ECSEL Joint Undertaking

European Commission

ECSEL Research and Innovation actions (RIA)

AMASS

Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems

AMASS Platform – Prototype P1
User Manual

Work Package:	WP2: Reference Architecture and Integration
Dissemination level:	CO = Confidential, only for members of the consortium
Status:	Draft
Date:	January 2, 2016
Responsible partner:	A. López (TECNALIA)
Contact information:	angel.lopez@tecnalia.com
Document reference:	AMASS_UserManual_WP2_V1.0

PROPRIETARY RIGHTS STATEMENT
This document contains information that is proprietary to the AMASS Consortium. Neither this document nor the information contained herein shall be used, duplicated or communicated by any means to any third party, in whole or in parts, except with prior written consent of the AMASS consortium.

Contributors

Names	Organisation
A. Ruiz, H. Espinoza, A. Lopez, G. Juez	TECNALIA Research & Innovation
I. Ayala, B. Gallina, M. A. Javed, F. UL Muram, I. Sljivo	MDH
S. Puri	INTECS
J. L. de la Vara, J. M. Álvarez, E. Parra	Universidad Carlos III de Madrid
L. M. Alonso, B. López	The REUSE Company
A. Debiasi, S. Tonetta	Fondazione Bruno Kessler

Reviewers

Names	Organisation

Document History

Version	Date	Status	Author (Partner)	Remarks
V0.1	2016-07-26	ToC and draft content	H. Espinoza (TEC)	
V0.2				
V0.3				
V0.4				
V0.5				
V0.6				
V1.0		Final version		

TABLE OF CONTENTS

1 Executive Summary 15

2 AMASS platform - basic concepts 16
2.1 Naming conventions in AMASS clients 16
2.1 Approach for organising OpenCert projects and models 17
2.1.1 Project-independent information 17
2.1.2 Project-specific information 17

3 Installation 20
3.1 Installation of AMASS platform client 20
3.1.1 Client bundle download 20
3.1.2 Client configuration 20
3.1.3 Deleting Repository contents 23
3.2 Installation of the EPF Composer 24

4 Process Modeling with EPF 26
4.1 Modeling of reusable process elements 27
4.2 Modeling of processes mandated by standards 33
4.3 Modeling of reusable process patterns 37

5 Standards Modeling 39
5.1 Create Reference Framework model 39
5.2 How to edit a Reference Framework model 42
5.2.1 Add concepts to the diagram 42
5.2.2 Add links between concepts 42
5.2.3 Edit properties 43
5.2.4 Create multi-diagrams from a Reference Framework model 43
5.2.5 Non graphical editor 45
5.3 Creating Equivalence Maps 46
5.3.1 Equivalence Map using the editor 46
5.3.2 Equivalence Map using a tailored functionality. 50
5.4 Creating Applicability Tables 54

6 Assurance Project Management 58
6.1 Create Assurance Project and Baseline 58
6.2 Create or update Project Baseline 61
6.3 Edit Project Baseline 64
6.4 Edit Compliance Maps 65
6.4.1 Compliance Map using the editor 65
6.4.2 Compliance Map using a tailored functionality. 70
6.4.3 Compliance Mapping Table 72
6.5 Cross-Domain reuse 76
6.6 Cross-Project reuse 80
6.6.1 Cross Project reuse View 84
6.7 EPF to OpenCert Transformation 86
6.7.1 EPF Export to XML files 87
6.7.2 Import EPF XML Files into OpenCert 91
6.8 Creation of Mapping Model 94
6.9 Map Group edition 96

6.9.1 Add a map group 96
6.9.2 Delete a map group 98
6.10 Map edition 99
6.10.1 Add a map. 99
6.10.2 Delete a map. 100
6.11 Compliance via automatic generation of process-based arguments 102
6.12 Import models into Assurance Projects 104

7 System Component Specification 107
7.1 Create a CHESS project, model and diagrams 107
7.2 Creating Requirements 108
7.3 Create FormalProperties 109
7.4 Edit Formal Property 109
7.5 Creating a Contract 111
7.6 Specify Assumption and Guarantee for a Contract 112
7.7 Associate a Contract to a Block/Component 112
7.7.1 Selection of weak contract for Block/Component instances 113
7.7.2 Contract Refinement 114
7.8 Specify Component Behaviour (nominal and faulty) 117
7.9 Managing Analysis Contexts 118
7.10 View Status of System Architecture 118
7.11 Setup of External V&V Tools 120
7.12 Perform Check of Contract Refinement 121
7.13 Perform Check of Component Implementation on Contracts 122
7.14 Perform Consistency Check of Formal Properties 123
7.15 Perform Model Checking on Component Behaviour 124
7.16 V&V Manager 124
7.17 Generate Contract-based Fault Tree 125
7.18 Generate Documentation 125
7.19 Managing links between architecture and assurance 127

8 Assurance Argumentation Management 132
8.1 Preferences 132
8.2 Creating and Saving a Diagram 132
8.2.1 Creating a New Diagram. 133
8.2.2 Creating a Diagram at the project creation time 135
8.2.3 Opening a Diagram 135
8.2.4 Saving a Diagram 136
8.3 Editing Functions 136
8.3.1 Editing a Diagram 136
8.3.1.1 Copying and Pasting an Element 140
8.3.1.2 Deleting a Node or a Link 140
8.3.2 Create multi-diagrams from an Argumentation model 140
8.3.3 Connecting an Argument Diagram to Artefacts 143
8.3.3.1 Load evidence models in a File based Diagram 145
8.3.3.2 Connecting a Data based Diagram to Artefacts 145
8.4 Patterns 146
8.4.1 Creating a new Pattern Diagram 146
8.4.2 Editing a Diagram Using a Pattern or a Module 147
8.4.2.1 Editing a Pattern Diagram 147
8.4.2.2 Adding Elements from Patterns to a Diagram (instantiating a Pattern) 149

8.4.2.3 Creating a New Module Diagram 149
8.4.2.4 Editing a Module Diagram 149
8.4.2.5 Adding Elements from Modules to a Diagram (instantiating a Module) 150
8.4.3 Vocabulary 151
8.4.3.1 Defining Vocabularies 151
8.4.3.2 Using Vocabularies in the Argument Editor 153
8.5 Generating Argument Fragments 154
8.6 Printing 159
8.7 Export an Argument Model 159

9 Evidence Management 161
9.1 Define Artefact Repository Preferences 161
9.2 Artefact Definition 165
9.2.1 Add an artefact definition 165
9.2.2 Delete Artefact Definition. 167
9.3 Artefact 168
9.3.1 Add an artefact 168
9.3.2 Delete an artefact 172
9.4 Artefact Resource 173
9.4.1 Add an artefact resource to an artefact 173
9.4.2 Delete an artefact resource. 176
9.5 Artefact Property Value 177
9.5.1 Add an artefact property value to an artefact 178
9.5.2 Delete an artefact property value 180
9.6 Artefact Assurance Asset Evaluation 181
9.6.1 Add an artefact assurance asset evaluation to an artefact 181
9.6.2 Delete an artefact assurance asset evaluation 183
9.7 Artefact Assurance Asset Events. 184
9.7.1 Add an artefact assurance asset event to an artefact 184
9.7.2 Delete an artefact assurance asset event 186
9.8 Impact analysis 187
9.9 Create new Executed Process 189
9.10 Creating Process Assurance data 191
9.11 Deleting Process Assurance Objects 193
9.12 Creation of Property Model 194
9.13 Edition of Properties 196
9.13.1 Add a property 196
9.13.2 Delete a property. 198

10 OpenCert Web Client 200
10.1 OpenCert Web interface layout 200
10.2 Compliance report 201
10.2.1 Goal of the report 201
10.2.2 Viewing compliance data on the report 202
10.2.3 Adding evidence and compliance data 204
10.2.4 Generation of summary textual report 206
10.3 Change Impact Analysis 207
10.3.1 Change Impact Analysis in OpenCert Client 207
10.3.2 Change Impact Analysis algorithm 207
10.3.3 Impact Analysis result presentation on OpenCert server reports 209
10.4 Gap Analysis report - Compliance Assessment and Evidence Evaluation 210

10.4.1 Gap Analysis report core functionality 211
10.4.2 Viewing Evidence Evaluation in Gap Analysis report 213
10.5 Metrics reports 213
10.5.1 Metrics Estimation Report 213
10.5.2 Equivalence Map Report 214
10.6 Administration web GUI 215
10.6.1 Projects Administration 215
10.6.2 Create Sample Data 216
10.6.3 Configuration Settings 217

11 Specification of Families/Lines 217
11.1 Process Lines 217
11.1.1 Importing Library and Fixing the Problems 217
11.1.2 BVR VSpec and Resolution Editors 218
11.1.3 Deriving the Processes and Export Back 220
11.2 Product Lines 220
11.2.1 VSpec, Resolution, and Realization 221

12 Integration with External Tools 224
12.1 V&V Tools (FBK) 224
12.2 Requirements Quality Analyzer (RQA)/OSLC-KM 224
12.3 Reuse Discovery using OSLC-KM (TRC) 227
12.4 SVN 231

References 232

Appendix A. Standard Modeling and compliance in EPF 233
A.1 Standard modeling 233
A.2 Process compliance 234
A.3 Recommendation Tables Modeling 237
A.4 Web-based monitoring of Compliance status 241

Appendix B. OCRA Language to define Formal Properties and Contracts 244

List of Figures

Figure 1 - Standard and Process Modeling 17
Figure 2 - Assurance Project and System Component Specification structure 19
Figure 3 - Select the workspace menu 20
Figure 4 - Preference menu 21
Figure 5 - Model Repository Configuration Page linked to a common repository 21
Figure 6 - Disable the Repository timeout time 22
Figure 7 - Opencert Perspective 23
Figure 8 - Repository Explorer content with configuration error 23
Figure 9 - Delete folder menu 24
Figure 10 - Delete model menu 24
Figure 11 - The Authoring Perspective 26
Figure 12 - The Browsing perspective 27
Figure 13 - Content packages of the ecss-e-st-40c_lifecycle 28
Figure 14 - Description tab of the form for the modeling of a Task 29
Figure 15 - Steps tab of the form for the modeling of a Task 29
Figure 16 - Roles tab of the form for the modeling of a Task 30
Figure 17 - Preview tab of the form for the modeling of a Task 31
Figure 18 - Dialog for the selection of Guidance elements in the context of a Task. 32
Figure 19 - Creation of a tool and Tool Mentors tab in the EPF composer 33
Figure 20 - Creation of a new delivery process in the EPF composer 34
Figure 21 - Modeling of workbreakdown structure of a delivery process 35
Figure 22 - Properties view in a Milestone. 35
Figure 23 - Consolidated View of the Delivery Process ECSS-E-ST-40_LifeCycle 36
Figure 24 - Activity diagram of a delivery process 37
Figure 25 - Adding a capability pattern in a delivery process. 37
Figure 26 - Extended Capability Pattern 38
Figure 27 - New Reference Framework model 39
Figure 28 - Wizard Reference Framework model 40
Figure 29 - New Reframework Diagram 40
Figure 30 - New Reframework Domain Model 41
Figure 31 - Reframework editor perspective 42
Figure 32 - Show properties view 43
Figure 33 - Deleted concept shown in a diagram 43
Figure 34 - Reframework Diagram wizard I 44
Figure 35 - Reframework Diagram wizard II 44
Figure 36 - Reframework Diagram wizard III 45
Figure 37 - Model tree editor 45
Figure 38 - Edit model from Outline 46
Figure 39 - Load Resource I 47
Figure 40 - Load Resource Reference Framework II 48
Figure 41 - Load Resource Map Group III 49
Figure 42 - Activity Equivalence Map 49
Figure 43 - Equivalence Map 50
Figure 44 - How to create Equivalence Map. 51
Figure 45 - Equivalence Map form. 51
Figure 46 - Equivalence Map, select map element (I) 52
Figure 47 - Equivalence Map with Postcondition. 53
Figure 48 - Steps for making Equivalence Map. 54

Figure 49 - Applicability Table ISO 26262 54
Figure 50 - Requirement Applicability Table ISO 26262 55
Figure 51 - Applicability Table DO-178C 55
Figure 52 - Activity Applicability Table DO-178C 56
Figure 53 - Requirement Applicability Table DO-178C 56
Figure 54 - Summary Applicability Table DO-178C 57
Figure 55 - New Assurance Project wizard 58
Figure 56 - Assurance Project name page 59
Figure 57 - Reframework selection 60
Figure 58 - Assurance Project structure 60
Figure 59 - Assurance Project editor 61
Figure 60 - Other kind of projects option 62
Figure 61 - Creates or Updates Baseline wizard 62
Figure 62 - Selection of the Assurance Project to update 63
Figure 63 - Assurance Project with new baseline 64
Figure 64 - Baseline editor 64
Figure 65 - Baseline graphical editor 65
Figure 66 - Assets Package active 66
Figure 67 - Baseline Config active 67
Figure 68 - Load Resource 67
Figure 69 - Load Resource II 68
Figure 70 - Load Resource Evidence, Process or Argumentation model 68
Figure 71 - Load Resource Mapping model 69
Figure 72 - Artefact Compliance Map 69
Figure 73 - Compliance Map 70
Figure 74 - How to create Compliance Map 71
Figure 75 - Compliance Map form 71
Figure 76 - Compliance Map, select map element 72
Figure 77 - How to access Compliance Mapping Table 73
Figure 78 - Mapping Table window 74
Figure 79 - Showing the target list of the Base element selected 75
Figure 80 - Compliance editor accessed via compliance mapping table 75
Figure 81 - Compliance map target element details accessed from compliance mapping table 76
Figure 82 - Cross Domain button 77
Figure 83 - Create a new evidence model message 77
Figure 84 - Use existing evidence model message 78
Figure 85 - Cross domain window 78
Figure 86 - Cross domain window with base element selected 79
Figure 87 - Cross domain information messages about integrity 79
Figure 88 - Reuse not equivalence artefacts confirmation message 80
Figure 89 - Cross domain final confirmation message 80
Figure 90 - Cross Project button 81
Figure 91 - Cross project: Source project selection 82
Figure 92 - Cross project: Copy all models 83
Figure 93 - Cross project: Copy only evidences 83
Figure 94 - Cross Project information message 84
Figure 95 - Cross project reuse result 84
Figure 96 - Cross-project Reuse View 85
Figure 97 - Using the Reuse View 86
Figure 98 - Export wizard of the EPF composer. 88
Figure 99 - Creation of Method Configuration in EPF 88
Figure 100 - Plug-in and Package selection of Method Configuration View. 89

Figure 101 - Assign dialog for Custom Category. 90
Figure 102 - Views tab of the Method Configuration view. 91
Figure 103 - Import from EPF button 92
Figure 104 - Import from EPF: Source files selection 92
Figure 105 - Import from EPF: Imported model names specification 93
Figure 106 - Import from EPF: Result of the import operation 93
Figure 107 - Import from EPF: Imported models automatically linked to the Assurance Project 94
Figure 108 - New Property Model menu File -> New -> Other 94
Figure 109 - New Mapping Model I 95
Figure 110 - New Mapping Model II 95
Figure 111 - New Mapping Model III 96
Figure 112 - Mapping Model 96
Figure 113 - Add New Map Group (I) 97
Figure 114 - Add New Map Group (II) 97
Figure 115 - Map Group properties 98
Figure 116 - Delete Map Group I. 98
Figure 117 - Delete Map Group II. 99
Figure 118 - Add New Map (I) 99
Figure 119 - Add New Map (II) 100
Figure 120 - Map properties 100
Figure 121 - Delete Map I. 101
Figure 122 - Delete Map II. 101
Figure 123 - Generation of process-based arguments button 102
Figure 124 - EPF Process to Argumentation Model: XML file selection and generation buttons 102
Figure 125 - Assurance project editor with the import from file button 105
Figure 126 - Import from File window 105
Figure 127 - Block Definition Diagram example 108
Figure 128 - Internal Block Diagram example 108
Figure 129 - Property Editor with content assist 109
Figure 130 - Creating a Formal Property 110
Figure 131 - Formalizing Requirements 111
Figure 132 - Popup to create a new contract or instantiate an existing one. 111
Figure 133 - The Add Contract button in the ContractEditor+ 112
Figure 134 - Editing the Contract's Assume and Guarantee 112
Figure 135 - ContractProperty 113
Figure 136 - Contract tab for instances 114
Figure 137 - Set Contract Refinement Command 115
Figure 138 - Refinement Selection 115
Figure 139 - Composite Aggregation 116
Figure 140 - Contract Refinement 116
Figure 141 - ErrorModel behavior 117
Figure 142 - Propagation path between two components 118
Figure 143 - Hierarchical view of the system decomposed into sub-components and contracts 119
Figure 144 - Contract Refinement View 119
Figure 145 – Validating contracts status 120
Figure 146 – The preferences window to configure the V&V tools. 121
Figure 147 - Perform the check of the contract refinements 122
Figure 148 - Perform the check implementation based on contracts 122
Figure 149 - Perform the consistency check of formal properties 123
Figure 150 - Validation Property parameters 123
Figure 151 - Perform the model checking of system behavior 124
Figure 152 - Model Checking parameters 124

