or moved parameters

ECU-COMM-PORT-INSTANCES

In ECU-COMM-PORT-INSTANCES, the aggregated source model entities are upgraded to their target model counterparts: FRAME-PORT, I-PDU-PORT, SIGNAL-PORT.

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity FLEXRAY-PHYSICAL-CHANNEL.

NM-ADDRESS

The source model parameter NM-ADDRESS is configured in the target model entity FLEXRAY-NM-NODE, see [Section 8.1.2.7.7, "FLEXRAY-NM-NODE"](#).

8.1.3.1.5. LIN-CLUSTER

Source model entity: LIN-CLUSTER

Target model entity: LIN-CLUSTER/LIN-CLUSTER-CONDITIONAL

Target model property	Source model property
PHYSICAL-CHANNELS	PHYSICAL-CHANNELS
PROTOCOL-NAME	PROTOCOL-NAME
PROTOCOL-VERSION	PROTOCOL-VERSION
SPEED	SPEED/1000
No target model parameter is available.	MAX-FRAME-LENGTH
No target model parameter is available.	NM-REPEAT-MESSAGE-SUPPORT
No target model parameter is available.	NM-COORDINATOR-SYNC-PRIO

8.1.3.1.5.1. Changed or moved parameters

SPEED

In the target model, the parameter SPEED is given in kbit/s, whereas the source model provides it in bit/s. If the source model parameter is not divisible by 1000 without rest, it cannot be upgraded at all.

LIN-SCHEDULE-TABLE entities of the source model parameter SCHEDULE-TABLES

The LIN-SCHEDULE-TABLE entities aggregated in the source model in SCHEDULE-TABLES are moved to the corresponding LIN-PHYSICAL-CHANNEL in the target model, see [Section 8.1.1.16, "LIN-PHYSICAL-CHANNEL"](#).



8.1.3.1.6. LIN-COMMUNICATION-CONNECTOR

Source model entity: COMMUNICATION-CONNECTOR

Target model entity: LIN-COMMUNICATION-CONNECTOR

Target model property	Source model property
CATEGORY	CATEGORY
COMM-CONTROLLER-REF	COMM-CONTROLLER-REF
ECU-COMM-PORT-INSTANCES	ECU-COMM-PORT-INSTANCES
LIN-CONFIGURABLE-FRAMES/LIN-CONFIGURABLE-FRAME	No source model parameter is available.
LIN-ORDERED-CONFIGURABLE-FRAMES/LIN-ORDERED-CONFIGURABLE-FRAME	No source model parameter is available.
PNC-GATEWAY-TYPE	PNC-GATEWAY-TYPE
No target model parameter is available.	NM-ENABLED

8.1.3.1.6.1. Changed or moved parameters

ECU-COMM-PORT-INSTANCES

In ECU-COMM-PORT-INSTANCES, the aggregated source model entities are upgraded into their target model counterparts: FRAME-PORT, I-PDU-PORT, SIGNAL-PORT.

CHANNEL-REF

The source model parameter CHANNEL-REF has been replaced by the target model parameter COMM-CONNECTORS/COMMUNICATION-CONNECTOR-REF-CONDITIONAL of the target model entity LIN-PHYSICAL-CHANNEL.

INITIAL-NAD

The target model parameter INITIAL-NAD is configured by taking the CONFIGURED-NAD parameter value of the LIN-SLAVE which is referenced via COMM-CONTROLLER-REF.

8.1.3.1.7. LIN-SLAVE-CONFIG

The LIN-SLAVE-CONFIG entity is introduced in AUTOSAR 3.2.2 to model LIN-SLAVE parameters without the need to add a complete ECU-INSTANCE. Since the target model AUTOSAR 4.0.x does not support LIN-SLAVE-CONFIG, the source model LIN-SLAVE-CONFIG entity is translated into one ECU-INSTANCE, one LIN-SLAVE, and one LIN-COMMUNICATION-CONNECTOR in the target model.



8.1.3.1.7.1. New parameters in the target model

ECU-INSTANCE

A new ECU-INSTANCE is added to the target model for each source model LIN-SLAVE-CONFIG. Its SHORT-NAME parameter is set to LinSlave_<configured_nad>, where <configured_nad> is the CONFIGURED-NAD parameter value of the source model LIN-SLAVE-CONFIG. One LIN-SLAVE element is added to its COMM-CONTROLLERS parameter, one LIN-COMMUNICATION-CONNECTOR element is added to its CONNECTORS parameter.

LIN-SLAVE

A new LIN-SLAVE is added to the target model for each source model LIN-SLAVE-CONFIG. Its SHORT-NAME parameter is set to LinSlave_<configured_nad>_CommController, where <configured_nad> is the CONFIGURED-NAD parameter value of the source model LIN-SLAVE-CONFIG. The source model LIN-SLAVE-CONFIG/LIN-ERROR-RESPONSE parameter is upgraded to LIN-SLAVE/LIN-SLAVE-VARIANTS/LIN-SLAVE-CONDITIONAL/LIN-ERROR-RESPONSE in the target model. The source model parameters LIN-SLAVE-CONFIG/CONFIGURED-NAD and LIN-SLAVE-CONFIG/PROTOCOL-VERSION are upgraded to LIN-SLAVE/LIN-SLAVE-VARIANTS/LIN-SLAVE-CONDITIONAL/CONFIGURED-NAD and LIN-SLAVE/LIN-SLAVE-VARIANTS/LIN-SLAVE-CONDITIONAL/PROTOCOL-VERSION.

LIN-COMMUNICATION-CONNECTOR

A new LIN-COMMUNICATION-CONNECTOR is added to the target model for each source model LIN-SLAVE-CONFIG. Its SHORT-NAME parameter is set to LinSlave_<configured_nad>_CommConnector, where <configured_nad> is the CONFIGURED-NAD parameter value of the source model LIN-SLAVE-CONFIG. LIN-COMMUNICATION-CONNECTOR/COMM-CONTROLLER-REF references the target model LIN-SLAVE element that is created for the source model LIN-SLAVE-CONFIG. Moreover, a COMMUNICATION-CONNECTOR-REF-CONDITIONAL parameter that references this LIN-COMMUNICATION-CONNECTOR is added to all LIN-PHYSICAL-CHANNELS which also reference any of the LIN-COMMUNICATION-CONNECTORS that belong to the target model LIN-MASTER counterpart of the source model LIN-MASTER element which aggregates the LIN-SLAVE-CONFIG.

8.1.3.2. FlexRay ISO Transport Layer

NOTE

Non-ISO FlexRay Transport Layer upgrade is not supported



The system model upgrade only supports the upgrade of the FlexRay ISO Transport Layer. The upgrade of the Non-ISO FlexRay Transport Layer variant is *not* supported.

8.1.3.2.1. FLEXRAY-TP-ECU

Source model entity: ECU-INSTANCE



Target model entity: FLEXRAY-TP-ECU

Target model property	Source model property
CYCLE-TIME-MAIN-FUNCTION	CYCLE-TIME-MAIN-FUNCTION
FULL-DUPLEX-ENABLED	FULL-DUPLEX-ENABLED
No target model parameter is available.	TRANSMIT-CANCELLATION

8.1.3.3. Network Management

8.1.3.3.1. NM-COORDINATOR

In the following all entities used to create an NM-COORDINATOR are located in the target model if not stated otherwise.

To every NM-ECU, which is referenced by at least two NM-NODES of different NM-CLUSTERS, a NM-COORDINATOR is added. The NM-CLUSTERS must also reference COMMUNICATION-CLUSTERS which have an NM-GLOBAL-COORDINATOR-TIME defined in the corresponding COMMUNICATION-CLUSTER of the source model. NM-GLOBAL-COORDINATOR-TIME of the NM-COORDINATOR is then set to the maximum of all NM-GLOBAL-COORDINATOR-TIME values found in these COMMUNICATION-CLUSTERS.

For each NM-NODE referencing the NM-ECU to which an NM-COORDINATOR has been added, a NM-NODE-REF referencing the NM-NODE is added to the NM-COORDINATOR's NM-NODE-REFS.