Figure 153 - Invoke the V&V Manager 125
Figure 154 - Popup to set the preferences of the generated document and diagrams 126
Figure 155 - Connecting to the AMASS repository 128
Figure 156 - CDO Repositories view 128
Figure 157 - OpenCert tab for the Contract 129
Figure 158 - associating a Claim to a Contract 130
Figure 159 - Argumentation Preferences 132
Figure 160 - File-based Argumentation Diagram wizard I 133
Figure 161 - File-based Argumentation Diagram wizard II 133
Figure 162 - Database-based Argumentation Diagram wizard I 134
Figure 163 - Database-based Argumentation Diagram wizard II 134
Figure 164 - Open File-based Argumentation Diagram 135
Figure 165 - Open Database-based Argumentation Diagram 136
Figure 166 - Argumentation Palette 137
Figure 167 - Claim properties 139
Figure 168 - Initialize a diagram file 141
Figure 169 - Selection of the Case root element 141
Figure 170 - Database-based Argumentation Diagram wizard I 142
Figure 171 - Database-based Argumentation Diagram wizard II 142
Figure 172 - Database-based Argumentation Diagram wizard III 143
Figure 173 - Artefact selection as solution 143
Figure 174 - Artefact selection from resources 144
Figure 175 - Artefact edition form 144
Figure 176 - Load Resource to Argumentation Diagram 145
Figure 177 - Select Evidence model as resource 145
Figure 178 - Linking models to the Assurance Project's Assets Package 146
Figure 179- Selecting the evidence model to be included in the assets package 146
Figure 180 - Argumentation Templates View 147
Figure 181 - Claim properties (To Be Instantiated) 148
Figure 182 - Claim properties (Multiextension) 148
Figure 183 - Example of the software contribution safety argument pattern [5] 149
Figure 184 - Claim properties (declared as Public) 150
Figure 185 - ArgumentElementCitation properties (reference to a claim) 150
Figure 186 - Argumentation Module 150
Figure 187 - New Vocabulary 151
Figure 188 - Example Vocabulary Diagram 152
Figure 189 - Vocabulary Import 152
Figure 190 - Mark-up Rendering 153
Figure 191 - Tooltip 153
Figure 192 - Term Suggestions 154
Figure 193 - Initiating the argument-fragment generation (Step 1) 155
Figure 194 - Selecting the source analysis context (Step 2) 156
Figure 195 - Selecting the destination assurance case folder on the CDO repository (Step 3) 157
Figure 196 - Generation successfully completed with argument-fragments for each block 158
Figure 197 - An example of the generated argument-fragment 159
Figure 198 - Export wizard selection window 160
Figure 199 - Export argument model window 160
Figure 200 - Preference menu 161
Figure 201 - Artefact Repository Preferences 162
Figure 202 - New Evidence Model menu File -> New -> Other 162
Figure 203 - New Evidence Model I 163
Figure 204 - New Evidence Model II 163

Figure 205 - New Evidence Model III 164
Figure 206 - Evidence Model 164
Figure 207 - Add New Artefact Definition (I) 165
Figure 208 - Add New Artefact Definition (II) 165
Figure 209 - Artefact Definition Description (I) 166
Figure 210 - Artefact Definition Description (II) 166
Figure 211 - Description Artefact Definition Artefact 167
Figure 212 - Description Artefact Definition Evaluation 167
Figure 213 - Description Artefact Definition Events 167
Figure 214 - Delete Artefact Definition (I) 168
Figure 215 - Delete Artefact Definition (II) 168
Figure 216 - Add New Artefact (I) 169
Figure 217 - Add New Artefact (II) 169
Figure 218 - Artefact Description 170
Figure 219 - Description Artefact Version 171
Figure 220 - Description Artefact Property Value 171
Figure 221 - Description Artefact Evaluation 171
Figure 222 - Description Artefact Events 172
Figure 223 - Delete Artefact I. 172
Figure 224 - Delete Artefact II. 173
Figure 225 - Add Artefact Resource I. 173
Figure 226 - Resource properties 174
Figure 227 - Add Artefact Resource II. 174
Figure 228 - Resource dialog box 175
Figure 229 - Select Artefact from the local drive. 175
Figure 230 - Select Artefact from the SVN Remote Repository 176
Figure 231 - SVN History table of a File 176
Figure 232 - Delete Artefact Resource I. 177
Figure 233 - Delete Artefact Resource II. 177
Figure 234 - Load Resource Property model. 178
Figure 235 - Select Property model. 178
Figure 236 - Add Artefact Property Value I. 179
Figure 237 - Artefact Value dialog box 179
Figure 238 - Add Artefact Property Value II 180
Figure 239 - Artefact Property properties 180
Figure 240 - Delete Artefact Property Value I 181
Figure 241 - Delete Artefact Property Value II 181
Figure 242 - Add Artefact Assurance Asset Evaluation I 182
Figure 243 - Artefact Assurance Asset Evaluation dialog box 182
Figure 244 - Add Artefact Assurance Asset Evaluation II 183
Figure 245 - Artefact Assurance Asset Evaluation properties 183
Figure 246 - Delete Artefact Assurance Asset Evaluation I 184
Figure 247 - Delete Artefact Assurance Asset Evaluation II 184
Figure 248 - Add Artefact Assurance Asset Event I 185
Figure 249 - Artefact Assurance Asset Event dialog box 185
Figure 250 - Add Artefact Assurance Asset Event II 186
Figure 251 - Artefact Assurance Asset Event properties 186
Figure 252 - Delete Artefact Assurance Asset Event I 187
Figure 253 - Delete Artefact Assurance Asset Event II 187
Figure 254 - Artefact modified with automatically generated events 188
Figure 255 - Artefact analyser confirmation windows 188
Figure 256 - Artefact events created by Impact Analyser 189

Figure 257 - New Process Model I 190
Figure 258 - New Process Model II 190
Figure 259 - New Process Model III 191
Figure 260 - Process Model 191
Figure 261 - Create Process Model data using context menu 192
Figure 262 - Create Process Model data using properties View 192
Figure 263 - Delete Process Model data using context menu 193
Figure 264 - Delete Process Model data using properties view 194
Figure 265 - New Property Model menu File -> New -> Other 194
Figure 266 - New Property Model I 195
Figure 267 - New Property Model II 195
Figure 268 - New Property Model III 196
Figure 269 - Property Model 196
Figure 270 - Add New Property (I) 197
Figure 271 - Add New Property (II) 197
Figure 272 - Property properties 198
Figure 273 - Add enum values. 198
Figure 274 - Delete Property I. 199
Figure 275 - Delete Property II. 199
Figure 276 - Web interface layout 200
Figure 277 - Menu item directing to “Compliance report” 202
Figure 278 - Baseline Frameworks combo box for the specific project 202
Figure 279 - 4 panels of "Compliance report" 202
Figure 280 - Description of the selected baseline element presented at the bottom panel 203
Figure 281 - Details of Justification and mapped evidence 203
Figure 282 - Compliance evidence of the specific baseline asset 204
Figure 283 - Specific evidence details description presented at the bottom 204
Figure 284 - A window allowing to assign and describe evidence to the given baseline item 205
Figure 285 - Unassign button allowing to disassociate evidence from the given baseline item 205
Figure 286 - “Export to MS Word” button which generates textual overall detailed report of Project Compliance to the safety standard 206
Figure 287 - First page of the generated textual report 207
Figure 288 - Artefact Model 208
Figure 289 - Artefact lifecycle from the IA point of view 209
Figure 290 - Web interface showing two IA-induced actions required to be taken by user 210
Figure 291 - Gap Analysis report 211
Figure 292 - Baseline frameworks for the specific assurance project 211
Figure 293 - Project baseline compliance table 212
Figure 294 - Compliance details for the selected baseline element 212
Figure 295 - Compliance details 212
Figure 296 - Evidence evaluation details 213
Figure 297 - Menu item directing to "Metrics Estimation report" 213
Figure 298 - Metrics Menu in the top-left portion of the report 214
Figure 299 - Description of the selected metric type presented at the left 214
Figure 300 - Selection of reference frameworks 215
Figure 301 - Equivalence Map Report 215
Figure 302 - Administration menu 215
Figure 303 - Project Administration web page on OpenCert server 216
Figure 304 - Create sample data page 217
Figure 305 - New OSLC-KM menu option 224
Figure 306 - Import Papyrus file wizard page #1 225
Figure 307 - Import Papyrus file wizard page #2 225

Figure 308 - Import Papyrus file success message 225
Figure 309 - Evidence imported in AMASS database 226
Figure 310 - New OSLC-KM menu option for Reuse Discovery 227
Figure 311 - Wizard page #1 for file selection 228
Figure 312 - Wizard page #2 for “Index file” operation 228
Figure 313 - Wizard page #2 for “Look for similar models” operation 229
Figure 314 - Confirmation dialog for “Index file” operation 229
Figure 315 - Result dialog for “Look for similar models” operation 229
Figure 316 - Icon customization of Requirement practice. 233
Figure 317 - Definition of a new standard requirement. 234
Figure 318 - Standard requirements modeled in the EPF composer. 234
Figure 319 - Method library organization for standard mapping. 235
Figure 320 - Mapped Requirements in the EPF composer. 235
Figure 321 - Preview tab of the mapped requirement "Development of the Software". 236
Figure 322 - Compliance situations in the EPF composer. 236
Figure 323 - Customized practices to model recommendation tables 237
Figure 324 - Modeling of criticality level for ISO26262 238
Figure 325 - Recommended plug-in structure for modeling recommendation tables 238
Figure 326 - Mechanisms for error detection at the software architectural level of ISO 26262. 239
Figure 327 - Adding recommendations to the table 239
Figure 328 - Link between recommendation and recommended element. 240
Figure 329 - Reference tab for applicability practice. 240
Figure 330 - Recommendation tables for ISO 26262 in the Browsing perspective 241
Figure 331 - Dialog to select Mapped Requirements. 242
Figure 332 - Dialog for publishing options of the generated website. 243
Figure 333 - EPF composer generate website. 244
Figure 334 - The OCRA language grammar. 245

List of Tables

Table 1 - Workbenches AMASS tool platform 16
Table 2 - Argumentation graphical notation 138

1

1 Executive Summary

This document is a user manual of the AMASS platform tools prototype implementation. In this document the user can find the installing instructions, the tool environment description, and the functionalities starting from the creation of models representing Standards, and Company-specific Processes, creation of Assurance Projects and the associated Baseline (subset of Standards to be applied in a specific assurance project), Evidences models, executed Process model (Activities) Compliance Maps (so far, compliance maps from Reference Artefacts to Artefacts), Argumentation model and web interface reports.

Finally, functionality facilitated by AMASS web UI server has been described.

This document has been elaborated as a Fast User Manual. Further questions must be directed to the TECNALIA team.

2 AMASS platform - basic concepts

The AMASS platform is composed of a set of tools providing the functionalities described in the AMASS deliverable D2.2 (AMASS Reference Architecture, first prototype). This first prototype has been built upon three pre-existing toolsets:

- Tools from the pre-existing OpenCert project[1].
- Tools from the CHESS Project (Polarsys Platform)[2].
- Tools from the EPF (Eclipse Process Framework) Project[3].

2.1 Naming conventions in AMASS clients

In this document, the naming convention follows the following concepts:

- **Environment** supports (a large part of) the software tool process. AMASS tool platform is the main environment in this document.
- **Workbenches** support only one or a few activities. Example: "*Evidence Management*" workbench.
- **Tool** support only specific tasks in the software tool process. Example: "Evidence Analysis".

The AMASS tool platform has the following Workbenches:

Workbench	Description
Assurance Project Lifecycle Management	This functionality factorizes aspects such as the creation of safety assurance projects locally in AMASS and any project baseline information that may be shared by the different functional modules. A project baseline is a subset of reference framework (e.g., subset of a standard) that will be applied to a given assurance project.
Compliance Management	Functionality related to the management (edition, search, transfer, etc.) of process and standards' information as well as of any other information derived from them, such as interpretations about intents and mapping between processes and standards. This functional group maintains a knowledge database about "standards & processes", which can be consulted by other AMASS functionalities.
Assurance Argumentation Management	This group manages argumentation information in a modular fashion. It also includes mechanisms to support compositional multi-concern assurance, and assurance patterns management.
Evidence Management	This module manages the full life-cycle of evidences and evidence chains. This includes evidence traceability management and impact analysis. Evidence management includes a model for the executed process, which includes information about executed activities, participants and techniques used. In addition, this module is in charge of communicating with external engineering tools (requirements management, implementation, V&V, etc.).
System Component Management	This group manages System architecture specification by decomposing a system into components. It also includes mechanisms to support compositional assurance, contract based approaches, and architectural patterns management.

Table 1 - Workbenches AMASS tool platform

[1] Further information about the OPENCOSS toolset can be found at www.opencoss-project.eu and https://www.polarsys.org/projects/polarsys.opencert
[2] Further information about CHESS toolset can be found at https://www.polarsys.org/chess/
[3] Further information about the EPF toolset can be found at http://www.eclipse.org/epf/

2.1 Approach for organising OpenCert projects and models

The information managed by OpenCert tools can be organised in two types: project-independent information that can be used by various projects (e.g., models of generic Process and Standards) and project-specific information (e.g., evidence and argumentation models).

2.1.1 Project-independent information

Users can use the Reference Framework Editor to model Standards (IEC 61508, ISO 26262, DO-178C, EN 50126, and the like), any Regulations (either as additional Requirements or model elements in a given model representing a Standard or a new Reference Framework), and EPF to model Company-specific processes (e.g., the Alstom, Thales or Fiat process to develop safety-critical systems). Please note that EPF can be used to model Standards as described in Appendix A.