8.1.3.4. Software Component Description

8.1.3.4.1. APPLICATION-SW-COMPONENT-TYPE

Source model entities: APPLICATION-SOFTWARE-COMPONENT-TYPE, SENSOR-ACTUATOR-SOFT-WARE-COMPONENT-TYPE

Target model entity: APPLICATION-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
SYMBOL-PROPS/SHORT-NAME	SYMBOL-PROPS/SHORT-NAME
SYMBOL-PROPS/SYMBOL	SYMBOL-PROPS/SYMBOL



8.1.3.4.1.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.3.4.2. SERVICE-SW-COMPONENT-TYPE

Source model entity: SERVICE-COMPONENT-TYPE

Target model entity: SERVICE-SW-COMPONENT-TYPE

Target model property	Source model property
POR TS/P-PORT-PROTOTYPE	POR TS/P-PORT-PROTOTYPE
POR TS/R-PORT-PROTOTYPE	POR TS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
SYMBOL-PROPS/SHORT-NAME	SYMBOL-PROPS/SHORT-NAME
SYMBOL-PROPS/SYMBOL	SYMBOL-PROPS/SYMBOL
No upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MODULE-DESCRIPTION-REF

8.1.3.4.2.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.3.4.3. COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE

Source model entity: COMPLEX-DEVICE-DRIVER-COMPONENT-TYPE

Target model entity: COMPLEX-DEVICE-DRIVER-SW-COMPONENT-TYPE

Target model property	Source model property
POR TS/P-PORT-PROTOTYPE	POR TS/P-PORT-PROTOTYPE
POR TS/R-PORT-PROTOTYPE	POR TS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP



Target model property	Source model property
SYMBOL-PROPS/SHORT-NAME	SYMBOL-PROPS/SHORT-NAME
SYMBOL-PROPS/SYMBOL	SYMBOL-PROPS/SYMBOL
No upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MODULE-DESCRIPTION-REF
No upgrade is currently implemented.	HARDWARE-ELEMENT-REFS/HARDWARE-ELEMENT-REF

8.1.3.4.3.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.3.4.4. ECU-ABSTRACTION-SW-COMPONENT-TYPE

Source model entity: ECU-ABSTRACTION-COMPONENT-TYPE

Target model entity: ECU-ABSTRACTION-SW-COMPONENT-TYPE

Target model property	Source model property
PORTS/P-PORT-PROTOTYPE	PORTS/P-PORT-PROTOTYPE
PORTS/R-PORT-PROTOTYPE	PORTS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
SYMBOL-PROPS/SHORT-NAME	SYMBOL-PROPS/SHORT-NAME
SYMBOL-PROPS/SYMBOL	SYMBOL-PROPS/SYMBOL
No upgrade is currently implemented.	BSW-MODULE-DESCRIPTION-REFS/BSW-MODULE-DESCRIPTION-REF
No upgrade is currently implemented.	HARDWARE-ELEMENT-REFS/HARDWARE-ELEMENT-REF

8.1.3.4.4.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.



8.1.3.4.5. NV-BLOCK-SW-COMPONENT-TYPE

Source model entity: NV-BLOCK-SW-COMPONENT-TYPE

Target model entity: NV-BLOCK-SW-COMPONENT-TYPE

Target model property	Source model property
POR TS/P-PORT-PROTOTYPE	POR TS/P-PORT-PROTOTYPE
POR TS/R-PORT-PROTOTYPE	POR TS/R-PORT-PROTOTYPE
PORT-GROUPS/PORT-GROUP	PORT-GROUPS/PORT-GROUP
SYMBOL-PROPS/SHORT-NAME	SYMBOL-PROPS/SHORT-NAME
SYMBOL-PROPS/SYMBOL	SYMBOL-PROPS/SYMBOL
NV-BLOCK-DESCRIPTORS/NV-BLOCK-DESCRIP-TOR	NV-BLOCK-DESCRIPTORS/NV-BLOCK-DESCRIP-TOR

8.1.3.4.5.1. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.3.4.6. NV-BLOCK-DESCRIPTOR

Source model entity: NV-BLOCK-DESCRIPTOR

Target model entity: NV-BLOCK-DESCRIPTOR

Target model property	Source model property
CLIENT-SERVER-PORTS/ROLE-BASED-PORT-ASSIGNMENT	CLIENT-SERVER-PORTS/ROLE-BASED-PORT-ASSIGNMENT
NV-BLOCK-DATA-MAPPINGS/NV-BLOCK-DA-TA-MAPPING	NV-BLOCK-DATA-MAPPINGS/NV-BLOCK-DA-TA-MAPPING
NV-BLOCK-NEEDS	NV-BLOCK-NEEDS, SWC-NV-BLOCK-NEEDS
RAM-BLOCK	RAM-BLOCK
RAM-BLOCK/INIT-VALUE/CONSTANT-REFERENCE	RAM-BLOCK-INIT-VALUE-REF
ROM-BLOCK	ROM-BLOCK



Target model property	Source model property
ROM-BLOCK/INIT-VALUE/CONSTANT-REFERENCE	ROM-BLOCK-INIT-VALUE-REF

8.1.3.4.6.1. No configuration of target model parameters CONSTANT-VALUE-MAPPING-REFS, DATA-TYPE-MAPPING-REFS, INSTANTIATION-DATA-DEF-PROPS

The target model parameters CONSTANT-VALUE-MAPPING-REFS, DATA-TYPE-MAPPING-REFS, INSTANTIATION-DATA-DEF-PROPS are not configured because there are no source model counterparts.

8.1.3.4.6.2. New parameters in the target model

INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/SWC-INTERNAL-BEHAVIOR contains the SWC-INTERNAL-BEHAVIOR which corresponds to the source model INTERNAL-BEHAVIOR which references the APPLICATION-SOFTWARE-COMPONENT-TYPE via COMPONENT-REF.

8.1.3.4.7. NV-BLOCK-DATA-MAPPING

Source model entity: NV-BLOCK-DATA-MAPPING

Target model entity: NV-BLOCK-DATA-MAPPING

Target model property	Source model property
No target model parameter is available.	NV-RAM-BLOCK-ELEMENT-REF/ARRAY-INDEX
NV-RAM-BLOCK-ELEMENT/LOCAL-VARIABLE-REF	NV-RAM-BLOCK-ELEMENT-REF/SUB-ELEMENT-REF
READ-NV-DATA/AUTOSAR-VARIABLE-IREF/PORT-PROTOTYPE-REF	READ-NV-DATA/DATA-PROTOTYPE-IREF/PORT-PROTOTYPE-REF
READ-NV-DATA/AUTOSAR-VARIABLE-IREF/TARGET-DATA-PROTOTYPE-REF	READ-NV-DATA/DATA-PROTOTYPE-IREF/DATA-PROTOTYPE-REF
WRITTEN-NV-DATA/AUTOSAR-VARIABLE-IREF/PORT-PROTOTYPE-REF	WRITTEN-NV-DATA/DATA-PROTOTYPE-IREF/PORT-PROTOTYPE-REF
WRITTEN-NV-DATA/AUTOSAR-VARIABLE-IREF/TARGET-DATA-PROTOTYPE-REF	WRITTEN-NV-DATA/DATA-PROTOTYPE-IREF/DATA-PROTOTYPE-REF

8.1.3.4.8. ROLE-BASED-PORT-ASSIGNMENT

Source model entities: ROLE-BASED-P-PORT-ASSIGNMENT, ROLE-BASED-R-PORT-ASSIGNMENT



Target model entity: ROLE-BASED-PORT-ASSIGNMENT

Target model property	Source model property
PORT-PROTOTYPE-REF	P-PORT-PROTOTYPE-REF, R-PORT-PROTOTYPE-REF
ROLE	ROLE

8.1.3.4.9. P-PORT-PROTOTYPE

Source model entity: P-PORT-PROTOTYPE

Target model entity: P-PORT-PROTOTYPE

Target model property	Source model property
PROVIDED-COM-SPECS/MODE-SWITCH-SEN-DER-COM-SPEC	PROVIDED-COM-SPECS/MODE-SWITCH-COM-SPEC
PROVIDED-COM-SPECS/NONQUEUED-SEN-DER-COM-SPEC	PROVIDED-COM-SPECS/UNQUEUED-SENDER-COM-SPEC
PROVIDED-COM-SPECS/NV-PROVIDE-COM-SPEC	PROVIDED-COM-SPECS/NV-PROVIDE-COM-SPEC
PROVIDED-COM-SPECS/PARAMETER-PROVIDE-COM-SPEC	PROVIDED-COM-SPECS/PARAMETER-PROVIDE-COM-SPEC
PROVIDED-COM-SPECS/QUEUED-SENDER-COM-SPEC	PROVIDED-COM-SPECS/QUEUED-SENDER-COM-SPEC
PROVIDED-COM-SPECS/SERVER-COM-SPEC	PROVIDED-COM-SPECS/SERVER-COM-SPEC
PROVIDED-INTERFACE-TREF	PROVIDED-INTERFACE-TREF

8.1.3.4.10. NONQUEUED-RECEIVER-COM-SPEC

Source model entity: UNQUEUED-RECEIVER-COM-SPEC

Target model entity: NONQUEUED-RECEIVER-COM-SPEC

Target model property	Source model property
ALIVE-TIMEOUT	ALIVE-TIMEOUT
No target model parameter is available.	HANDLE-INVALID
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF
No target model parameter is available.	RESYNC-TIME



Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
FILTER	FILTER
HANDLE-NEVER-RECEIVED	HANDLE-NEVER-RECEIVED if the source model parameter value is available, else false.
ENABLE-UPDATE	ENABLE-UPDATE
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.
HANDLE-TIMEOUT-TYPE	No source model parameter is available, the target model parameter is set to NONE.
NETWORK-REPRESENTATION	No source model parameter is available.
USES-END-TO-END-PROTECTION	USES-END-TO-END-PROTECTION
EXTERNAL-REPLACEMENT-REF	No source model parameter is available.
MAX-DELTA-COUNTER-INIT	MAX-DELTA-COUNTER-INIT
No target model parameter is available.	MAX-NO-NEW-OR-REPEATED-DATA
No target model parameter is available.	SYNC-COUNTER-INIT

8.1.3.4.10.1. Processing of source model parameter HANDLE-INVALID

If the parameter HANDLE-INVALID is provided in the source model, a INVALIDATION-POLICY element is created in the related target model SENDER-RECEIVER-INTERFACE. The INVALIDATION-POLICY parameter HANDLE-INVALID is configured according to the source model UNQUEUED-RECEIVER-COM-SPEC parameter HANDLE-INVALID if all UNQUEUED-RECEIVER-COM-SPEC elements belonging to the source model SENDER-RECEIVER-INTERFACE and DATA-ELEMENT are providing the same HANDLE-INVALID value.