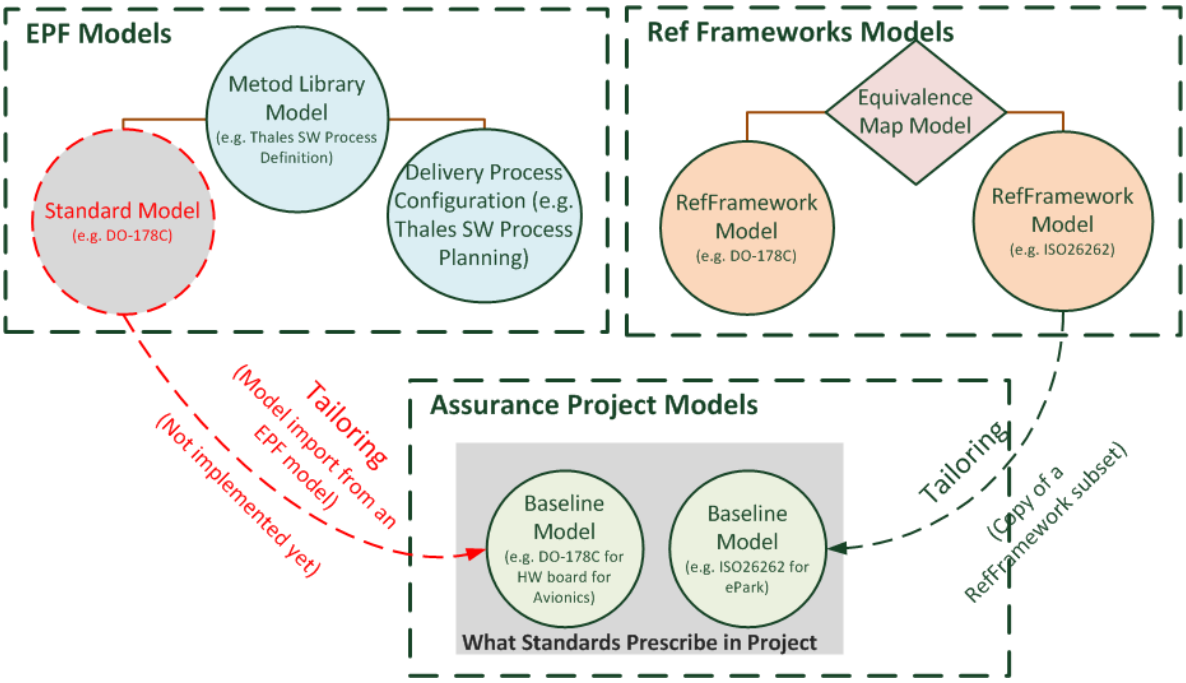


Figure 1 - Standard and Process Modeling

Each Reference Framework model can be also mapped to other Reference Framework models by using the concept of Equivalence Map.

Finally, Reference Frameworks can be used to create Assurance Project Baselines. Baseline Models represent the subset of the Reference Frameworks tailored for individual Assurance Projects.

The import option to create Baseline Models from EPF has not implemented yet.

2.1.2 Project-specific information

The main element of project-specific information is the Assurance Project. An Assurance Project has three main elements:

1. **Baseline Configuration**: a Baseline Configuration has a set of Baseline Models. Each baseline model results from importing (copying) a Reference Framework model and adding information about its Selection in the current project (it answers to the question: does a given Reference Framework model element apply to the current Assurance Project?). A Baseline model represents what is planned to comply with, in a specific assurance project.
2. **Permissions Configuration**. This has not been implemented yet. It will support profile creation to enable restricted access to AMASS functionality and data.
3. **Assurance Assets Package**. This is a pointer to project-specific Artefacts models, and Argumentation models, and Process models. These three models represent what has been done in a specific assurance project. The mapping of these three models with Baseline Models is modelled using the concept of Compliance Map.

One Assurance Project can have multiple Baseline Configurations, Permissions Configurations and Assurance Assets Package, but only one is active at once.

Assurance Projects are linked to System Component models that will be managed by the CHESS toolset. The link has not been implemented yet.

Additionally, Evidence and Process models can be created using EPF process planning models. This helps users to get a first version of their evidence and executed process models to demonstrate compliance with standards.

The next figure illustrates the elements of an Assurance Project:

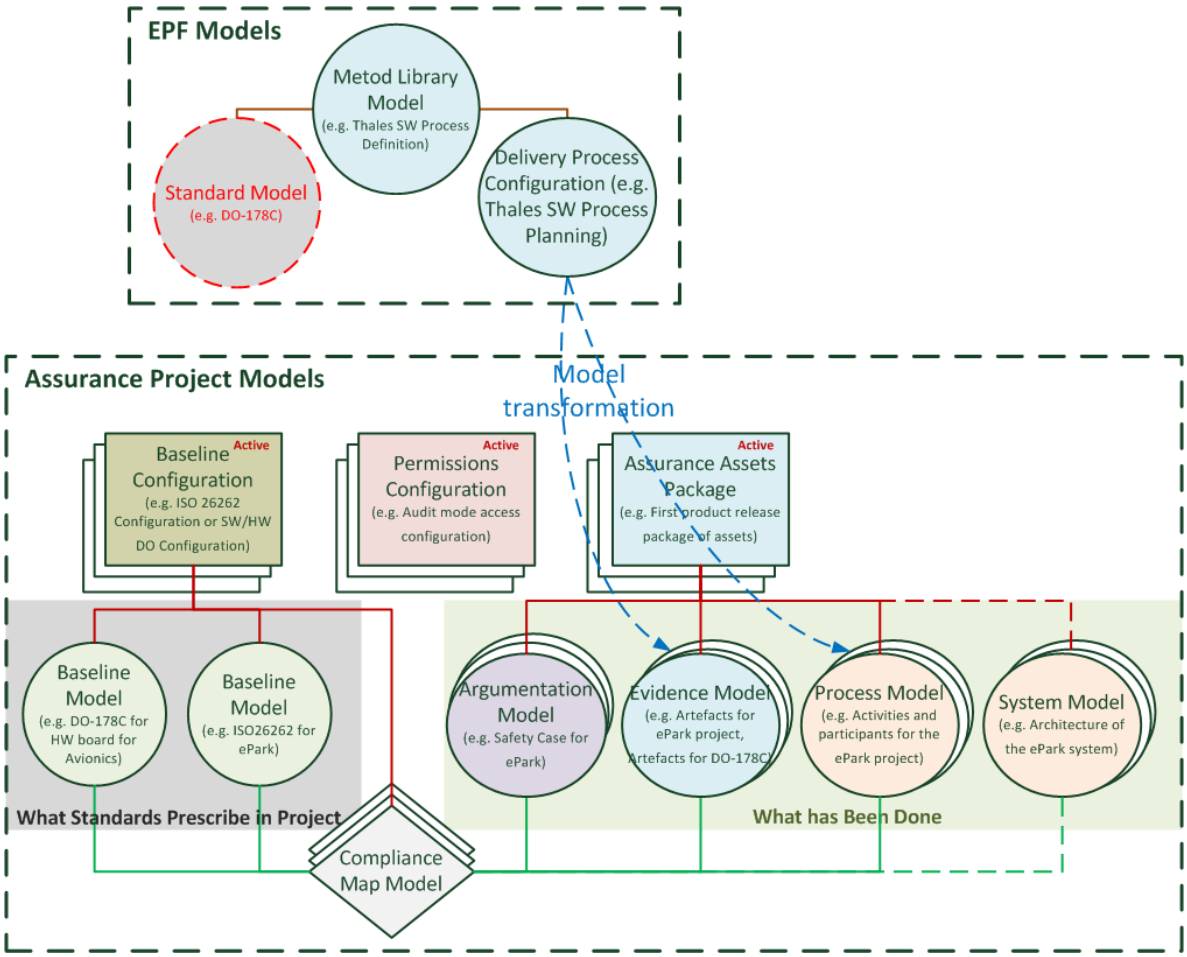


Figure 2 - Assurance Project and System Component Specification structure

3 Installation

The following chapters describe how to install the different toolsets.

It must be noted that the pre-existing OpenCert tools are designed to follow client-server architecture approach:

- **OpenCert server** - installed in a central host machine
- One or many **OpenCert clients** - each of which installed on specific user machines

The CHESS toolset also follows this approach.

EPF toolset is deployed in an Eclipse standalone version.

3.1 Installation of AMASS platform client

3.1.1 Client bundle download

It is required to have installed (minimum) **Java Environment 1.8**.

To install the AMASS platform client download it from project SVN using the links below for Windows 64 bits and uncompress it into your hard disk.

WP-transversal\ImplementationTeam\PrototypeCore\Tools\OpenCertCHESS\20170206_OpenCertCHESSClient_Win_x64.zip

For those users behind proxies, they should take into account that different clients use ports 2036 and 8080 for communications.

3.1.2 Client configuration

To use the platform execute the eclipse.exe file and introduce a select a folder that will be used as workspace.

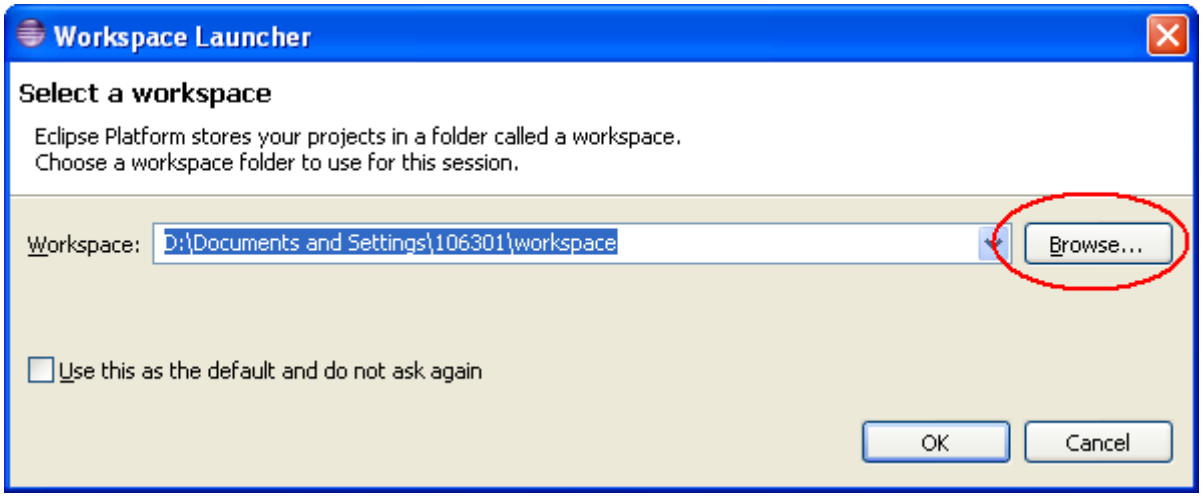


Figure 3 - Select the workspace menu

The first step after the installation process is to configure the connection settings with the CDO repository where all the models generated using the platform will be stored. This information must be introduced in the Model Repository Preference page inside the OpenCert category. Go to menu Window → Preferences to open this window.

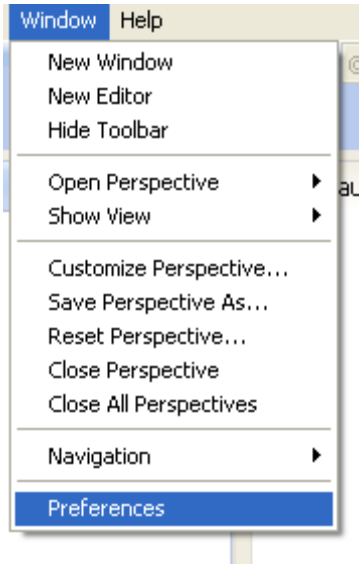


Figure 4 - Preference menu

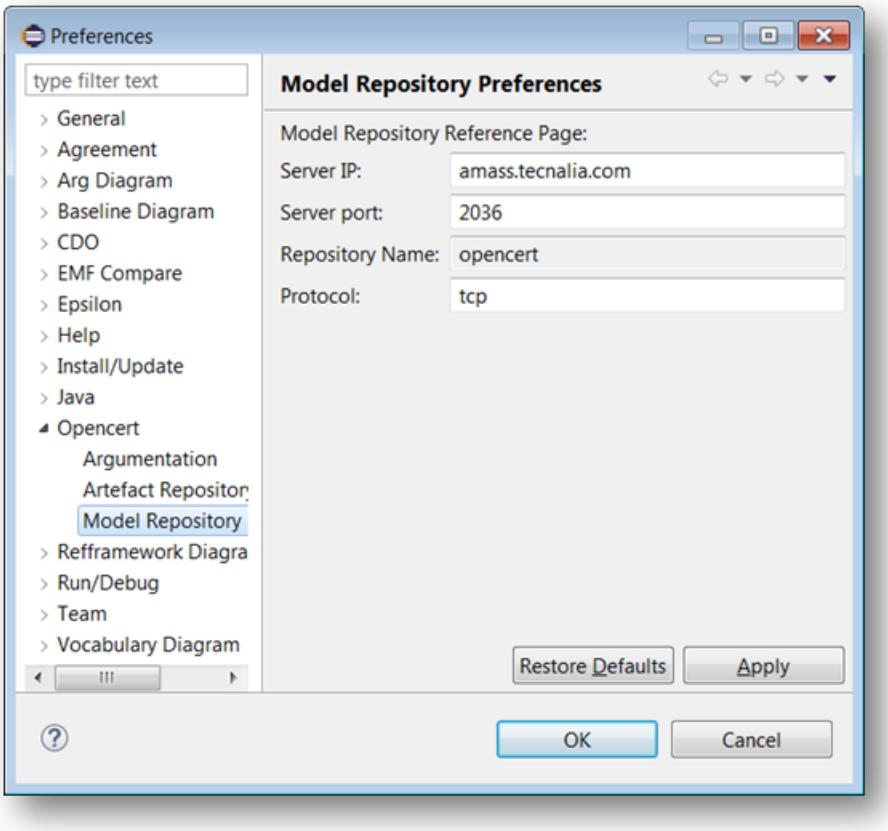


Figure 5 - Model Repository Configuration Page linked to a common repository

The information to introduce is:

- **Server IP**: The IP of the centralised CDO Server (amass.tecnalia.com).

- **Server Port**: The port used by the running CDO Server. Take into account that if the client is behind a proxy, the port 2036 shall be open, otherwise the communication will fail.
- **Repository name**: The name of the repository where all the date will be stored (read only).
- **Protocol:** The protocol used to connect to the CDO Server.

Disable the Repository timeout to avoid the automatic closing of the connection with the server for inactivity using the menu Window → Preference as shown in the image below.

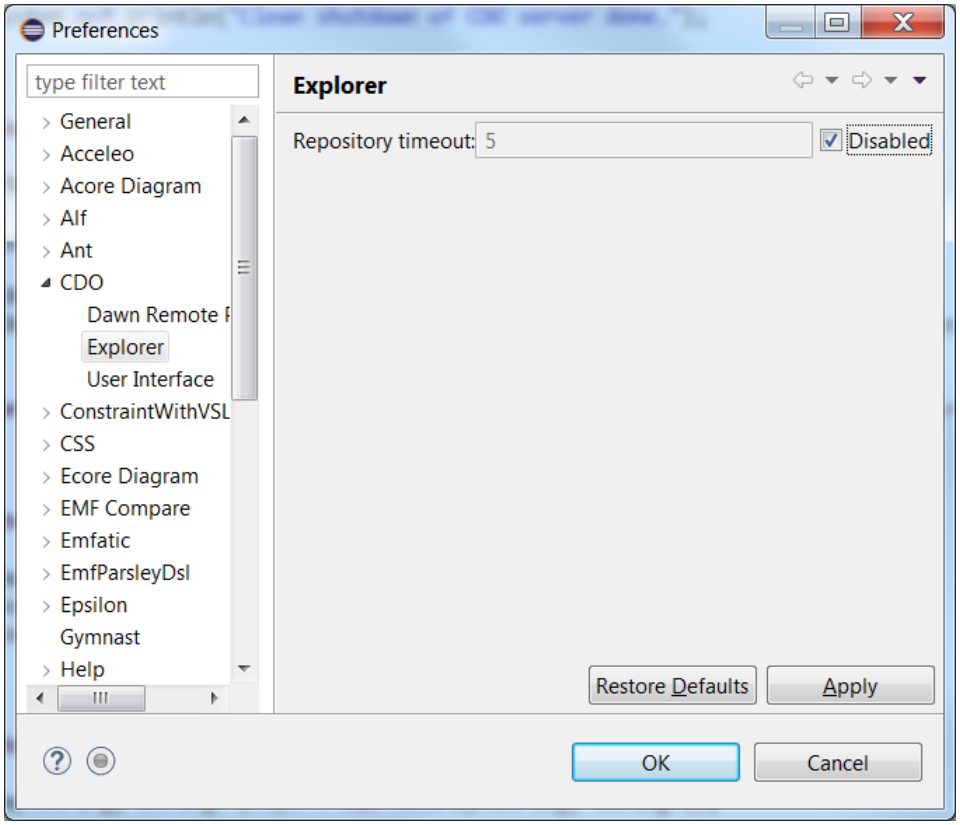


Figure 6 - Disable the Repository timeout time

After introducing this data the OpenCert Perspective can be used to connect to the server and view the data of the repository configured. To open it, go to menu Window→ Perspective → Open Perspective→ Other.