8.1.3.4.11. QUEUED-RECEIVER-COM-SPEC

Source model entity: QUEUED-RECEIVER-COM-SPEC

Target model entity: QUEUED-RECEIVER-COM-SPEC

Target model property	Source model property
DATA-ELEMENT-REF	DATA-ELEMENT-REF
No target model parameter is available.	FILTER
HANDLE-OUT-OF-RANGE	No source model parameter is available, the target model parameter is set to NONE.



Target model property	Source model property
HANDLE-OUT-OF-RANGE-STATUS	No source model parameter is available.
NETWORK-REPRESENTATION	No source model parameter is available.
QUEUE-LENGTH	QUEUE-LENGTH
USES-END-TO-END-PROTECTION	No source model parameter is available, the target model parameter is set to false.
MAX-DELTA-COUNTER-INIT	MAX-DELTA-COUNTER-INIT
No target model parameter is available.	MAX-NO-NEW-OR-REPEATED-DATA
No target model parameter is available.	SYNC-COUNTER-INIT

8.1.3.4.12. NV-PROVIDE-COM-SPEC

Source model entity: NV-PROVIDE-COM-SPEC

Target model entity: NV-PROVIDE-COM-SPEC

Target model property	Source model property
VARIABLE-REF	NV-DATA-ELEMENT-REF
RAM-BLOCK-INIT-VALUE/CONSTANT-REFERENCE	RAM-BLOCK-INIT-VALUE-REF
ROM-BLOCK-INIT-VALUE/CONSTANT-REFERENCE	ROM-BLOCK-INIT-VALUE-REF

8.1.3.4.13. RUNNABLE-ENTITY

Source model entity: RUNNABLE-ENTITY

Target model entity: RUNNABLE-ENTITY

Target model property	Source model property
ARGUMENTS/RUNNABLE-ENTITY-ARGUMENT	ARGUMENTS/RUNNABLE-ENTITY-ARGUMENT
Currently no upgrade is available.	BSW-ENTITY-REF
PARAMETER-ACCESSSS/PARAMETER-ACCESS/ACCESSED-PARAMETER	CALPRM-ACCESSSS/CALPRM-ACCESS
CAN-BE-INVOKED-CONCURRENTLY	CAN-BE-INVOKED-CONCURRENTLY
CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF	CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF
DATA-READ-ACCESSSS/VARIABLE-ACCESS	DATA-READ-ACCESSSS/DATA-READ-ACCESS



Target model property	Source model property
DATA-RECEIVE-POINT-BY-ARGUMENTS/VARIABLE-ACCESS	DATA-RECEIVE-POINTS/DATA-RECEIVE-POINT
DATA-SEND-POINTS/VARIABLE-ACCESS	DATA-SEND-POINTS/DATA-SEND-POINT
DATA-WRITE-ACCESSSS/VARIABLE-ACCESS	DATA-WRITE-ACCESSSS/DATA-WRITE-ACCESS
MINIMUM-START-INTERVAL	MINIMUM-START-INTERVAL
MODE-SWITCH-POINTS/MODE-SWITCH-POINT	MODE-SWITCH-POINTS/MODE-SWITCH-POINT
PARAMETER-ACCESSSS/PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	PER-INSTANCE-CALPRM-ACCESS-REFS/PER-INSTANCE-CALPRM-ACCESS-REF
RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF	RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF
SERVER-CALL-POINTS/ASYNCHRONOUS-SERVER-CALL-POINT	SERVER-CALL-POINTS/ASYNCHRONOUS-SERVER-CALL-POINT
SERVER-CALL-POINTS/SYNCHRONOUS-SERVER-CALL-POINT	SERVER-CALL-POINTS/SYNCHRONOUS-SERVER-CALL-POINT
PARAMETER-ACCESSSS/PARAMETER-ACCESS/ACCESSED-PARAMETER/LOCAL-PARAMETER-REF	SHARED-CALPRM-ACCESS-REFS/SHARED-CALPRM-ACCESS-REF
SYMBOL	SYMBOL
WAIT-POINTS/WAIT-POINT	WAIT-POINTS/WAIT-POINT
WRITTEN-LOCAL-VARIABLES/VARIABLE-ACCESS/ACCESSED-VARIABLE/LOCAL-VARIABLE-REF	WRITTEN-VARIABLE-REFS/WRITTEN-VARIABLE-REF
READ-LOCAL-VARIABLES/VARIABLE-ACCESS/ACCESSED-VARIABLE/LOCAL-VARIABLE-REF	READ-VARIABLE-REFS/READ-VARIABLE-REF

8.1.3.4.13.1. Migration of source model DATA-RECEIVE-POINT elements

Note that for all source model DATA-RECEIVE-POINT elements are migrated to DATA-RECEIVE-POINT-BY-ARGUMENTS/VARIABLE-ACCESS elements in the target model, i.e. there is no conversion to the target model entities DATA-RECEIVE-POINT-BY-VALUES/VARIABLE-ACCESS.

8.1.3.4.13.2. Creation of target model ASYNCHRONOUS-SERVER-CALL-RESULT-POINT elements

Note that for every source model ASYNCHRONOUS-SERVER-CALL-RETURNS-EVENT entity which either references a RUNNABLE-ENTITY via START-ON-EVENT-REF or references a SERVER-CALL-POINT of



the RUNNABLE-ENTITY via EVENT-SOURCE-REF, one ASYNCHRONOUS-SERVER-CALL-RESULT-POINTS/ASYNCHRONOUS-SERVER-CALL-RESULT-POINT is created in the target model.

8.1.3.4.14. R-PORT-PROTOTYPE

Source model entity: R-PORT-PROTOTYPE

Target model entity: R-PORT-PROTOTYPE

Target model property	Source model property
REQUIRED-COM-SPECS/CLIENT-COM-SPEC	REQUIRED-COM-SPECS/CLIENT-COM-SPEC
REQUIRED-COM-SPECS/PARAMETER-REQUIRE-COM-SPEC	REQUIRED-COM-SPECS/PARAMETER-REQUIRE-COM-SPEC
REQUIRED-COM-SPECS/QUEUED-RECEIVER-COM-SPEC	REQUIRED-COM-SPECS/QUEUED-RECEIVER-COM-SPEC
REQUIRED-COM-SPECS/NONQUEUED-RECEIVER-COM-SPEC	REQUIRED-COM-SPECS/UNQUEUED-RECEIVER-COM-SPEC
REQUIRED-COM-SPECS/NV-REQUIRE-COM-SPEC	REQUIRED-COM-SPECS/NV-REQUIRE-COM-SPEC
REQUIRED-INTERFACE-TREF	REQUIRED-INTERFACE-TREF

8.1.3.4.15. NV-REQUIRE-COM-SPEC

Source model entity: NV-REQUIRE-COM-SPEC

Target model entity: NV-REQUIRE-COM-SPEC

Target model property	Source model property
VARIABLE-REF	NV-DATA-ELEMENT-REF
INIT-VALUE/CONSTANT-REFERENCE	INIT-VALUE-REF

8.1.3.4.16. NV-DATA-INTERFACE

Source model entity: NV-DATA-INTERFACE

Target model entity: NV-DATA-INTERFACE

Target model property	Source model property
NV-DATAS/VARIABLE-DATA-PROTOTYPE	NV-DATAS/DATA-ELEMENT-PROTOTYPE



Target model property	Source model property
IS-SERVICE	IS-SERVICE
SERVICE-KIND	No source model parameter is available.