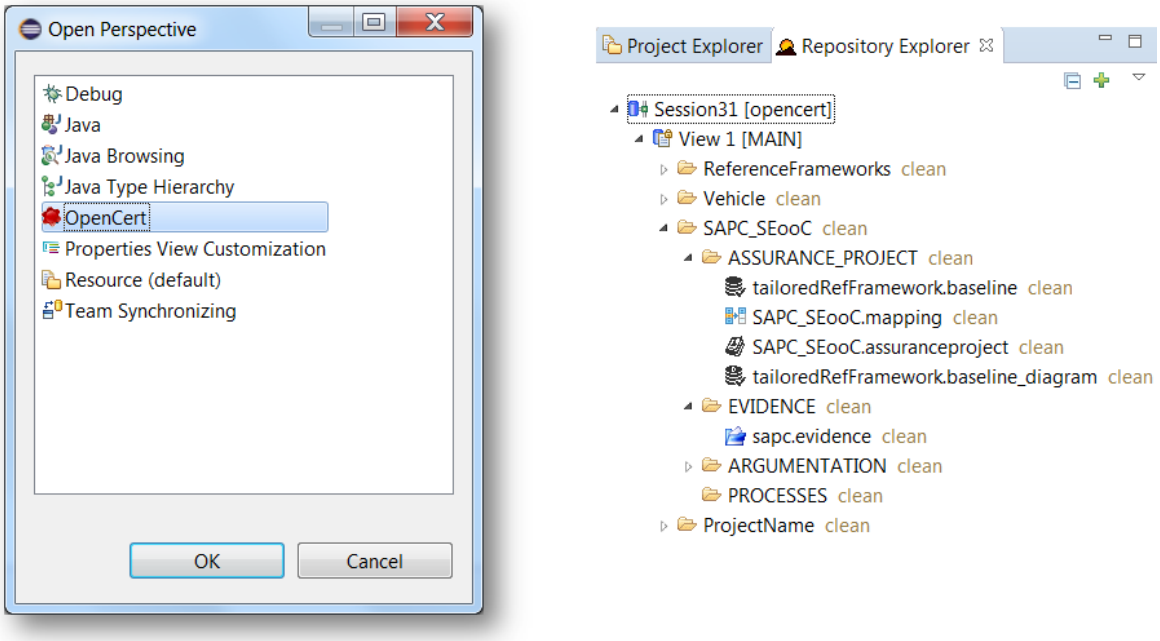


Figure 7 - Opencert Perspective

If the provided connection settings with the repository are incorrect or the server is not running, this view will display the error in the screenshot below instead of the contents of the repository. To solve it, check the server is running (using the page http://amass.tecnalia.com:8080) and the configuration settings are correct, ensure the communication ports (2036 and 8080) are open, close the Repository Explorer view and open it again (Window→Show View→Other→Opencert→Repository Explorer).

Figure 8 - Repository Explorer content with configuration error

3.1.3 Deleting Repository contents

To delete a folder and its contents, right click over it and left click the “Delete” menu.

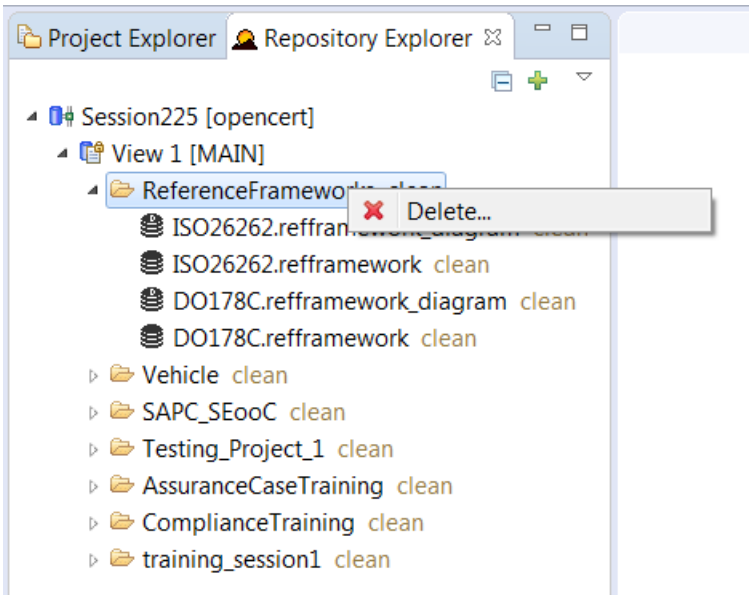


Figure 9 - Delete folder menu

To delete a model, right click over it and left click the “Delete” menu.

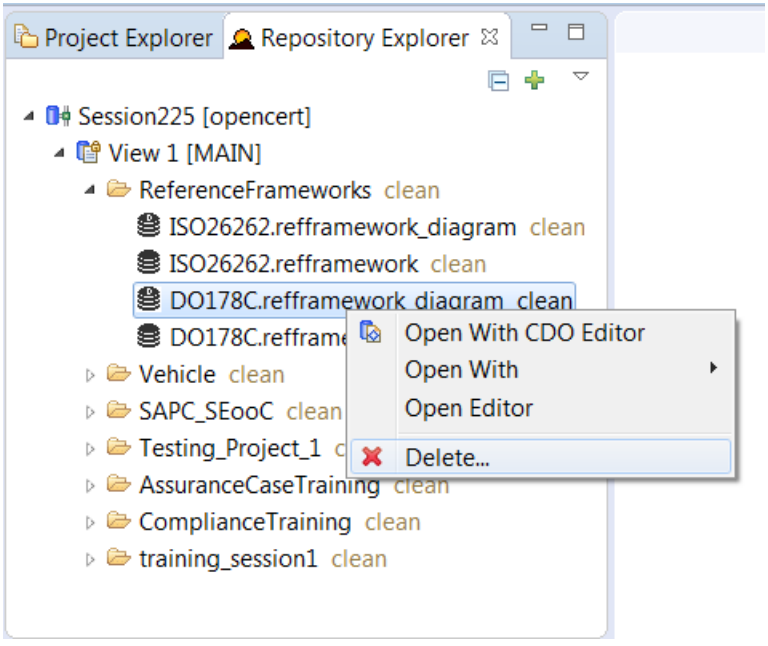


Figure 10 - Delete model menu

3.2 Installation of the EPF Composer

The installation instructions of EPF can be found in Section 1 of the EPF composer manual [1]. The last version of EPF is 1.5.18 and can be downloaded in the EPF download website[4]. The system requirements for this version are the following:

- Microsoft Windows XP SP3, 2003 SP2 (or later),Windows 7, Windows 10
- Red Hat Enteprise Linux Release 4 Update 5, Release 5 or later, (note: compat-libstdc++ is needed for RHEL5) SUSE Enterprise Linux v9 or v10
- Internet Explorer, Mozilla, or Firefox

[4] https://eclipse.org/epf/downloads/tool/epf1.5.0_downloads.php

- Java Runtime Environment 1.5, 1.6, 1.7, 1.8

EPF is a standalone Eclipse application, so once it is downloaded and unzipped, you do not need additional installation to start the work.

4 Process Modeling with EPF

The purpose of this part of the manual is to provide an overview of the EPF composer functionalities that are relevant for the compliance management building block and cannot be seen as a replacement of the official manual of this tool [[1]].

The functionality of the EPF composer is organized in two views, the Authoring (opened by default in the EPF Composer) and the Browsing perspective. The goal of the Authoring perspective is to provide functionality to formally model process element and processes, while the goal of the Browsing perspective is to present the contents modeled of the Authoring perspective. So, most of the work of the user will take place in this last perspective.

Figure 11 shows a screenshot of the Authoring Perspective in the EPF Composer. In this perspective we can distinguish three parts, the Method library (left top of the workbench), the configuration (left bottom of the workbench) and the process element/process modeling space (right part of the workbench) that in this case, it is showing the modeling of a delivery process.

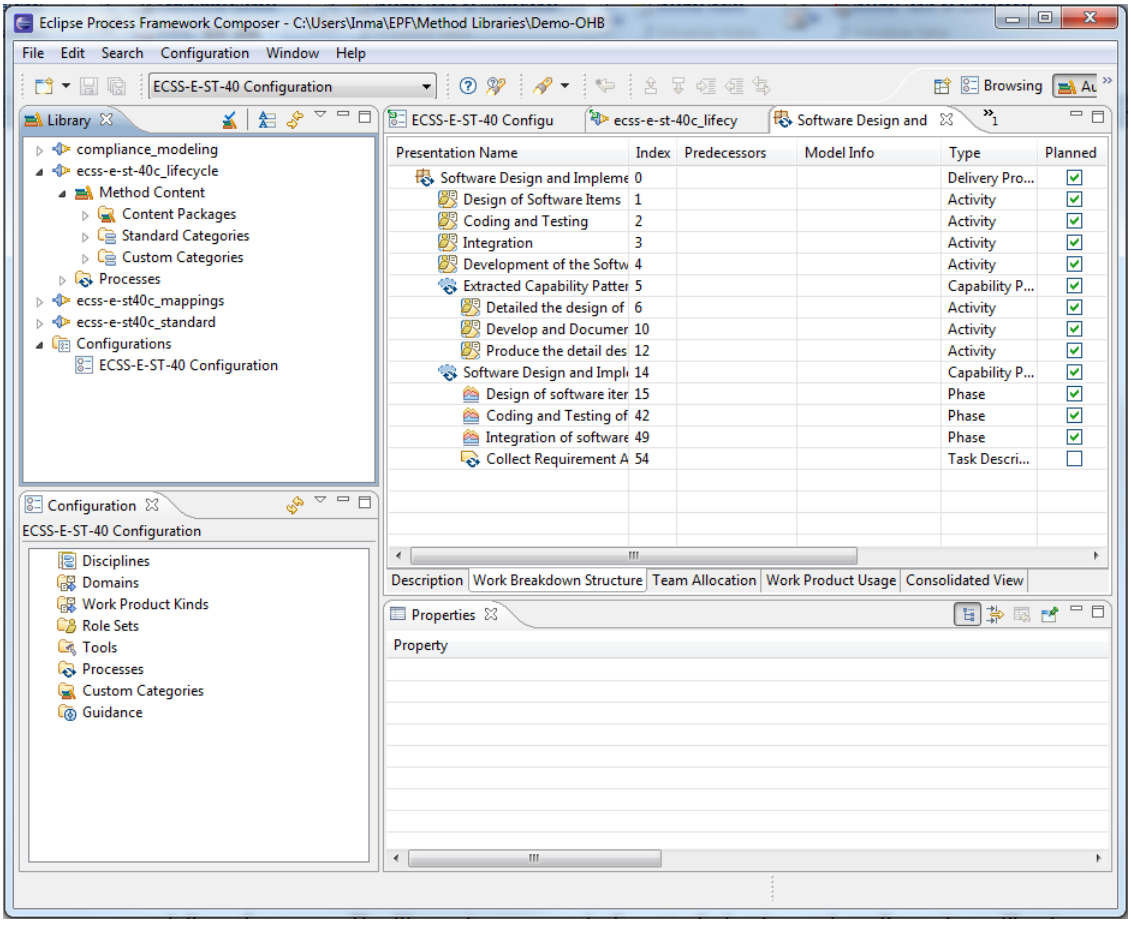


Figure 11 - The Authoring Perspective

The Method Library is the structure used by EPF to organize the contents related to the modeling of process. The library is composed of a set of plug-ins and configurations. Plug-ins are containers of process related information, while a configuration is a selection of the contents of the library to be shown in the Browsing perspective.

In order to open the Browsing perspective, we select **Windows -> Open perspective -> Browsing**. This perspective is merely for presentation purposes and the contents modeled in the Authoring perspective are classified in the Configuration view. If you click one of the elements in the configuration view, the content window depicts detailed information of the process or process element. In this case, it is depicted the Work Product Usage of the capability pattern: ECSS-E-ST-40_LifeCycle_Pattern.

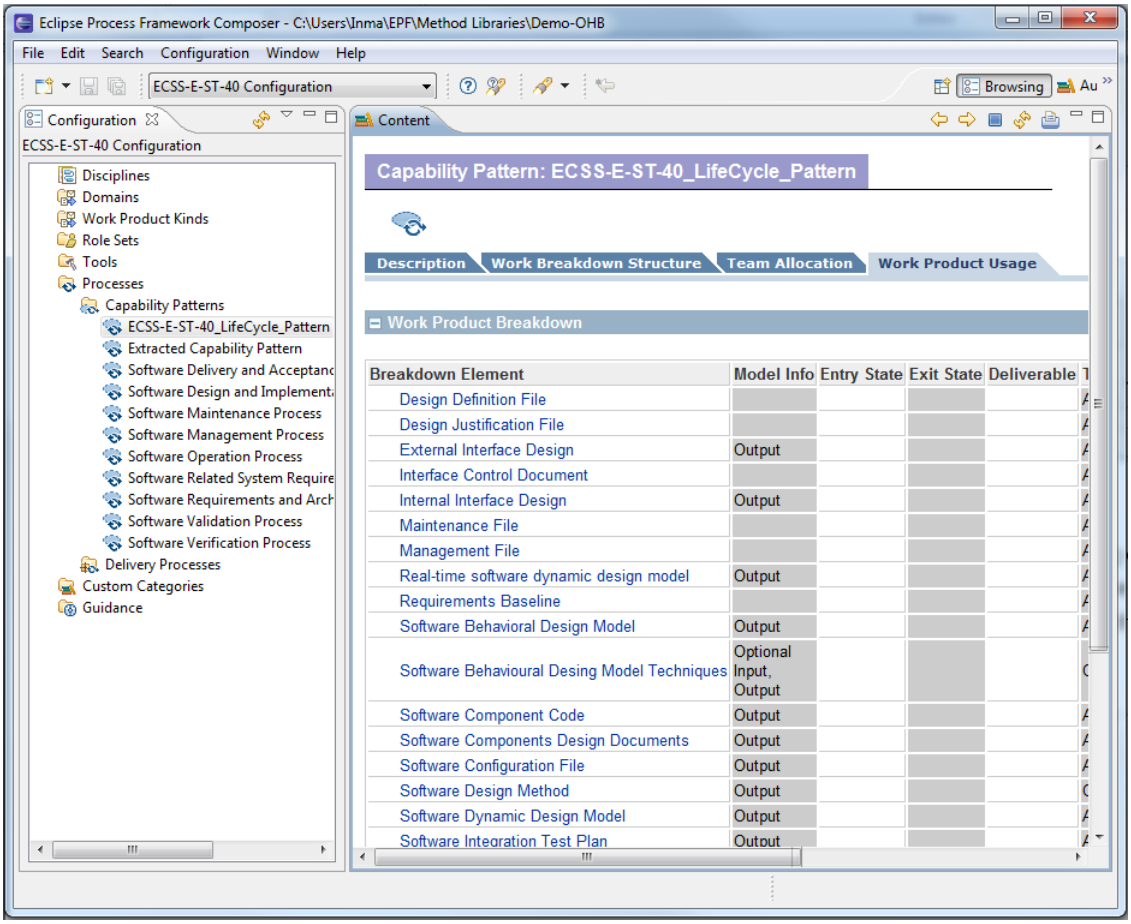


Figure 12 - The Browsing perspective

The Section 4.1 of the EPF user manual provides a detailed explanation of the different workbenches available in the EPF composer.

4.1 Modelling of reusable process elements

The EPF composer fully supports the re-use capacities of the SPEM 2.0 standard [2]. Therefore, it is possible to define a library of process elements that can be re-used to assemble different processes. Section 4.2 of the EPF composer manual provides guidance in the definition and re-use of the process elements. In this section, we show a small example of the standard ECSS-E-ST-40C. Specifically, we will model process elements for the software design and implementation phase of this standard.

In order to model contents related to a process, we firstly create or import a method Library and later we create Method plug-ins or to import an existing one (Section 6 of EPF user manual). You can create as much plug-ins as you need; in our case we create/import four method plug-ins (see Figure 11): compliance_modelling, ecss-e-st-40c_lifecycle, ecss-e-st40c_mappings and ecss-e-st40c_standard.