8.1.3.4.17. COMPU-SCALE

Source model entity: COMPU-SCALE

Target model entity: COMPU-SCALE

Target model property	Source model property
LOWER-LIMIT	LOWER-LIMIT
UPPER-LIMIT	UPPER-LIMIT
COMPU-CONST	COMPU-CONST
COMPU-INVERSE-VALUE	COMPU-INVERSE-VALUE
COMPU-RATIONAL-COEFFS/COMPU-NUMERATOR	COMPU-RATIONAL-COEFFS/COMPU-NUMERATOR
COMPU-RATIONAL-COEFFS/COMPU-DENOMINATOR	COMPU-RATIONAL-COEFFS/COMPU-DENOMINATOR
MASK	No source model parameter is available.
SHORT-LABEL	SHORT-LABEL
SYMBOL	SYMBOL

8.1.3.4.18. NV-BLOCK-NEEDS

Source model entities: BLOCK-NEEDS, SWC-NV-BLOCK-NEEDS

Target model entity: NV-BLOCK-NEEDS

Target model property	Source model property
N-DATA-SETS	NV-BLOCK-NEEDS/N-DATA-SETS
READONLY	NV-BLOCK-NEEDS/READONLY
RESISTANT-TO-CHANGED-SW	NV-BLOCK-NEEDS/RESISTANT-TO-CHANGED-SW
RESTORE-AT-START	NV-BLOCK-NEEDS/RESTORE-AT-START
STORE-AT-SHUTDOWN	NV-BLOCK-NEEDS/STORE-AT-SHUTDOWN
WRITE-ONLY-ONCE	NV-BLOCK-NEEDS/WRITE-ONLY-ONCE
WRITING-FREQUENCY	NV-BLOCK-NEEDS/WRITING-FREQUENCY



Target model property	Source model property
WRITING-PRIORITY	NV-BLOCK-NEEDS/WRITING-PRIORITY

8.1.3.4.18.1. No upgrade of source model SWC-NV-BLOCK-NEEDS/CALL-BACK-PORTS, SWC-NV-BLOCK-NEEDS/NV-DATA-PORT-PROTOTYPE-REFS, SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF, SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF, SWC-NV-BLOCK-NEEDS/SERVICE-CALL-PORTS

The source model parameters SWC-NV-BLOCK-NEEDS/CALL-BACK-PORTS, SWC-NV-BLOCK-NEEDS/NV-DATA-PORT-PROTOTYPE-REFS, SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF, SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF, SWC-NV-BLOCK-NEEDS/SERVICE-CALL-PORTS are not upgraded into the target model because there are no corresponding parameters.

8.1.3.4.18.2. No upgrade of source model NV-BLOCK-NEEDS/RAM-BLOCK-STATUS-CONTROL

The source model parameter NV-BLOCK-NEEDS/RAM-BLOCK-STATUS-CONTROL is not upgraded into the target model because there is no corresponding parameter.

8.1.3.4.18.3. No configuration of target model parameters CALC-RAM-BLOCK-CRC, CHECK-STATIC-BLOCK-ID, N-ROM-BLOCKS, WRITE-VERIFICATION

The target model parameters CALC-RAM-BLOCK-CRC, CHECK-STATIC-BLOCK-ID, N-ROM-BLOCKS, WRITE-VERIFICATION are not configured during model upgrade because there is no source model counterpart.

8.1.3.4.18.4. No configuration of target model parameter RELIABILITY

The source model parameter RELIABILITY is not upgraded to the target model parameter RELIABILITY because the parameter values are not related.

8.1.3.4.18.5. Additional configuration of ASSIGNED-DATAS/ROLE-BASED-DATA-ASSIGNMENT for SWC-NV-BLOCK-NEEDS elements

For every SWC-NV-BLOCK-NEEDS element in the source model which either contains a valid DEFAULT-BLOCK_REF or a valid MIRROR-BLOCK-REF, one ASSIGNED-DATAS/ROLE-BASED-DATA-ASSIGNMENT element is created in the target model. The ROLE-BASED-DATA-ASSIGNMENT element is aggregated within the target model SWC-SERVICE-DEPENDENCY element which also aggregates the target model SWC-



NV-BLOCK-NEEDS **element**. ROLE-BASED-DATA-ASSIGNMENT/ROLE is set to ramMirror. ROLE-BASED-DATA-ASSIGNMENT/USED-PIM-REF references the target model counterpart of the source model PER-INSTANCE-MEMORY which is referenced by SWC-NV-BLOCK-NEEDS/MIRROR-BLOCK-REF. ROLE-BASED-DATA-ASSIGNMENT/USED-PARAMETER-ELEMENT aggregates a AUTOSAR-PARAMETER-REF element which in turn references in LOCAL-PARAMETER-REF the target model counterpart of the source model CALPRM-ELEMENT-PROTOTYPE which is referenced by SWC-NV-BLOCK-NEEDS/DEFAULT-BLOCK-REF.

8.1.3.4.19. END-TO-END-PROTECTION

Source model entity: END-TO-END-PROTECTION

Target model entity: END-TO-END-PROTECTION

Target model property	Source model property
END-TO-END-PROFILE/COUNTER-OFFSET	END-TO-END-PROFILE/COUNTER-OFFSET
END-TO-END-PROFILE/CRC-OFFSET	END-TO-END-PROFILE/CRC-OFFSET
END-TO-END-PROFILE/DATA-ID-MODE	END-TO-END-PROFILE/DATA-ID-MODE
END-TO-END-PROFILE/DATA-IDS/DATA-ID	END-TO-END-PROFILE/DATA-IDS/DATA-ID
END-TO-END-PROFILE/DATA-LENGTH	END-TO-END-PROFILE/DATA-LENGTH
END-TO-END-PROFILE/MAX-DELTA-COUNTER-INIT	END-TO-END-PROFILE/MAX-DELTA-COUNTER-INIT
No target model parameter is available.	END-TO-END-PROFILE/MAX-NO-NEW-OR-REPEATED-DATA
No target model parameter is available.	END-TO-END-PROFILE/SYNC-COUNTER-INIT
END-TO-END-PROTECTION-I-SIGNAL-I-PDUS/ END-TO-END-PROTECTION-I-SIGNAL-I-PDU	END-TO-END-PROTECTION-I-SIGNAL-I-PDUS/ END-TO-END-PROTECTION-I-SIGNAL-I-PDU
END-TO-END-PROTECTION-VARIABLE-PROTOTYPES/END-TO-END-PROTECTION-VARIABLE-PROTOTYPE	END-TO-END-PROTECTION-DATA-ELEMENT-PROTOTYPES/END-TO-END-PROTECTION-DATA-ELEMENT-PROTOTYPE

8.1.3.5. BSW Module Description

8.1.3.5.1. BSW-MODULE-DESCRIPTION

Source model entities: BSW-MODULE-DESCRIPTION

Target model entity: BSW-MODULE-DESCRIPTION



Target model property	Source model property
BSW-MODULE-DEPENDENCIES/BSW-MODULE-DEPENDENCY	BSW-MODULE-DEPENDENCIES/BSW-MODULE-DEPENDENCY
MODULE-ID	MODULE-ID
OUTGOING-CALLBACKS/BSW-MODULE-ENTRY-TRY-REF-CONDITIONAL	OUTGOING-CALLBACK-REFS/OUTGOING-CALLBACK-REF
PROVIDED-ENTRIES/BSW-MODULE-ENTRY-REF-CONDITIONAL	PROVIDED-ENTRY-REFS/PROVIDED-ENTRY-REF
BSW-MODULE-DOCUMENTATIONS/SW-COMPONENT-DOCUMENTATION	No source model parameter is available.
PROVIDED-MODE-GROUPS/MODE-DECLARATION-GROUP-PROTOTYPE	No source model parameter is available.
RELEASED-TRIGGERS/TRIGGER	No source model parameter is available.
REQUIRED-MODE-GROUPS/MODE-DECLARATION-GROUP-PROTOTYPE	No source model parameter is available.
REQUIRED-TRIGGERS/TRIGGER	No source model parameter is available.

8.1.3.5.1.1. New parameters in the target model

INTERNAL-BEHAVIORS/BSW-INTERNAL-BEHAVIOR

The parameter INTERNAL-BEHAVIORS/BSW-INTERNAL-BEHAVIOR contains the BSW-INTERNAL-BEHAVIOR which corresponds to the source model BSW-BEHAVIOR which references the BSW-MODULE-DESCRIPTION via MODULE-REF.

8.1.3.5.2. BSW-MODULE-ENTRY

Source model entities: BSW-MODULE-ENTRY

Target model entity: BSW-MODULE-ENTRY

Target model property	Source model property
CALL-TYPE	CALL-TYPE
IS-SYNCHRONOUS	IS-SYNCHRONOUS
SERVICE-ID	SERVICE-ID
ARGUMENTS/SW-SERVICE-ARG	SW-SERVICE-ARGS/SW-SERVICE-ARG
RETURN-TYPE	SW-SERVICE-RETURN
ROLE	No source model parameter is available.



8.1.3.5.2.1. Configuration of target model parameter EXECUTION-CONTEXT

The target model parameter `EXECUTION-CONTEXT` is configured using the source model parameter `EXE-CONTEXT` according to the following table:

Target model value	Source model value
TASK	TASK
INTERRUPT-CAT-1	INTERRUPT
UNSPECIFIED	UNSPECIFIED

8.1.3.5.2.2. Configuration of target model parameter IS-REENTRANT

The target model parameter `IS-REENTRANT` is configured using the source model parameter `SW-SERVICE-PROPS/SW-SERVICE-REENTRANCE`. If the source model parameter is available and has its value set to `REENTRANCE`, `IS-REENTRANT` is set to `true`, in all other cases to `false`.