The creation of the contents for the process is made in the Content Packages of the plug-in (see Figure 11). In Section 4.2.3 of the EPF user manual, we can find a detailed tutorial for this. Figure 13 shows all the content packages defined for the plug-in that will be used to model the process depicted in the standard ECSS-E-ST-40C. As in the case of the plug-ins, it is recommendable to have different packages to organize the contents. Inside each package, we have different folders that will be used to model Roles, Tasks, Work Products and Guidance. By just right-click one of the folder, we can create a new process element (i.e. role, task, work product or guidance) and the edition panel for it opens automatically. Sections from 4.2.4 to 4.2.9 provide information about how to define the different process elements of EPF. In this section, we will focus on the definition of the task "Definition and Documentation of software unit tests" and the tool mentor "Eclipse Tool Mentor".

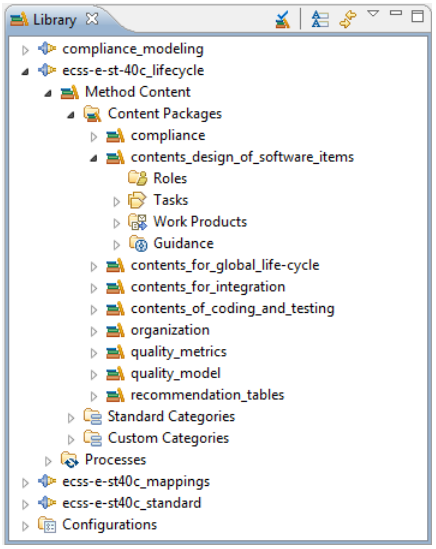


Figure 13 - Content packages of the ecss-e-st-40c_lifecycle

In order to create a new task, we right click the Tasks folder of a content package and select "**New -> Task"**. Then, a form will be automatically opened (Figure 14) that will be used to add information about the task. The form is composed by different tabs: Description, Steps, Role, Work Products, Guidance, Categories and Preview. In the different tabs of the form, you can enrich the definition of your process element (in this case a task) by adding other process elements (i.e. roles, work products…) and providing additional information like descriptions, alternatives in the use of the process element. The Description tab is common to all the elements that can be defined in EPF and it used to give a name to the element and to add detailed information. The naming conventions of EPF, distinguish between the Name of an element ("definition_and_documentation_of_software_unit_tests") and the Presentation name ("Definition and Documentation of software unit tests"). The Name of an element is mainly used in the Authoring perspective, while the presentation name is used in the Browsing Perspective and in the generated website.

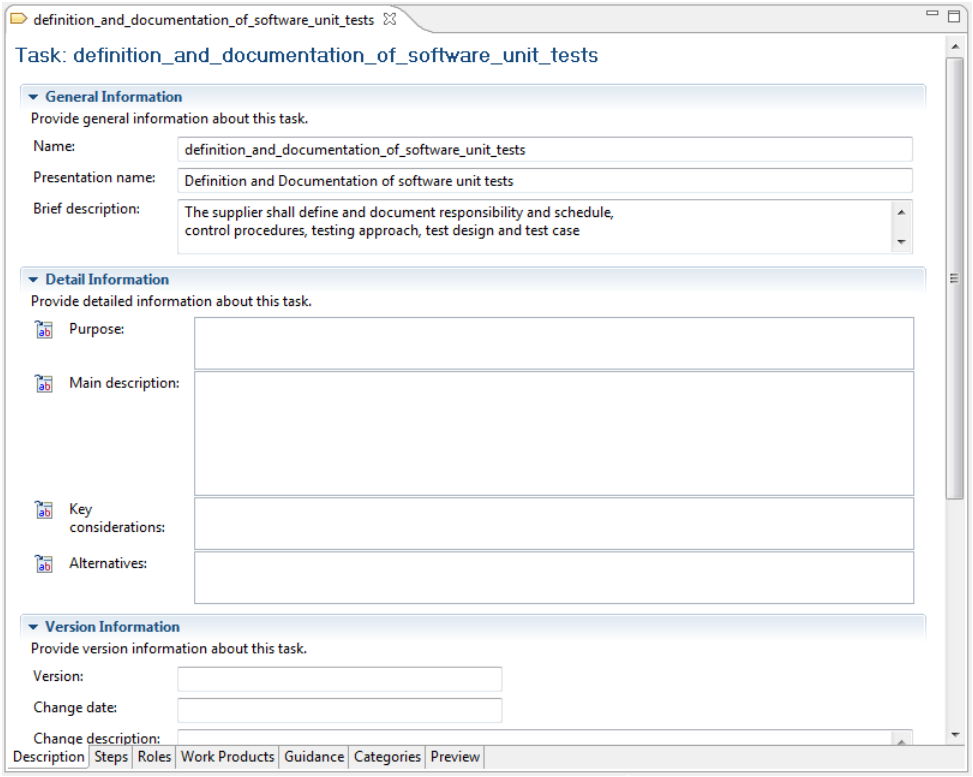


Figure 14 - Description tab of the form for the modeling of a Task

In the Step tab (Figure 15), it is possible to add the different steps that compose the task. Steps are added by clicking the button “Add..”, can be deleted using the “Delete” button and their order can be modified using the buttons “Up”, “Down” or “Order”. Additionally, we can provide a description of the step in the field “Description”.

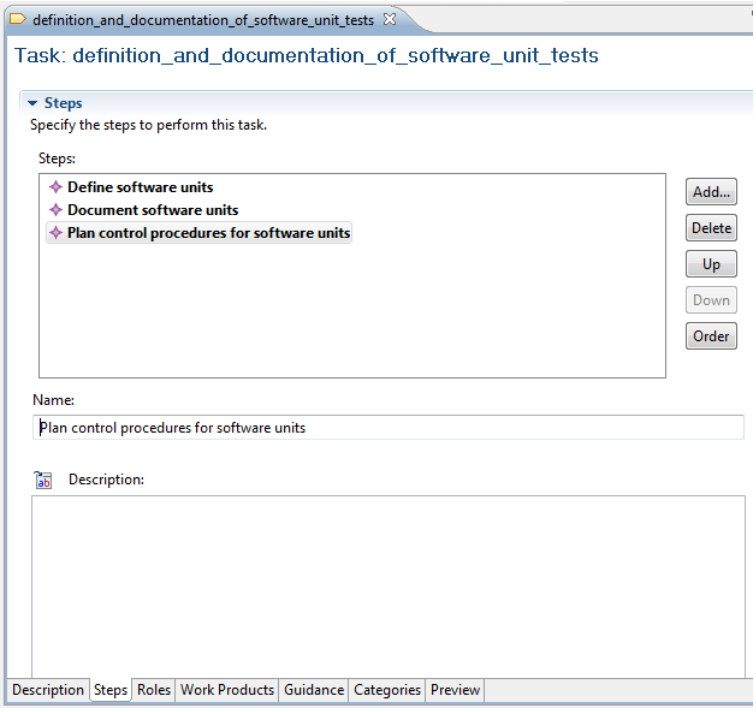


Figure 15 - Steps tab of the form for the modeling of a Task

In the Roles tab, all the roles involved in the execution of the task are added. EPF distinguish between Primary performer and Additional Performers. You can add an already defined Role (see Section 4.2.5 of EPF manual) in the corresponding category by using the buttons “Add…”.

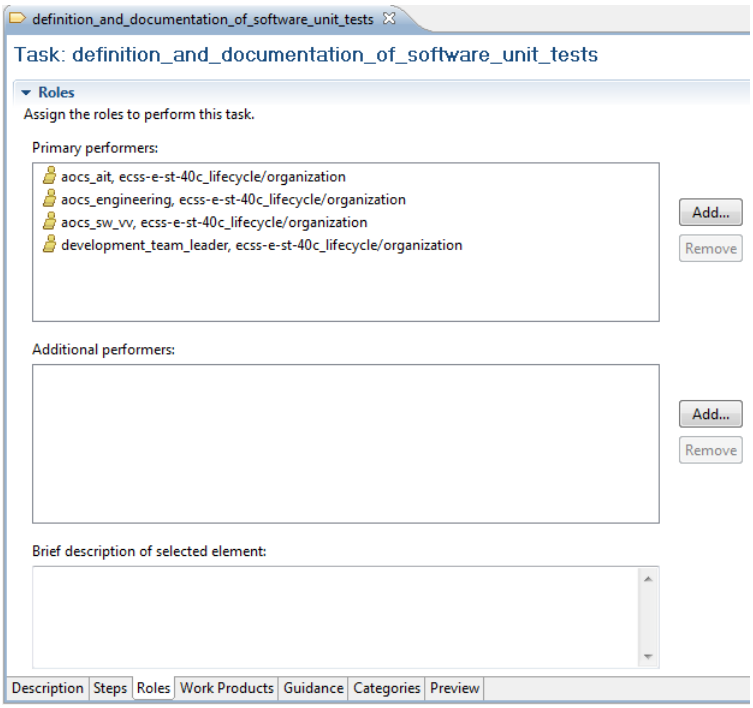


Figure 16 - Roles tab of the form for the modeling of a Task

In the Work Product tab, it is possible to add work products that are input, optional input or output of the task. While in the Guidance tab, supporting material for the execution of the tasks like tool mentors, white papers or guidelines are added. Categories are used to classify tasks in different disciplines. Finally, in the Preview tab (see Figure 17), we can see the result of our modeling work. This preview shows how this element will be presented in the Browsing perspective and in the generated website. In this is interesting to note that all sentences in blue are links to process elements and process. So, it is possible to navigate between the different elements related to our task by simply clicking in the corresponding link.

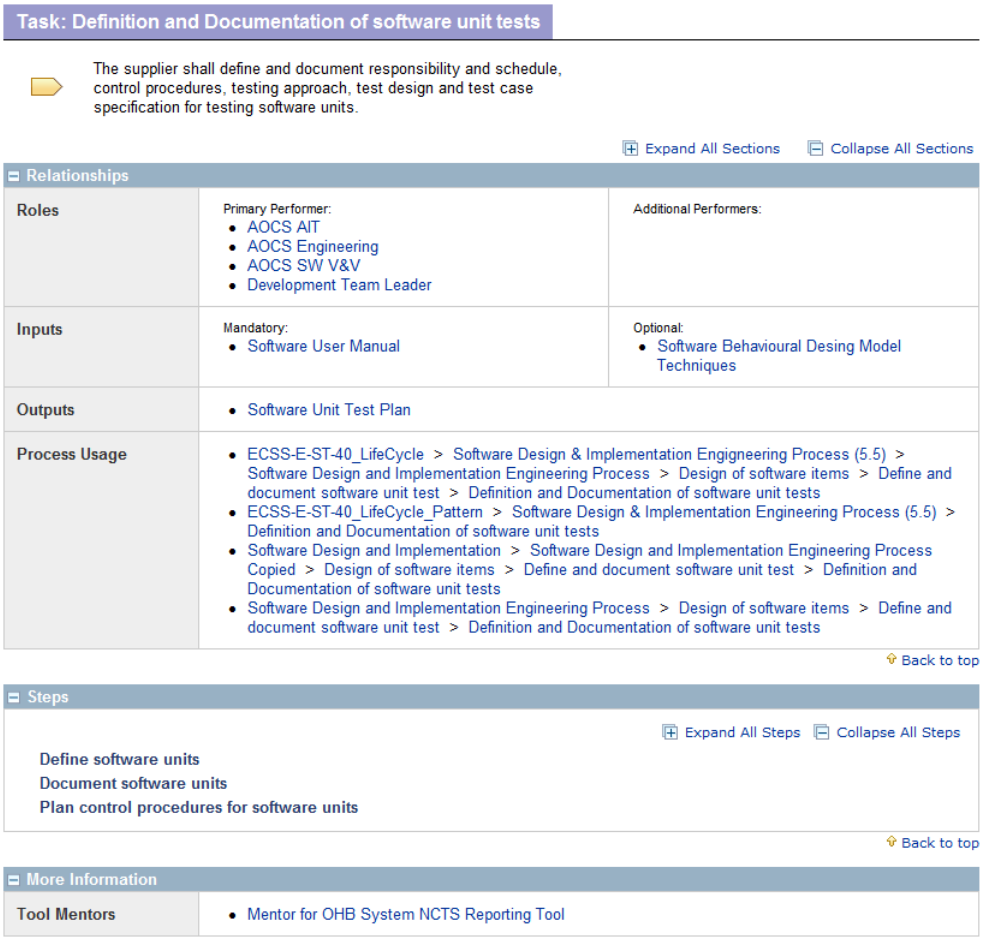


Figure 17 - Preview tab of the form for the modeling of a Task

In certification processes, tools are an important element in the undertaken of an activity. So, in this section, we will explain how tools are modeled in the EPF composer. This is done in two parts, the definition of a Tool Mentor in the Guidance package and a Tool in the Standard Category folder of our plug-in.

Tool mentors represent how to use a specific tool to accomplish some piece of work, in the context of, or independently from, a task or activity. They are modeled in the Guidance folder of a Content Package. So, we right click in the Guidance folder and select **"New -> Tool Mentor"**. Then a form similar to the Task for the modeling of a form will appear in which we can add Description and Guidance to this process element, and to see a preview of it. In order to add a tool mentor to a task, we select the tab Guidance of the task, push the button "Add…" and a dialog will appear (see Figure 18) in which we can select the tool mentor.

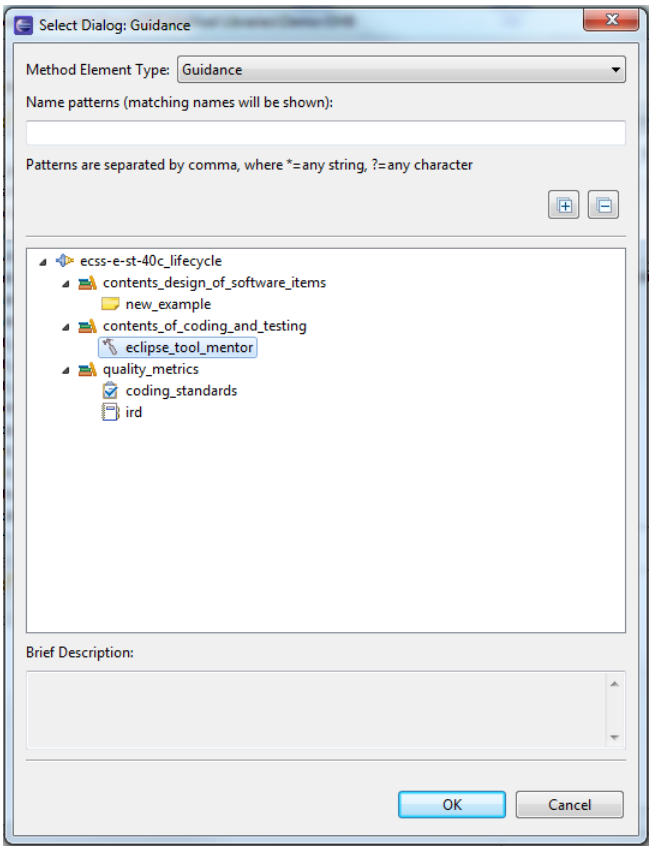


Figure 18 - Dialog for the selection of Guidance elements in the context of a Task.

Finally, tool mentors should be linked to tools. In order to define a tool, we click in the Standard Category package of our plug-in and right-click in the Tools folder and select "**New -> Tool**". Then, a form similar to the form for the tasks will appear (see Figure 19) in which we can add Description, Tool Mentors, Guidance and to have a preview of our modeled tool. In the Tool Mentor tab, we can add our Tool Mentor by using the button "Add…".

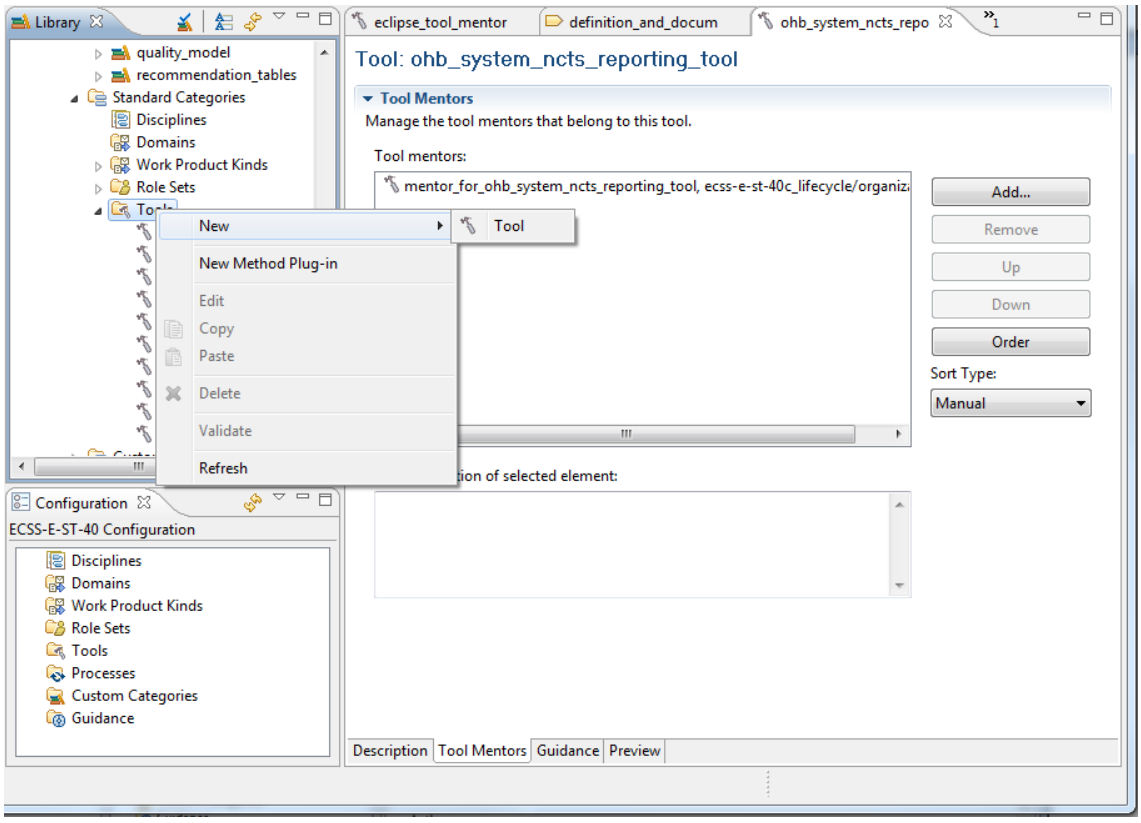


Figure 19 - Creation of a tool and Tool Mentors tab in the EPF composer

4.2 Modeling of processes mandated by standards

The modeling of process mandated by standard is done by the modeling of a Delivery process. Section 4.4 and Section 9 of the EPF manual provide detailed instructions on how to accomplish this task. Previous to the modeling of the delivery process, it is necessary to analyze the standard and derive the process elements (tasks, roles, work products and supporting material) that will be part of our process.

In order to create a new delivery process, we right click in the package Delivery Process of the Processes folder of our plug-in and select **"New-> Delivery Process"** (see Figure 20). Then, a form for the modeling of the process will appear with the Description tab (as in the case of the Task) and other tabs specific for process modeling: Work breakdown Structure, Team allocation, Work Product usage and Consolidated View.

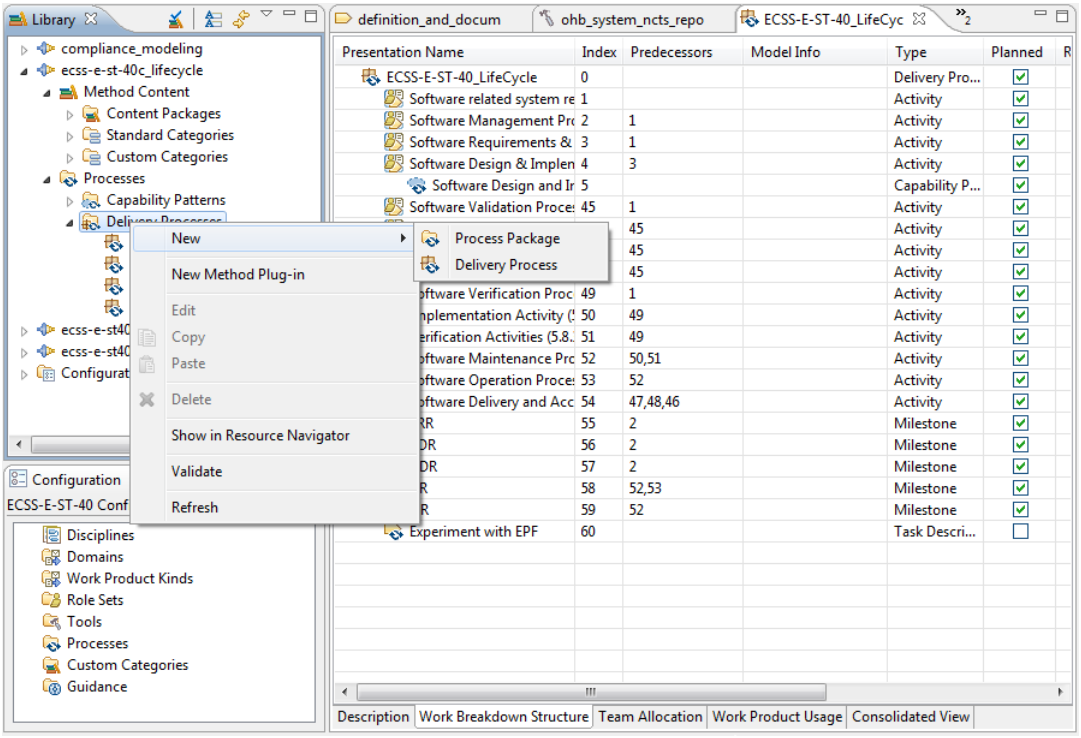


Figure 20 - Creation of a new delivery process in the EPF composer

The Work breakdown Structure is used to describe the structure of the work. The work can be decomposed by using the following elements: Activity, Iteration and Phase. Additionally, we can model Milestones, which are points in the development process in which specific work products are released and we can add tasks by means of Task Descriptors. All these elements can be added in this tab by right clicking in the process (see Figure 21). If we right click in one of this elements and select “Show properties view”, we can add additional information to this process element like supporting material, descriptions or work products delivered in the milestone.

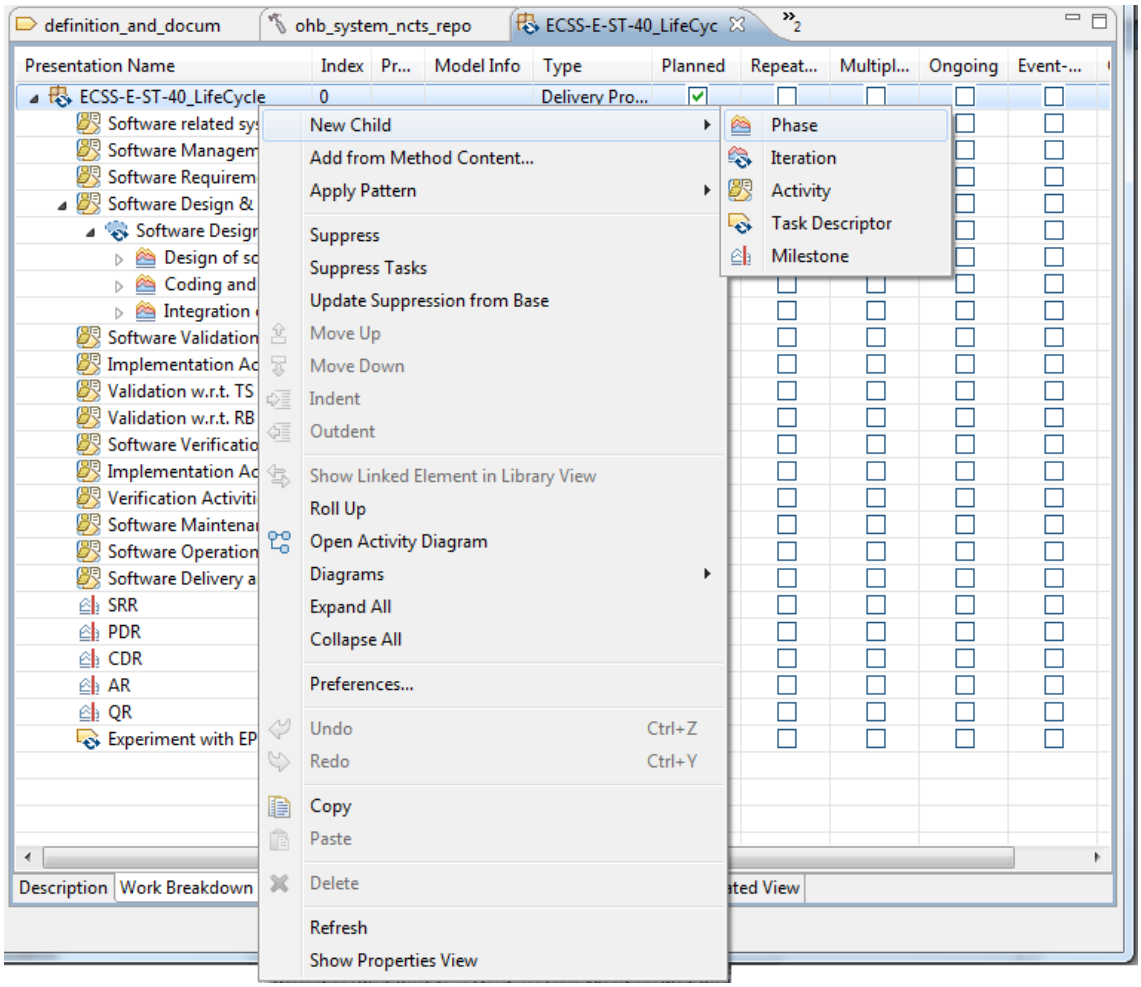


Figure 21 - Modeling of workbreakdown structure of a delivery process

As stated in the introduction of this section, it is possible to re-use process elements defined in the Content Packages to assemble process. This can be done in two ways. The first method is by dragging the specific task from content package and dropping it in the process. The second method is right click in the process and select “New -> Task Descriptor”. Then, we right click in the Task Descriptor that has been created and select “Show Properties View”. In the General tab of the Properties View, we push the button “Link Method Element…” and a dialog appears in which we can select the Task that we want to re-use. Once a task is added to a delivery process, all its related process elements (i.e. work products, roles, …) are added to the process automatically.

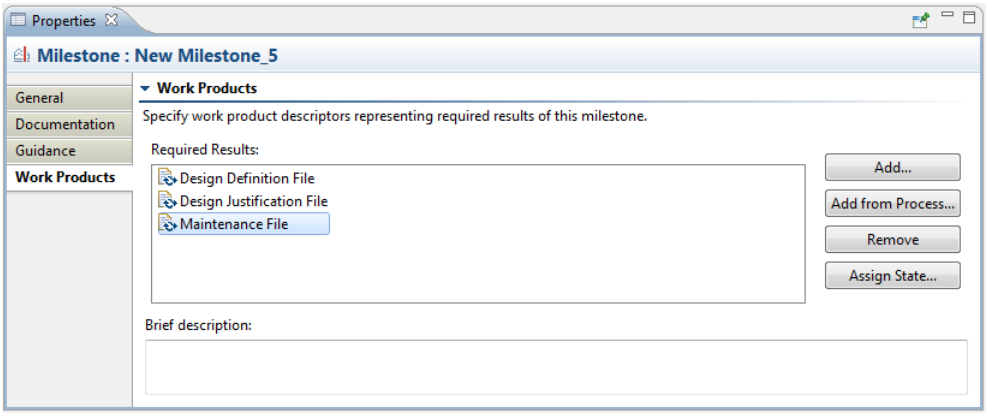


Figure 22 - Properties view in a Milestone.

The Team Allocation tab shows roles that are involved in the different activities of the delivery process. It can be used too, to add roles defined in the content packages to activities using the same procedure described for tasks. Additionally, in this we can define Team Profile as it is described in Section 9.4 of the EPF Manual. The Work Product Usage tab has the same role as the Team Profile tab but specific for Work Products. Finally, in the Consolidated View tab, we can see how Task, Roles and Work Products are involved in the Activities of the Process (see Figure 23).

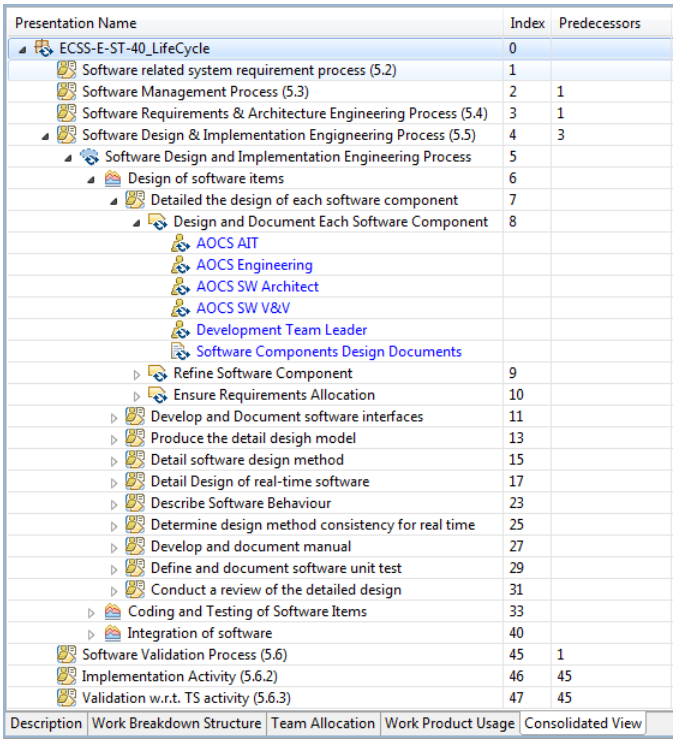


Figure 23 - Consolidated View of the Delivery Process ECSS-E-ST-40_LifeCycle

EPF offers the possibility of modeling a process using UML-Style Activity Diagrams as it is described in Section 4.4.5 of the EPF Manual. To open this view, in the Work Breakdown Structure tab of the delivery process, we right click in the process and select “Open Activity Diagram”. The main purpose of this view is to model how activities are done (order, parallelism) in the delivery process.

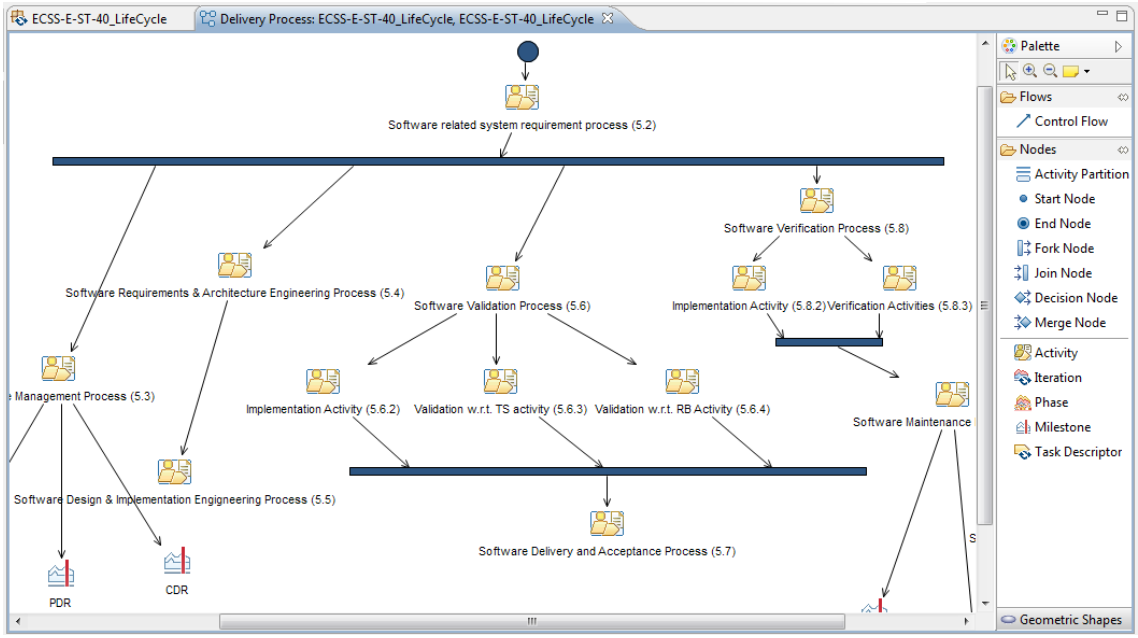


Figure 24 - Activity diagram of a delivery process

4.3 Modeling of reusable process patterns

Capability patterns are a special type of process for a key area of interest or such as a discipline or best practice. The can be used as building blocks to assemble larger capability patterns and delivery processes. Sections 4.4 and 9 provide detailed information about how to define and use capability patterns in the EPF composer. The modeling of the work that is contained in a Capability Pattern can be done using the same method followed for the Delivery Process. So, we overview how to use a capability pattern in this section.