8.1.3.5.2.3. Configuration of target model parameter SW-SERVICE-IMPL-POLICY

The target model parameter `SW-SERVICE-IMPL-POLICY` is configured using the source model parameter `SW-SERVICE-PROPS/SW-IMPL-POLICY`. If the source model parameter is not available, the target value is set to `STANDARD`.

8.1.3.5.3. SW-SERVICE-ARG

Source model entities: `SW-SERVICE-ARG`

Target model entity: `SW-SERVICE-ARG`

Target model property	Source model property
<code>SW-ARRAYSIZE</code>	<code>SW-ARRAYSIZE</code>
<code>SW-DATA-DEF-PROPS</code>	<code>SW-DATA-DEF-PROPS</code>
<code>DIRECTION</code>	No source model parameter is available.

8.1.3.5.4. RETURN-TYPE

Source model entities: `SW-SERVICE-RETURN`

Target model entity: `RETURN-TYPE`



The RETURN-TYPE is upgraded to SW-SERVICE-RETURN in the same way as described in [Section 8.1.3.5.3, "SW-SERVICE-ARG"](#).

8.1.3.5.5. BSW-MODULE-DEPENDENCY

Source model entities: BSW-MODULE-DEPENDENCY

Target model entity: BSW-MODULE-DEPENDENCY

Target model property	Source model property
EXPECTED-CALLBACKS/BSW-MODULE-EN-TRY-REF-CONDITIONAL	EXPECTED-CALLBACK-REFS/EXPECTED-CALLBACK-REF
REQUIRED-ENTRYS/BSW-MODULE-ENTRY-REF-CONDITIONAL	REQUIRED-ENTRY-REFS/REQUIRED-ENTRY-REF
TARGET-MODULE-ID	TARGET-MODULE-ID

8.1.3.5.5.1. Configuration of SERVICE-NEEDS parameters in BSW-MODULE-DEPENDENCY/SERVICE-ITEMS

The target model SERVICE-NEEDS elements in BSW-MODULE-DEPENDENCY/SERVICE-ITEMS are configured using their source model counterparts in BSW-MODULE-DEPENDENCY/SERVICE-ITEMS.

8.1.3.5.6. BSW-INTERNAL-BEHAVIOR

Source model entities: BSW-BEHAVIOR

Target model entity: BSW-INTERNAL-BEHAVIOR

Target model property	Source model property
ENTITYS/BSW-INTERRUPT-ENTITY	ENTITYS/BSW-INTERRUPT-ENTITY
ENTITYS/BSW-CALLED-ENTITY	ENTITYS/BSW-MODULE-ENTITY
ENTITYS/BSW-SCHEDULABLE-ENTITY	ENTITYS/BSW-SCHEDULABLE-ENTITY
EVENTS/BSW-TIMING-EVENT	EVENTS/BSW-CYCLIC-EVENT
EVENTS/BSW-INTERNAL-TRIGGER-OC-CURRED-EVENT	EVENTS/BSW-SPORADIC-EVENT
EXCLUSIVE-AREAS/EXCLUSIVE-AREA	EXCLUSIVE-AREAS/EXCLUSIVE-AREA
CONSTANT-MEMORYS/PARAMETER-DATA-PROTO-TYPE	No source model parameter is available.



Target model property	Source model property
CONSTANT-VALUE-MAPPING-REFS/CONSTANT-VALUE-MAPPING-REF	No source model parameter is available.
DATA-TYPE-MAPPING-REFS/DATA-TYPE-MAPPING-REF	No source model parameter is available.
INTERNAL-TRIGGERING-POINTS/BSW-INTERNAL-TRIGGERING-POINT	No source model parameter is available.
MODE-RECEIVER-POLICYS/BSW-MODE-RECEIVER-POLICY	No source model parameter is available.
MODE-SENDER-POLICYS/BSW-MODE-SENDER-POLICY	No source model parameter is available.
PER-INSTANCE-PARAMETERS/PARAMETER-DATA-PROTOTYPE	No source model parameter is available.
SCHEDULER-NAME-PREFIXS/BSW-SCHEDULER-NAME-PREFIX	No source model parameter is available.
SERVICE-DEPENDENCYS/BSW-SERVICE-DEPENDENCY	No source model parameter is available.
STATIC-MEMORYS/VARIABLE-DATA-PROTOTYPE	No source model parameter is available.
TRIGGER-DIRECT-IMPLEMENTATIONS/BSW-TRIGGER-DIRECT-IMPLEMENTATION	No source model parameter is available.

8.1.3.5.6.1. No configuration of `BSW-BACKGROUND-EVENT`, `BSW-EXTERNAL-TRIGGER-OCCURRED-EVENT`, `BSW-MODE-SWITCH-EVENT`, `BSW-MODE-SWITCHED-ACK-EVENT` in target model

The target model parameters `EVENTS/BSW-BACKGROUND-EVENT`, `EVENTS/BSW-EXTERNAL-TRIGGER-OCCURRED-EVENT`, `EVENTS/BSW-MODE-SWITCH-EVENT`, and `EVENTS/BSW-MODE-SWITCHED-ACK-EVENT` are not configured since there are no source model counterparts.

8.1.3.5.6.2. Configuration of source model parameter `MODULE-REF`

The target model counterpart of the source model `BSW-MODULE-DESCRIPTION` referenced via `MODULE-REF` contains the upgraded `BSW-INTERNAL-BEHAVIOR` in the target model.

8.1.3.5.7. BSW-INTERRUPT-ENTITY

Source model entities: `BSW-INTERRUPT-ENTITY`



Target model entity: BSW-INTERRUPT-ENTITY

Target model property	Source model property
ACTIVATION-POINTS/BSW-INTERNAL-TRIGGERING-POINT-REF-CONDITIONAL/BSW-INTERNAL-TRIGGERING-POINT-REF	ACTIVATION-POINT-REFS/ACTIVATION-POINT-REF
CALLED-ENTRYS/BSW-MODULE-ENTRY-REF-CONDITIONAL/BSW-MODULE-ENTRY-REF	CALLED-ENTRY-REFS/CALLED-ENTRY-REF
CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF	CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF
No target model parameter is available.	CANCELLATION-POINT-REFS/CANCELLATION-POINT-REF
IMPLEMENTED-ENTRY-REF	IMPLEMENTED-ENTRY-REF
INTERRUPT-CATEGORY	INTERRUPT-CATEGORY
INTERRUPT-SOURCE	INTERRUPT-SOURCE
ACCESSED-MODE-GROUPS/MODE-DECLARATION-GROUP-PROTOTYPE-REF-CONDITIONAL	No source model parameter is available.
ISSUED-TRIGGERS/TRIGGER-REF-CONDITIONAL	No source model parameter is available.
MANAGED-MODE-GROUPS/MODE-DECLARATION-GROUP-PROTOTYPE-REF-CONDITIONAL	No source model parameter is available.
MINIMUM-START-INTERVAL	No source model parameter is available, set to 0.
RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF	No source model parameter is available.
SCHEDULER-NAME-PREFIX-REF	No source model parameter is available.
SW-ADDR-METHOD-REF	No source model parameter is available.

8.1.3.5.8. BSW-SCHEDULABLE-ENTITY

Source model entities: BSW-SCHEDULABLE-ENTITY

Target model entity: BSW-SCHEDULABLE-ENTITY

Target model property	Source model property
ACTIVATION-POINTS/BSW-INTERNAL-TRIGGERING-POINT-REF-CONDITIONAL/BSW-INTERNAL-TRIGGERING-POINT-REF	ACTIVATION-POINT-REFS/ACTIVATION-POINT-REF



Target model property	Source model property
CALLED-ENTRYS/BSW-MODULE-ENTRY-REF-CONDITIONAL/BSW-MODULE-ENTRY-REF	CALLED-ENTRY-REFS/CALLED-ENTRY-REF
CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF	CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF
No target model parameter is available.	CANCELLATION-POINT-REFS/CANCELLATION-POINT-REF
IMPLEMENTED-ENTRY-REF	IMPLEMENTED-ENTRY-REF
ACCESSED-MODE-GROUPS/MODE-DECLARATION-GROUP-PROTOTYPE-REF-CONDITIONAL	No source model parameter is available.
ISSUED-TRIGGERS/TRIGGER-REF-CONDITIONAL	No source model parameter is available.
MANAGED-MODE-GROUPS/MODE-DECLARATION-GROUP-PROTOTYPE-REF-CONDITIONAL	No source model parameter is available.
MINIMUM-START-INTERVAL	No source model parameter is available, set to 0.
RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF	No source model parameter is available.
SCHEDULER-NAME-PREFIX-REF	No source model parameter is available.
SW-ADDR-METHOD-REF	No source model parameter is available.