In order to add a capability pattern to a process or other capability pattern, we right click in the process or activity in which we want to add the pattern and select “Apply Pattern”. As it is depicted in Figure 25, we have three options: Copy, Extend and Deep Copy. Specific details of the differences between these ways of application of the patterns are described in Section 9.7. In a few words, the main difference between Copy (or Deep Copy) and Extend is the type of relationship between the original capability pattern and the pattern which is placed in the process. When we select any of these options, a dialog to select the capability pattern appears.

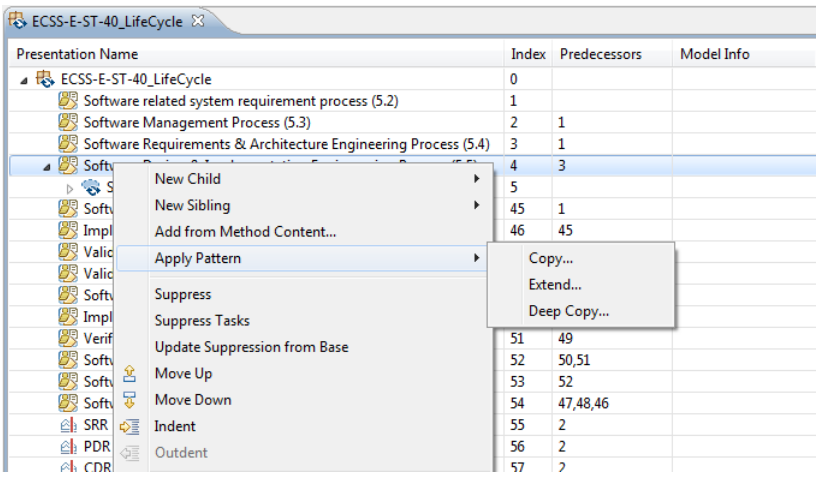


Figure 25 - Adding a capability pattern in a delivery process.

When a capability pattern is copied (or deep copied) in a delivery process or other capability pattern, all the information of the process contained in the pattern (i.e. activities, roles, milestones) is automatically added. We can modify this copied capability pattern to meet requirements of the delivery process. So, we can add tasks, modify a role or rename an activity, just to mention a few. These modifications do not affect to the original capability pattern. On the other hand, if we apply the pattern using the Extend…, a link is set between the original capability pattern and the extended capability pattern (this is highlighted in EPF using green letters as it is depicted in Figure 26). So, we cannot modified this extended capability pattern and additionally, any modification in the original capability pattern will be propagated to its extensions. This is particularly useful to update a family of delivery processes for a domain.

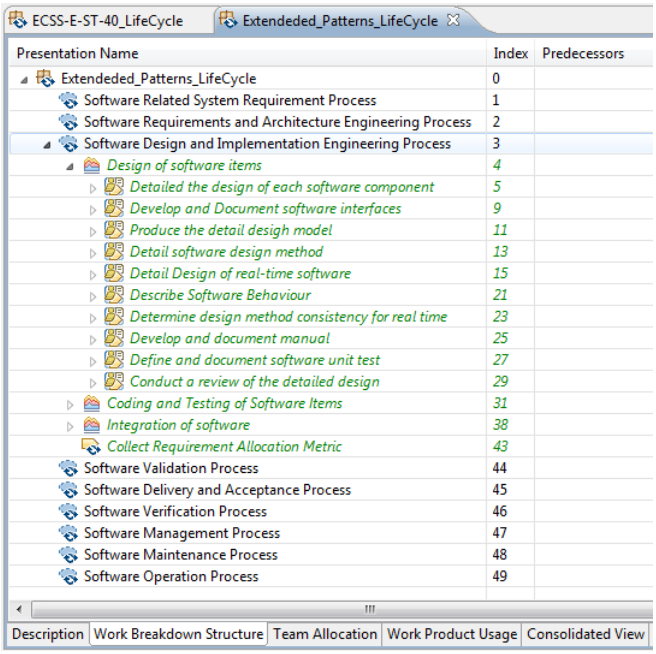


Figure 26 - Extended Capability Pattern

5 Standards Modeling

As described in Section 2.1, users can use the Reference Framework Editor to model Standards (IEC 61508, ISO 26262, DO-178C, EN 50126, and the like), any Regulations (either as additional Requirements or model elements in a given model representing a Standard or a new Reference Framework). Each Reference Framework model can be also mapped to other Reference Framework models by using the concept of Equivalence Map.

5.1 Create Reference Framework model

In order to create a new Reference Framework model, follow the next steps:

- From the File menu, choose New -> Other …

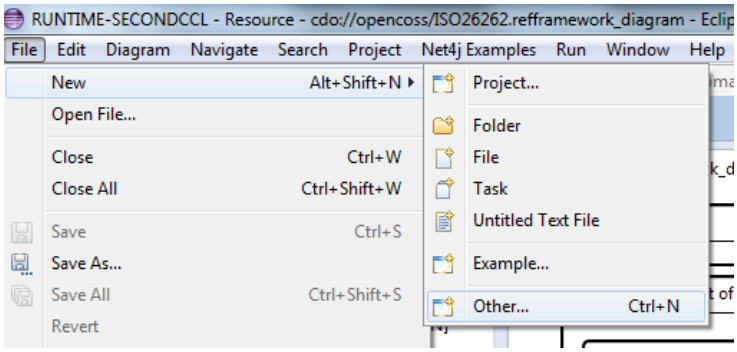


Figure 27 - New Reference Framework model

- In the Wizard dialog, open the AMASS category, and the select Reffframework Diagram, and press the Next button.

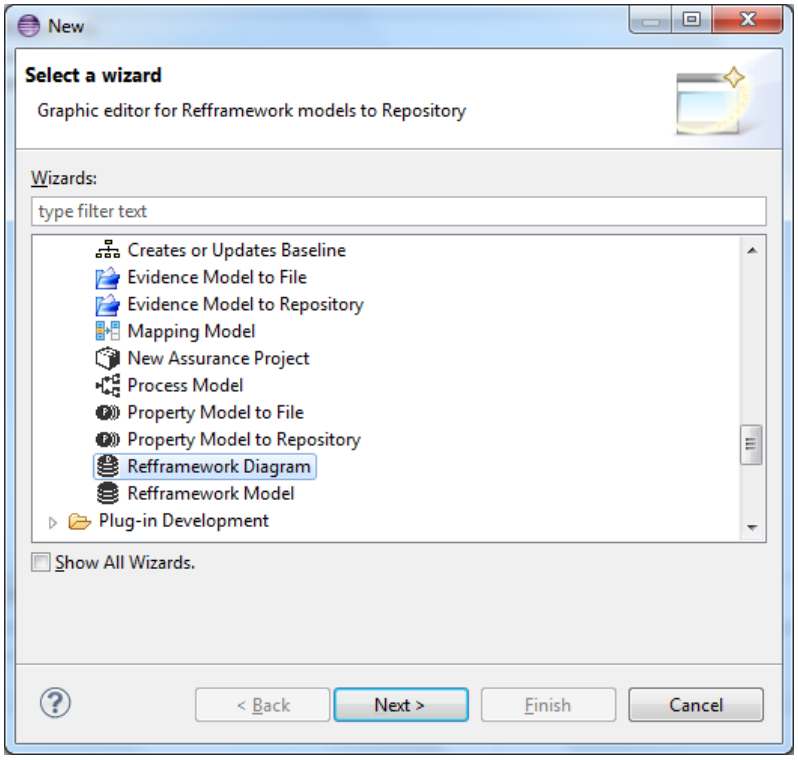


Figure 28 - Wizard Reference Framework model

- In the New Reframework Diagram dialog, select or enter the parent folder, the name of the diagram to be created, and press the Next button.

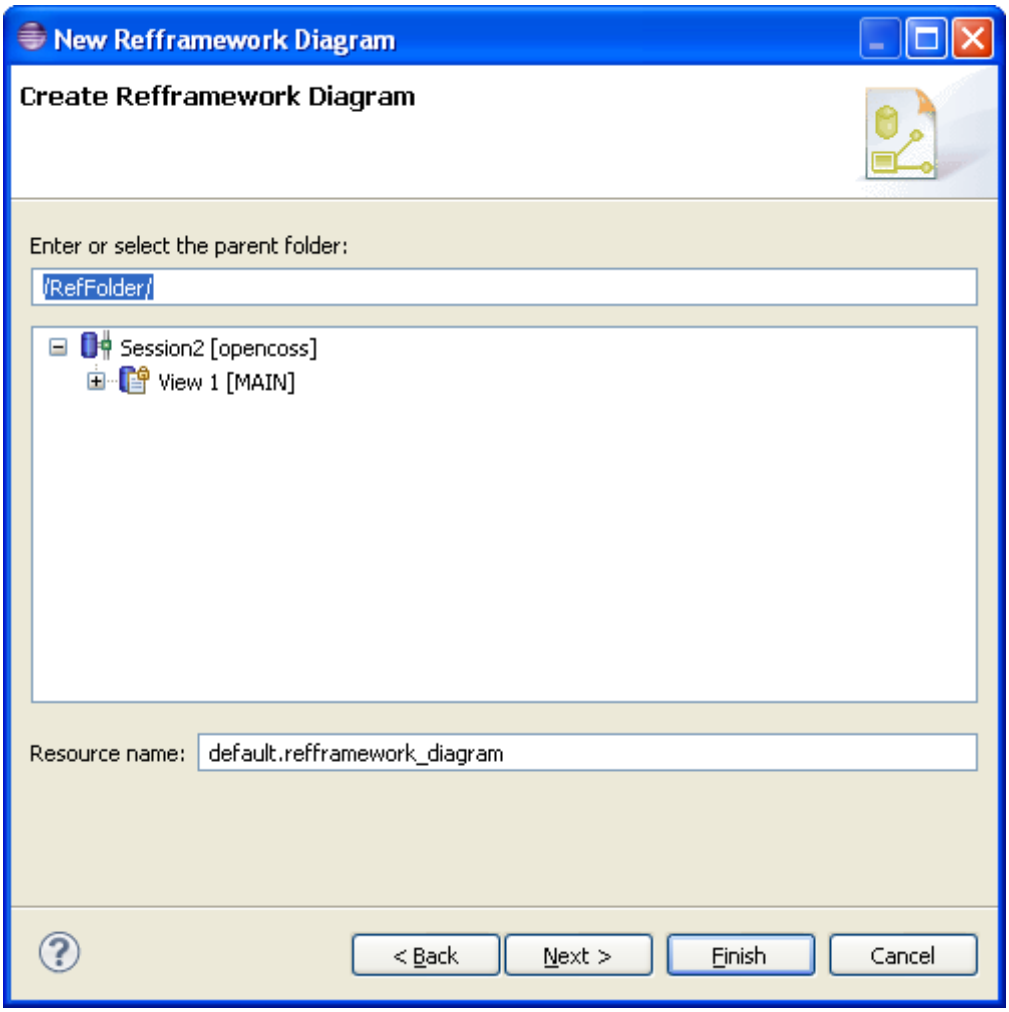


Figure 29 - New Reframework Diagram

- In the New Reframework Domain model page, select or enter the same name as in the previous step as parent folder, enter the name of the diagram to be created, and press the Finish button.

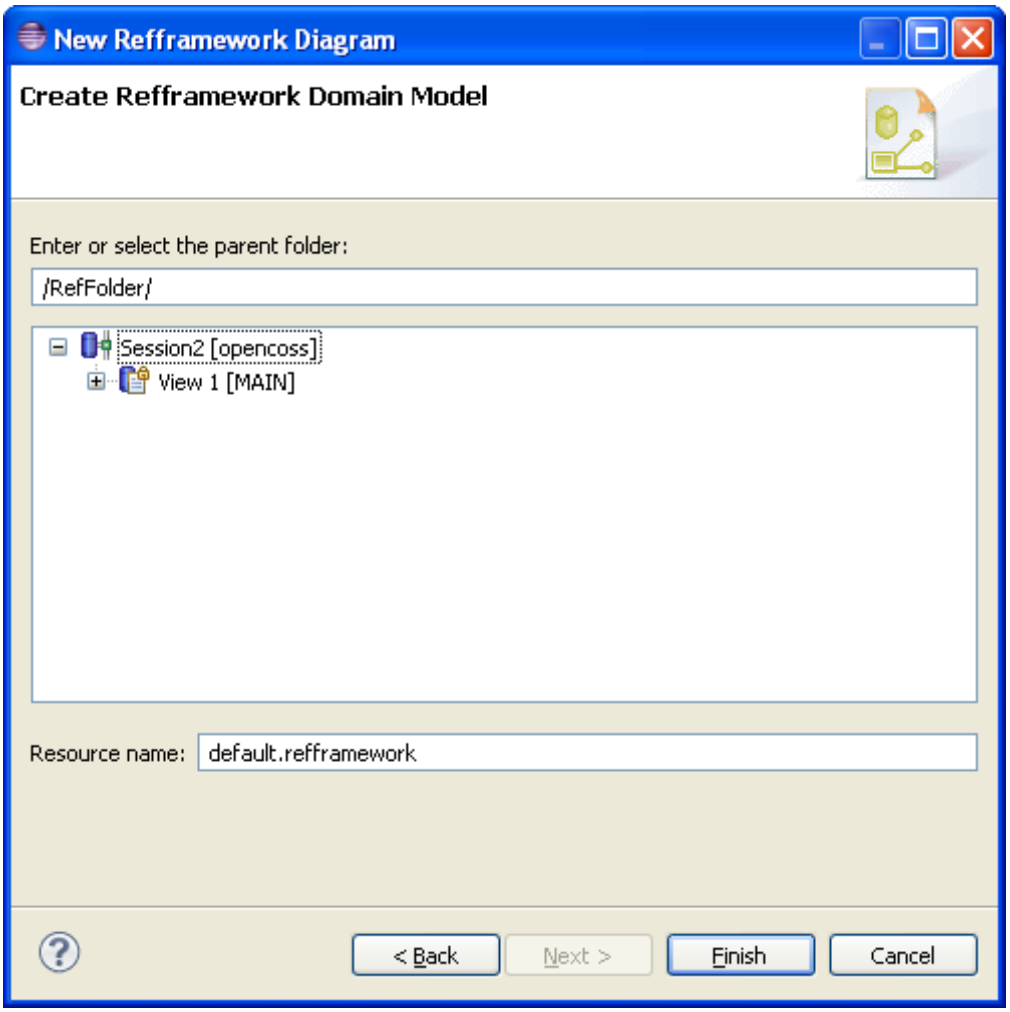


Figure 30 - New Reframework Domain Model

5.2 How to edit a Reference Framework model

After complete the Reffframework Diagram creation wizard, the perspective of the tool will be opened composed by five views:

1. The **Repository Explorer** shows the contents of the repository.
2. The **Outline** shows the elements of the model and permits its edition.
3. The **Diagram Editor** permits the graphical modelling of a subset of concepts of the Reference Framework.
4. The **Palette** is a toolbox with the concepts of the model and the connections between them to add to the diagram.
5. The **Properties** to edit the properties of the element of the model selected.