8.1.3.5.9. BSW-CALLED-ENTITY

Source model entities: BSW-MODULE-ENTITY

Target model entity: BSW-CALLED-ENTITY

Target model property	Source model property
ACTIVATION-POINTS/BSW-INTERNAL-TRIGGERING-POINT-REF-CONDITIONAL/BSW-INTERNAL-TRIGGERING-POINT-REF	ACTIVATION-POINT-REFS/ACTIVATION-POINT-REF
CALLED-ENTRYS/BSW-MODULE-ENTRY-REF-CONDITIONAL/BSW-MODULE-ENTRY-REF	CALLED-ENTRY-REFS/CALLED-ENTRY-REF
CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF	CAN-ENTER-EXCLUSIVE-AREA-REFS/CAN-ENTER-EXCLUSIVE-AREA-REF
No target model parameter is available.	CANCELLATION-POINT-REFS/CANCELLATION-POINT-REF



Target model property	Source model property
IMPLEMENTED-ENTRY-REF	IMPLEMENTED-ENTRY-REF
ACCESED-MODE-GROUPS/MODE-DE-CLARATION-GROUP-PROTOTYPE-REF-CONDITIONAL	No source model parameter is available.
ISSUED-TRIGGERS/TRIGGER-REF-CONDITIONAL	No source model parameter is available.
MANAGED-MODE-GROUPS/MODE-DE-CLARATION-GROUP-PROTOTYPE-REF-CONDITIONAL	No source model parameter is available.
MINIMUM-START-INTERVAL	No source model parameter is available, set to 0.
RUNS-INSIDE-EXCLUSIVE-AREA-REFS/RUNS-INSIDE-EXCLUSIVE-AREA-REF	No source model parameter is available.
SCHEDULER-NAME-PREFIX-REF	No source model parameter is available.
SW-ADDR-METHOD-REF	No source model parameter is available.

8.1.3.5.10. BSW-TIMING-EVENT

Source model entities: BSW-CYCLIC-EVENT

Target model entity: BSW-TIMING-EVENT

Target model property	Source model property
PERIOD	CYCLE-TIME
STARTS-ON-EVENT-REF	STARTS-ON-EVENT-REF
DISABLED-IN-MODE-IREFS/DISABLED-IN-MODE-IREF	No source model parameter is available.

8.1.3.5.11. BSW-INTERNAL-TRIGGER-OCCURRED-EVENT

Source model entities: BSW-SPORADIC-EVENT

Target model entity: BSW-INTERNAL-TRIGGER-OCCURRED-EVENT

Target model property	Source model property
No target model parameter is available.	DELAY-TIME
STARTS-ON-EVENT-REF	STARTS-ON-EVENT-REF



Target model property	Source model property
DISABLED-IN-MODE-IREFS/DISABLED-IN-MODE-IREF	No source model parameter is available.

8.1.3.5.11.1. Creation of target model element BSW-INTERNAL-BEHAVIOR/INTERNAL-TRIGGERING-POINTS/BSW-INTERNAL-TRIGGERING-POINT

For every source model element BSW-BEHAVIOR/EVENTS/BSW-SPORADIC-EVENT one additional target model element BSW-INTERNAL-BEHAVIOR/INTERNAL-TRIGGERING-POINTS/BSW-INTERNAL-TRIGGERING-POINT is created, its SHORT-NAME is set to ITP_<BswEventName>, where <BswEventName> is the SHORT-NAME of the source model BSW-SPORADIC-EVENT. The target model BSW-SPORADIC-EVENT references the BSW-INTERNAL-TRIGGERING-POINT via EVENT-SOURCE-REF.

8.1.3.5.12. BSW-IMPLEMENTATION

Source model entity: BSW-IMPLEMENTATION

Target model entity: BSW-IMPLEMENTATION

Target model property	Source model property
AR-RELEASE-VERSION	AR-MAJOR-VERSION.AR-MINOR-VERSION.AR-PATCH-VERSION
CODE-DESCRIPTORS/CODE	CODE-DESCRIPTORS/CODE
USED-CODE-GENERATOR	CODE-GENERATOR
DEBUG-INFOS/BSW-DEBUG-INFO	No source model parameter is available.
GENERATED-ARTIFACTS/DEPENDENCY-ON-ARTIFACT	No source model parameter is available.
MC-SUPPORT	No source model parameter is available.
PROGRAMMING-LANGUAGE	PROGRAMMING-LANGUAGE
REQUIRED-ARTIFACTS/DEPENDENCY-ON-ARTIFACT	No source model parameter is available.
REQUIRED-GENERATOR-TOOLS/DEPENDENCY-ON-ARTIFACT	No source model parameter is available.
SW-VERSION	SW-MAJOR-VERSION.SW-MINOR-VERSION.SW-PATCH-VERSION
USED-CODE-GENERATOR	No source model parameter is available.



Target model property	Source model property
VENDOR-ID	VENDOR-ID
BEHAVIOR-REF	BEHAVIOR-REF
VENDOR-API-INFIX	VENDOR-API-INFIX

8.1.3.5.12.1. No upgrade of source model parameters COMPILERS/COMPILER, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-FILE, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-LIBRARY, LINKERS/LINKER, PROCESSOR-REFS/PROCESSOR-REF, RESOURCE-CONSUMPTION, PRECONFIGURED-CONFIGURATION-REF, RECOMMENDED-CONFIGURATION-REF, REQUIRED-HW-REFS/REQUIRED-HW-REF, VENDOR-SPECIFIC-MODULE-DEF-REF

The source model parameters COMPILERS/COMPILER, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-FILE, IMPLEMENTATION-DEPENDENCIES/DEPENDENCY-ON-LIBRARY, LINKERS/LINKER, PROCESSOR-REFS/PROCESSOR-REF, RESOURCE-CONSUMPTION, PRECONFIGURED-CONFIGURATION-REF, RECOMMENDED-CONFIGURATION-REF, REQUIRED-HW-REFS/REQUIRED-HW-REF, VENDOR-SPECIFIC-MODULE-DEF-REF are not upgraded into the target model.

8.1.3.5.12.2. Creation of target model RESOURCE-CONSUMPTION

For each SWC-IMPLEMENTATION in the target model, one RESOURCE-CONSUMPTION sub element is set up. Its SHORT-NAME is the SHORT-NAME of the SWC-IMPLEMENTATION plus the suffix `_resourceConsumption`.

8.1.3.5.12.3. Configuration of target model SWC-BSW-MAPPING-REF

If a SWC-BSW-MAPPING to which the BSW-IMPLEMENTATION is related, see [Section 8.1.3.5.13, "SWC-BSW-MAPPING"](#), then this SWC-BSW-MAPPING is referenced in SWC-BSW-MAPPING-REF.

8.1.3.5.12.4. Upgrade of source model parameter PROGRAMMING-LANGUAGE

If the source model parameter PROGRAMMING-LANGUAGE is not defined, C is configured as default value in the target model.

8.1.3.5.12.5. Upgrade of source model parameter VENDOR-ID

If the source model parameter VENDOR-ID is not defined, 1 is configured as default value in the target model.



8.1.3.5.13. SWC-BSW-MAPPING

If the source model contains an INTERNAL-BEHAVIOR aggregating at least one RUNNABLE-ENTITY which references a BSW-MODULE-ENTITY/BSW-SCHEDULABLE-ENTITY/BSW-INTERRUPT-ENTITY via BSW-ENTITY-REF, a SWC-BSW-MAPPING element is created in the target model.

Its SHORT-NAME is SwcBswMapping_<SWCBhShortName>_<BSWBhShortName>. <SWCBhShortName> is the SHORT-NAME of the INTERNAL-BEHAVIOR aggregating the source model RUNNABLE-ENTITY and <BSWBhShortName> is the SHORT-NAME of the BSW-BEHAVIOR aggregating the source model BSW-MODULE-ENTITY/BSW-SCHEDULABLE-ENTITY/BSW-INTERRUPT-ENTITY.

BSW-BEHAVIOR-REF references the target model counterpart of the source model BSW-BEHAVIOR, SWC-BEHAVIOR-REF references the target model counterpart of the source model INTERNAL-BEHAVIOR.

For each RUNNABLE-ENTITY referencing a BSW-MODULE-ENTITY/BSW-SCHEDULABLE-ENTITY/BSW-INTERRUPT-ENTITY, one RUNNABLE-MAPPINGS/SWC-BSW-RUNNABLE-MAPPING parameter is set up. Its BSW-ENTITY-REF references the target model counterpart of the source model BSW-MODULE-ENTITY/BSW-SCHEDULABLE-ENTITY/BSW-INTERRUPT-ENTITY, SWC-RUNNABLE-REF references the target model counterpart of the source model RUNNABLE-ENTITY.

8.1.4. AUTOSAR 4.0 platform data types

During a model upgrade run, the AUTOSAR 4.0 *platform* datatypes are created according to the table below. Each line corresponds to one IMPLEMENTATION-DATA-TYPE element referencing one SW-BASE-TYPE element via SW-DATA-DEF-PROPS/W-DATA-DEF-PROPS-VARIANTS/SW-DATA-DEF-PROPS-CONDITIONAL/BASE-TYPE-REF and one DATA-CONSTR element via SW-DATA-DEF-PROPS/W-DATA-DEF-PROPS-VARIANTS/SW-DATA-DEF-PROPS-CONDITIONAL/DATA-CONSTR-REF. The IMPLEMENTATION-DATA-TYPE are created in the AR-PACKAGE path /AUTOSAR_Platform/Implementation-Datatypes, the SW-BASE-TYPE are situated in /AUTOSAR_Platform/BaseTypes, the DATA-CONSTR are created in /AUTOSAR_Platform/DataConstrs. If UPPER-LIMITS and/or LOWER-LIMITS are available, they are considered "closed" limits, i.e. the limit is included by the value range of the data type. SHORT-NAME refers to the SHORT-NAME parameters of the created IMPLEMENTATION-DATA-TYPE, SW-BASE-TYPE, and DATA-CONSTR elements. LOWER-LIMIT and UPPER-LIMIT refer to the respective DATA-CONSTR parameters. BASE-TYPE-ENCODING and BASE-TYPE-SIZE refer to the respective parameters in SW-BASE-TYPE.

For the IMPLEMENTATION-DATA-TYPE boolean, an IMPLEMENTATION-DATA-TYPE reference to uint8 is set up, therefore no SW-BASE-TYPE and DATA-CONSTR are required.

For the IMPLEMENTATION-DATA-TYPES float32 and float64, no dedicated limits are provided, therefore no dedicated DATA-CONSTR entries exist for them.

SHORT-NAME	LOWER-LIMIT	UPPER-LIMIT	BASE-TYPE-ENCODING	BASE-TYPE-SIZE
boolean	n/a	n/a	n/a	n/a



SHORT-NAME	LOWER-LIMIT	UPPER-LIMIT	BASE-TYPE-ENCODING	BASE-TYPE-SIZE
uint8	0	255	NONE	8
uint16	0	65535	NONE	16
uint32	0	4294967295	NONE	32
sint8	-128	127	2C	8
sint16	-32768	32767	2C	16
sint32	-2147483648	2147483647	2C	32
float32	n/a	n/a	IEEE754	32
float64	n/a	n/a	IEEE754	64

If the AUTOSAR 4.0 platform data types are upgraded into a 4.0.3 target model, every platform data type additionally has its **TYPE-EMITTER** parameter value set to BSW.

Bibliography

- [1] AUTOSAR *Layered Software Architecture*, Issue Version 2.1.0, Revision 0014, Publisher: www.autosar.org
- [2] OSI *Reference Model - The ISO Model of Architecture for Open Systems Interconnection*, Author: Zimmermann, Pages: 425 to 432,
- [3] AUTOSAR - *Specification of ECU Configuration*, Issue Document number 087. Document version 2.0.1, Revision 0002, Part of release 3.0, Publisher: www.autosar.org
- [4] AUTOSAR - *Specification of ECU Configuration*, Issue Document number 087. Document version 3.2.0, Revision 3, Part of release 4.0, Publisher: www.autosar.org
- [5] AUTOSAR - *Specification of the Virtual Functional Bus*, Publisher: www.autosar.org
- [6] AUTOSAR - *Specification of the Software Component Template*, Publisher: www.autosar.org
- [7] AUTOSAR *Model Persistence Rules for XML*, Issue Version 2.1.2, Release 3.0, Revision 0001, Publisher: www.autosar.org
- [8] LIN *Specification Package*, URL: <http://www.lin-subbus.org/>, Publisher: Lin Consortium



Appendix A. Third party license

This product includes third-party components that require the following notices. For respective license terms refer to the subfolder /licenses

- ▶ This product includes Java(TM) 2 Runtime Environment 1.8.0_121.

Java(TM) 2 Runtime Environment 1.8.0_121 is licensed under the Oracle Corporation Binary Code License Agreement. For additional Licenses Terms according to Section G of the Oracle Corporation Binary Code License Agreement please refer to "jre/THIRDPARTYLICENSEREADME.txt".

- ▶ The product is based in part on software developed for eclipse.org licensed under the EPL 1.0. (Eclipse runtime environment 4.6.2)

Source code for these components is available from <http://subclipse.tigris.org/> and www.eclipse.org, under the Eclipse Public License ("EPL", see <http://www.eclipse.org/org/documents/epl-v10.php>).

As a requirement of the EPL, Elektrobit hereby:

(1) disclaims on behalf of all Eclipse.org Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose;

(2) excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits; and

(3) states that any provisions which differ from the Eclipse Public License are offered by Elektrobit alone and not by any other party.

- ▶ The product is based in part on software developed for eclipse.org licensed under the EPL 1.0. (Eclipse Modeling Framework 2.8.0)

Source code for these components is available from <http://subclipse.tigris.org/> and www.eclipse.org, under the Eclipse Public License ("EPL", see <http://www.eclipse.org/org/documents/epl-v10.php>).

As a requirement of the EPL, Elektrobit hereby:

(1) disclaims on behalf of all Eclipse.org Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose;

(2) excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits; and

(3) states that any provisions which differ from the Eclipse Public License are offered by Elektrobit alone and not by any other party.



- ▶ This product includes jaxen 1.1-beta-7.

Copyright 2003-2006 The Werken Company. All Rights Reserved.

- ▶ This product includes commons-jxpath.Apache Commons.

JXPath Copyright 2001-2008 The Apache Software Foundation.

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

- ▶ This product includes commons-jxpath.Apache Commons.

JXPath Copyright 2001-2015 The Apache Software Foundation.

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

- ▶ This product includes java cup 0.10k.

Copyright 1996 by Scott Hudson, Frank Flannery, C. Scott Ananian

- ▶ This product includes log4j 1.2.7.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

Copyright (C) The Apache Software Foundation. All rights reserved.

- ▶ This product includes software developed by the JDOM Project (<http://www.jdom.org/>).

Copyright (C) 2000-2004 Jason Hunter & Brett McLaughlin. All rights reserved.

- ▶ This product includes poi 2.5.1.

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

- ▶ This product includes sqlite-jdbc 3.6.19.

sqlite-jdbc Copyright 2013 The Apache Software Foundation.

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

- ▶ This product includes Xerces-J 2.8.0.

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

Apache Xerces Java, Copyright 1999-2007 The Apache Software Foundation

Portions of this software were originally based on the following:

- software copyright (c) 1999, IBM Corporation., <http://www.ibm.com>.

- software copyright (c) 1999, Sun Microsystems., <http://www.sun.com>.

- voluntary contributions made by Paul Eng on behalf of the Apache Software Foundation that were originally developed at iClick, Inc., software copyright (c) 1999.

- ▶ This product includes Xerces 3.1.1

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

Portions of this software were originally based on the following:

- software copyright (c) 1999, IBM Corporation., <http://www.ibm.com>.

- ▶ This product includes trang version 20091111.

Copyright (c) 2002, 2003, 2008 Thai Open Source Software Center Ltd

- ▶ The product is based in part on software developed for eclipse.org licensed under the EPL 1.0. (subclipse 1.8.18)

Source code for these components is available from <http://subclipse.tigris.org/> and www.eclipse.org, under the Eclipse Public License ("EPL", see <http://www.eclipse.org/org/documents/epl-v10.php>).

As a requirement of the EPL, Elektrobit hereby:

(1) disclaims on behalf of all Eclipse.org Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose;

(2) excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits; and

(3) states that any provisions which differ from the Eclipse Public License are offered by Elektrobit alone and not by any other party.

- ▶ The product is based in part on software developed for eclipse.org licensed under the EPL 1.0. (collabnet merge 2.1)

Source code for these components is available from <http://subclipse.tigris.org/> and www.eclipse.org, under the Eclipse Public License ("EPL", see <http://www.eclipse.org/org/documents/epl-v10.php>).

As a requirement of the EPL, Elektrobit hereby:

(1) disclaims on behalf of all Eclipse.org Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose;

(2) excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits; and

(3) states that any provisions which differ from the Eclipse Public License are offered by Elektrobit alone and not by any other party.

- ▶ This product includes javaHL 1.7.8.



This product includes software developed by CollabNet (<http://www.Collab.Net/>). Copyright (c) 2000-2007 CollabNet. All rights reserved.

- ▶ This product includes software developed by Computing Services at Carnegie Mellon University (<http://www.cmu.edu/computing/>).
- ▶ This product includes jsch 0.1.53.

Copyright (c) 2002-2012 Atsuhiko Yamanaka, JCraft, Inc.

- ▶ This product includes antlr 3.5.2.

Copyright (c) 2013 Terence Parr All rights reserved.

- ▶ This product includes antlr 4.7.

Copyright (c) 2012 Terence Parr and Sam Harwell. All rights reserved.

- ▶ This product includes ICU4J 56.1.0

Copyright (c) 1995-2011 International Business Machines Corporation and others

ICU4J contains further third party products. icu_56.1.0_license.txt contains all additional licenses.

- ▶ This product includes Google Guava 20.0

This product includes software developed by Google Inc. (<http://www.google.com/>).

Portions of this software were originally based on the following:

- software copyright (c) 2010-2016, Google Inc., <http://www.google.com>.

- ▶ This product includes Apache Ant.

Apache Ant Copyright 1999-2017 The Apache Software Foundation

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

- ▶ This product includes Apache Batik.

Apache Batik Copyright 1999-2017 The Apache Software Foundation

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

Batik includes the following software:

DOM

DOM is developed by the World Wide Web Consortium. Your use of DOM is subject to the terms and conditions of the license found in the file LICENSE.dom-software.txt which is included with this product and can also be found at <http://www.w3.org/Consortium/Legal/copyright-software-19980720>.



SAX

SAX is developed by the SAX project (<http://www.saxproject.org>). Your use of SAX is subject to the terms and conditions of the license found in the file LICENSE.sax.txt which is included with this product.

- ▶ This product includes Javax EL API.

Copyright © 1996-2015, Oracle and/or its affiliates. All Rights Reserved. Use is subject to license terms.

This product includes software developed by Sun/Oracle as part of the Glassfish project (<https://javaee.github.io/glassfish/>).

- ▶ This product includes javax.annotation.

Copyright © 1996-2015, Oracle and/or its affiliates. All Rights Reserved. Use is subject to license terms.

This product includes software developed by Sun/Oracle as part of the Glassfish project (<https://javaee.github.io/glassfish/>).

- ▶ This product includes the package "javax.inject" from the atinject project

Copyright © 1996-2015, Oracle and/or its affiliates. All Rights Reserved. Use is subject to license terms.

This product includes software developed by Sun/Oracle as part of the Glassfish project (<https://javaee.github.io/glassfish/>).

- ▶ This product includes the package "xml-apis"

Copyright 1999-2017 The Apache Software Foundation

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

- ▶ This product includes Javax Servlet API.

Copyright © 1996-2015, Oracle and/or its affiliates. All Rights Reserved. Use is subject to license terms.

This product includes software developed by Sun/Oracle as part of the Glassfish project (<https://javaee.github.io/glassfish/>).

- ▶ This product includes Javax JSP API.

Copyright © 1996-2015, Oracle and/or its affiliates. All Rights Reserved. Use is subject to license terms.

This product includes software developed by Sun/Oracle as part of the Glassfish project (<https://javaee.github.io/glassfish/>).

- ▶ This product includes Java Server Pages Engine.

Copyright © 1996-2015, Oracle and/or its affiliates. All Rights Reserved. Use is subject to license terms.



This product includes software developed by Sun/Oracle as part of the Glassfish project (<https://javaee.github.io/glassfish/>).

- ▶ This product includes Apache Lucene.

Copyright 1999-2017 The Apache Software Foundation.

This product includes software developed by The Apache Software Foundation (<http://www.apache.org/>).

- ▶ This product includes SAT4J.

SAT4J includes content that was obtained under licenses that differ from the SAT4J licenses.

is based on code obtained from the Minisat 1.1.4 implementation, the source code for which can be found at www.minisat.se, under a permissive license.

MiniSat -- Copyright (c) 2003-2005, Niklas Een, Niklas Sorensson.

- ▶ This product includes CSS SAC 1.3 Java Binding developed by the World Wide Web Consortium ("W3C").

This product includes Batik SVG Toolkit 1.6 (subset) developed by the Apache Software Foundation.

This product includes SAX developed by the SAX project.

Index

A

about EB tresos Studio, 83
 autoconfiguration wizards (see unattended wizards)
 AUTOSAR format
 module configuration, 58
 system description, 59

B

building
 projects, 334
 Bundle Registry, 149

C

collaboration (see team collaboration)
 command line interface, 416
 command shell, 416
 compare editor, 253
 configuration
 configuration class
 enabling/disabling parameters, 133
 setting in your project, 175
 verifying the, 329
 what is this?, 56
 configuration time
 dependency to code generators, 331
 enabling/disabling parameters, 133
 verifying the, 329
 what is this?, 56
 configuration variant
 setting the, 295
 what is this?, 56
 configuration and generation tool
 EB tresos Studio, 50
 configuration data
 export, 351
 import, 351
 content type, 57, 351
 CVS
 version control system

working with CVS in EB tresos Studio, 224

D

datamodel, 253
 DBC format, 61
 DBC Importer, 378
 attaching attributes, 378
 attributes, 378
 I-Pdu Multiplexer, 387
 messages, 378
 multiplexed messages, 378
 NM messages, 377
 signals, 378
 DBC parser
 grammar, 384, 384
 differ, 253
 displayed entity
 filtering, 524
 dongled licenses, 163

E

EB tresos Details dialog, 146
 EB tresos Studio verifier, 111
 ECU Configuration
 importing, 407
 ECU Configuration Editor, 292
 Bulk Change dialog, 305
 opening the, 294
 ECU Extract
 extract, 411
 importing, 362
 editing parameter values
 unattended wizards, 308
 Editor view, 98, 103, 293
 Bulk Change dialog, 305
 overview, 96
 page name, 96
 Search dialog, 322
 tables
 changing rows simultaneously, 305
 tabs, 96
 unattended wizards, 308

- configuring the, 309
- Element Description view**, 106
- Element Information view**
 - parameter meta information, 104
- Element Outline view**, 293
- entity tree table**
 - icons, 523
- Entity tree table**
 - displaying, 524
- Error Log view**, 106, 293
 - filtering messages, 109
 - log entries, 108
- export**
 - AUTOSAR, 351
 - system descriptions (Sys-D), 371
- extending**
 - EB tresos Studio
 - with your own features, 169
- F**
 - features (see extending)
 - Fibex format, 62
 - Fibex Importer, 396
 - file type, 57
 - FlexLM licenses, 163
 - floating licenses, 163
 - floating network licenses, 163
 - functional safety
 - suitability, 48
- G**
 - generating code
 - of your project, 329
 - generating source code
 - from projects, 331
 - graphical user interface, 67
 - GUI, 67
 - configuring a module, 294
 - ECU Configuration Editor, 294
 - Editor view
 - comments, 299
 - Element Description view
- H**
 - parameter, 106
- Element Information view**
 - parameter meta information, 104
- Generic Configuration Editor**, 131
 - choices, 138
 - optional elements, 143
 - references, 136
 - tables, 139
- I**
 - import
 - AUTOSAR, 351
 - importing system descriptions (Sys-D)
 - adding a System Description Importer, 363
- L**
 - LDF files, 391
 - LDF format, 61, 392
 - LDF Importer, 391, 392
 - license files, 163
 - license path, 163
 - license server, 163
 - license sources, 163
 - licenses, 163
 - LIN configuration, 391
 - linking
 - external folders, 169
 - Loadable, 483, 484
 - log files, 54
- M**
 - Module Registry, 148
- N**
 - network licenses, 163
 - Node Outline view, 293
 - appearance of displayed items, 102
 - copy name, 100
 - Delete variant, 100

Edit variant conditions, 100
 editor contents, 98
 expand collapse, 99
 filtering displayed items, 101
 show in editor, 100

creation of child elements, 204
 generation path, 203
 module configurations, 205
 system, 222
 Project Wizard, 170
 proprietary format, 384

O

OIL files, 374
 OIL format, 374
 OIL Importer, 374
 Os configurations, 374, 374
 Os module, 374, 374

P

parameter, 106
 (see also Element Description view)
 parameter meta information, 104
 (see also Element Information view)
 path mapping
 package structure, 355
 Plugin Registry, 149
 PostBuild, 483, 484, 484
 PostBuild variation point, 484
 PostBuildVariants, 484, 484
 Problems view, 111, 293
 filtering messages, 114
 project
 building a, 334
 convert to newer version, 289
 delete, 196
 export, 342
 generate code, 329
 generating source code, 331
 import, 337
 verifying, 329
 project configuration data
 exporting, 346
 importing, 346
 Project Explorer
 hiding layers, 88
 Project Explorer view, 194, 195, 195, 196, 293
 project properties, 200

R

Results view
 filtering messages, 130
 Reusing importers and exporters, 406

S

safe use of
 EB tresos Studio, 48
 Search
 dialog, 322
 searching for parameters
 Editor view, 322
 Selectable, 484, 484
 source code
 building from projects, 334
 generating from projects, 331
 storing projects as compressed archive, 344
 SVN
 version control system
 working with SVN in EB tresos Studio, 234
 synchronizing Project Explorer with editor, 94
 Sys-D format (see system description)
 system description
 exporting, 370
 adding a System Description Exporter, 371
 importing, 362
 system model
 viewing, 522
 System Model Viewer
 opening, 524

T

TDB Importer, 403
 team collaboration, 253
 using, 224

Third-party licenses, 791

tresosDB format, 60

U

unattended wizards, 308

Calculate Derivable Values

configuring the, 317

Calculate Handle IDs

configuring the, 318

configuring the, 308

Execute multiple tasks

configuring the, 319

Generate Code

CodeGen, 321

Switch PostBuildVariant

Selectable, 320

Unix line endings, 203

use of OIL Importer, 377

V

variant resolving, 480

vendor-specific module definition, 454, 460, 463

verifying

project, 329

version control system, 224

W

workspace, 54

X

XDM files, 253