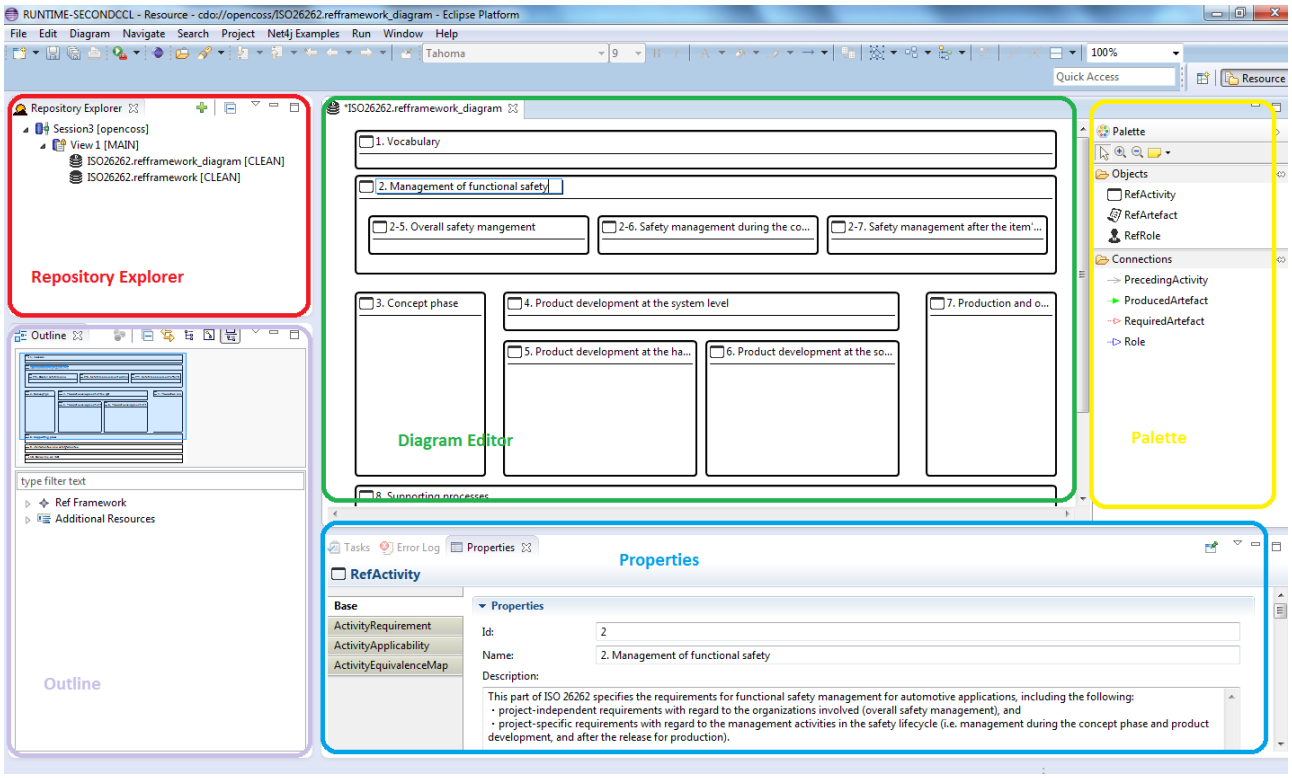


Figure 31 - Reffframework editor perspective

5.2.1 Add concepts to the diagram

To add concepts to the diagram, left click in a category Object of the palette and move the cursor over the diagram zone. This cursor appears if it is possible to add this object in the target diagram location according to the modelling rules (Reference Framework metamodel), if not this other will appear.
A figure representing the concept will be displayed in the diagram.

5.2.2 Add links between concepts

To add link between concepts, select it from the Connections category of the palette. This cursor appears if this object can be the origin of the connection, according to the modelling rules (Reference

Framework metamodel), if not this other will appear. Maintain the left mouse clicked, the cursor will become , and move to the destination object, the same icons will appear if the destination is correct or not.

5.2.3 Edit properties

Some model elements from Reference Framework cannot be edited graphically (RefRequirements, RefApplicability tables, among others). These model elements can be edited by using the Properties view.

If the properties view is not visible, you can open it by using the contextual menu of the figures "Show Properties View" of the figures.

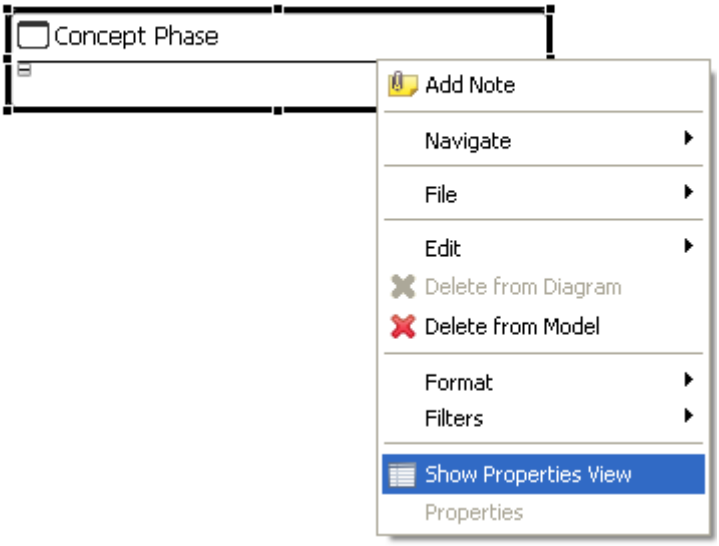


Figure 32 - Show properties view

5.2.4 Create multi-diagrams from a Reference Framework model

The tool allows managing different views of a model through a set of diagrams. Once a model is available, a new diagram view can be created and special edition functionalities are available as follows:

1. Thanks to the Outline view, it is possible to drag and drop concepts from the model to the diagram.
2. Once a concept has been selected, it can be hidden through the "Delete from diagram" option available in the contextual menu. This option does not delete the concept from the model.
3. Once a concept has been selected, it can be deleted through the "Delete from model" option available in the contextual menu. This option delete the concept from the model permanently. If this deleted concept is visible in another diagram files, this concepts will be shown with a cross icon in the upper right corner to show that it does not exist anymore.

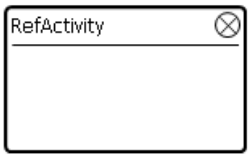


Figure 33 - Deleted concept shown in a diagram

Once a model is available, a new diagram view can be created following the procedure below.

Select "New" → "Other…" from the File menu of eclipse

Select "Reframework Diagram" in the Opencert category and click the "Next" button.

Select the folder to store the new diagram. Then type the name of the diagram to be created and click the "Next" button.

Type the name of the model created previously and click the "Finish" button.

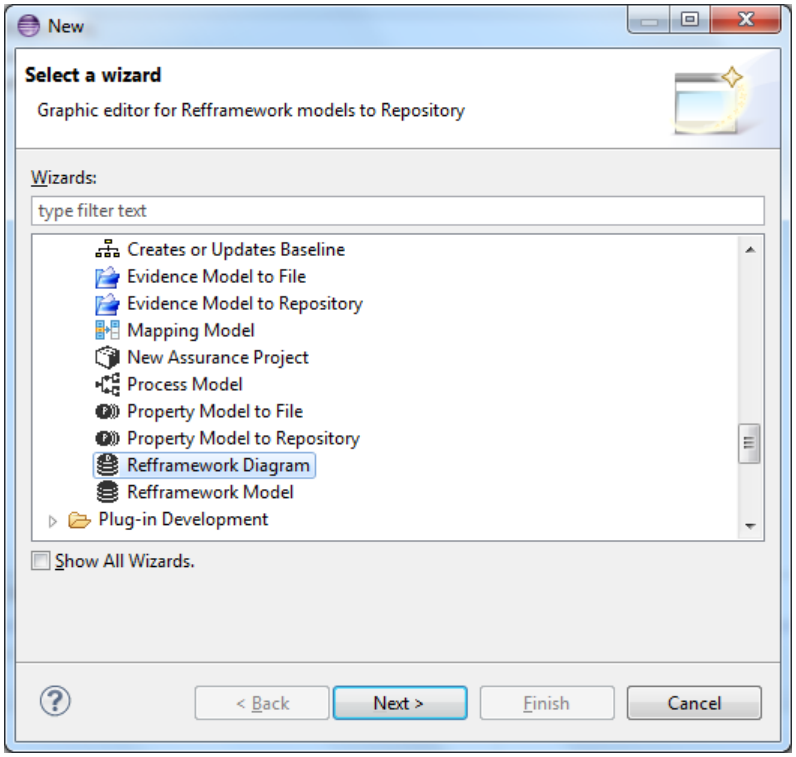


Figure 34 - Reframework Diagram wizard I

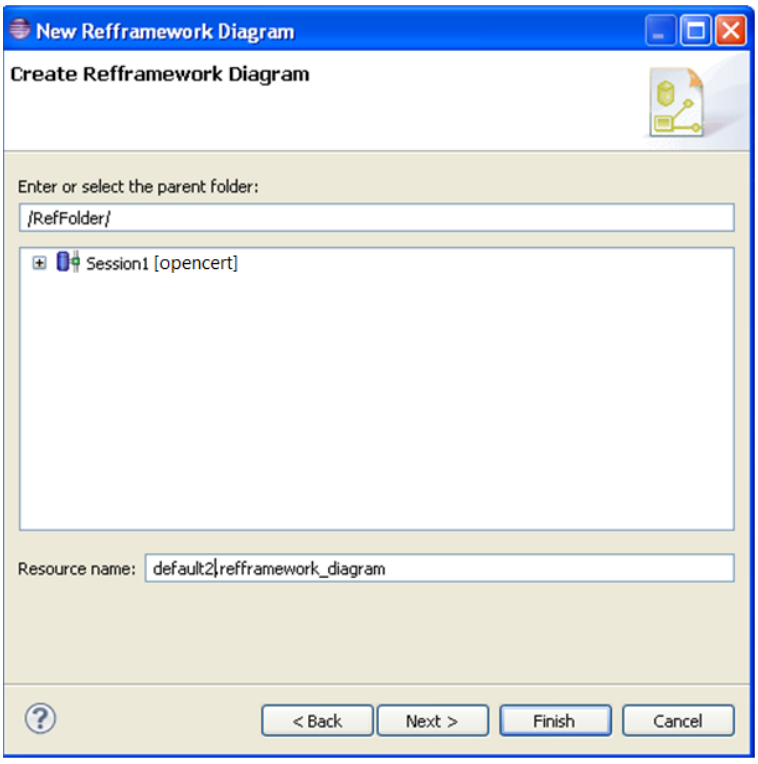


Figure 35 - Reframework Diagram wizard II

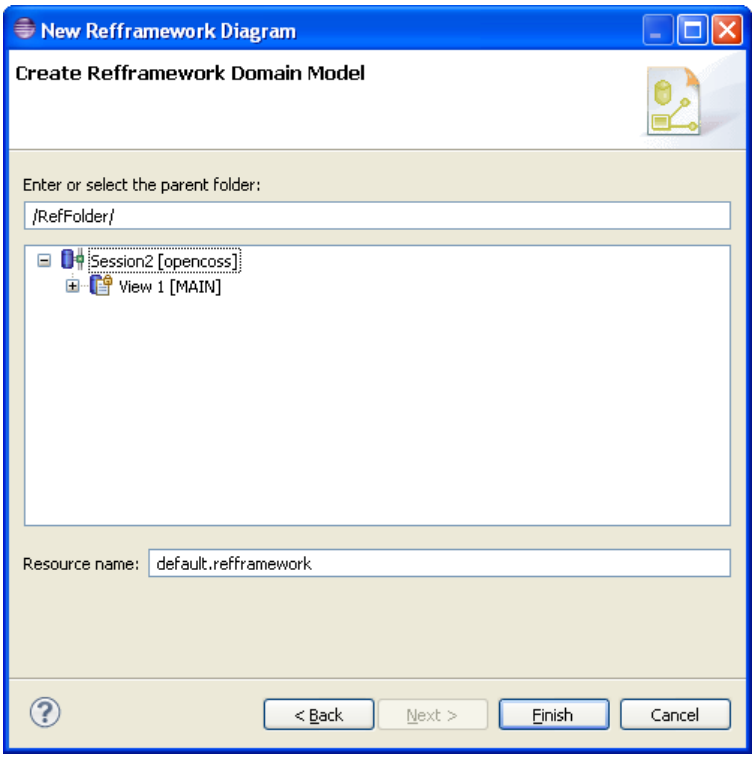


Figure 36 - Reffframework Diagram wizard III

After that, the diagram is ready for edition.

5.2.5 Non graphical editor

Alternatively to the graphical Editor, the Reference Framework model can be edited by using a purely Form Editor. To do so, open the file created together with the Diagram file (extension: **xxx.reffframework**).

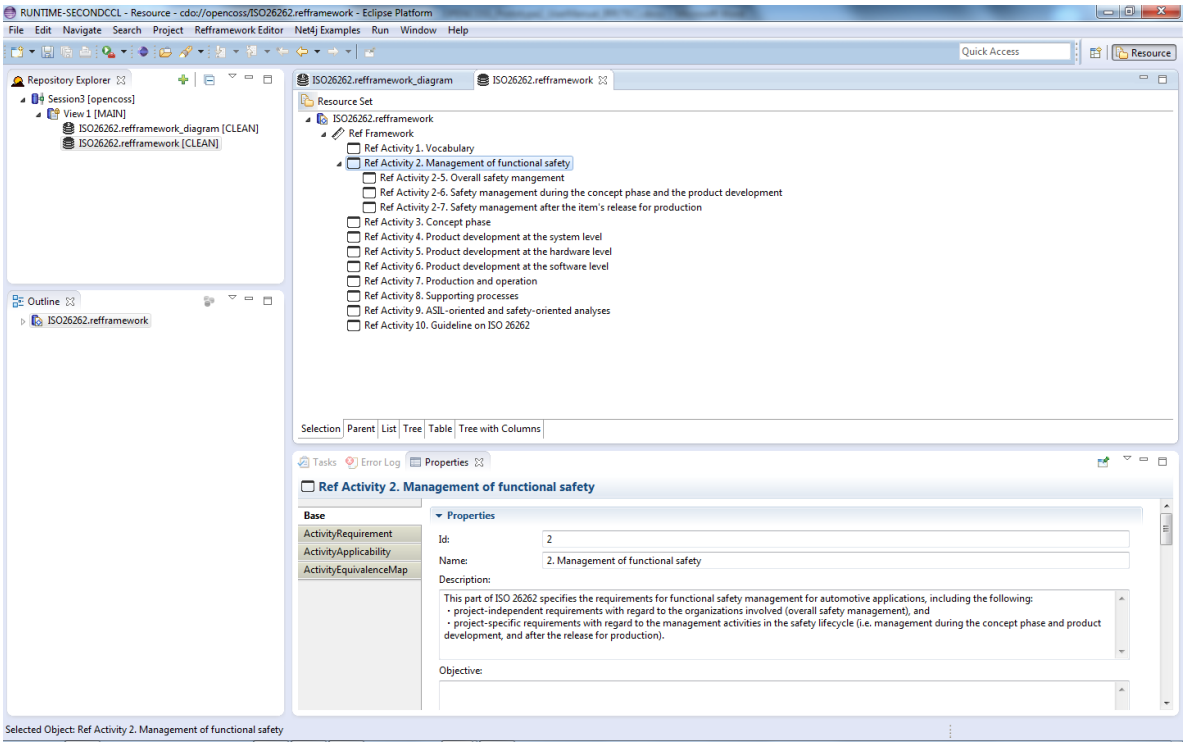


Figure 37 - Model tree editor

It is also possible to use the Outline view to create new model elements, as shown below.

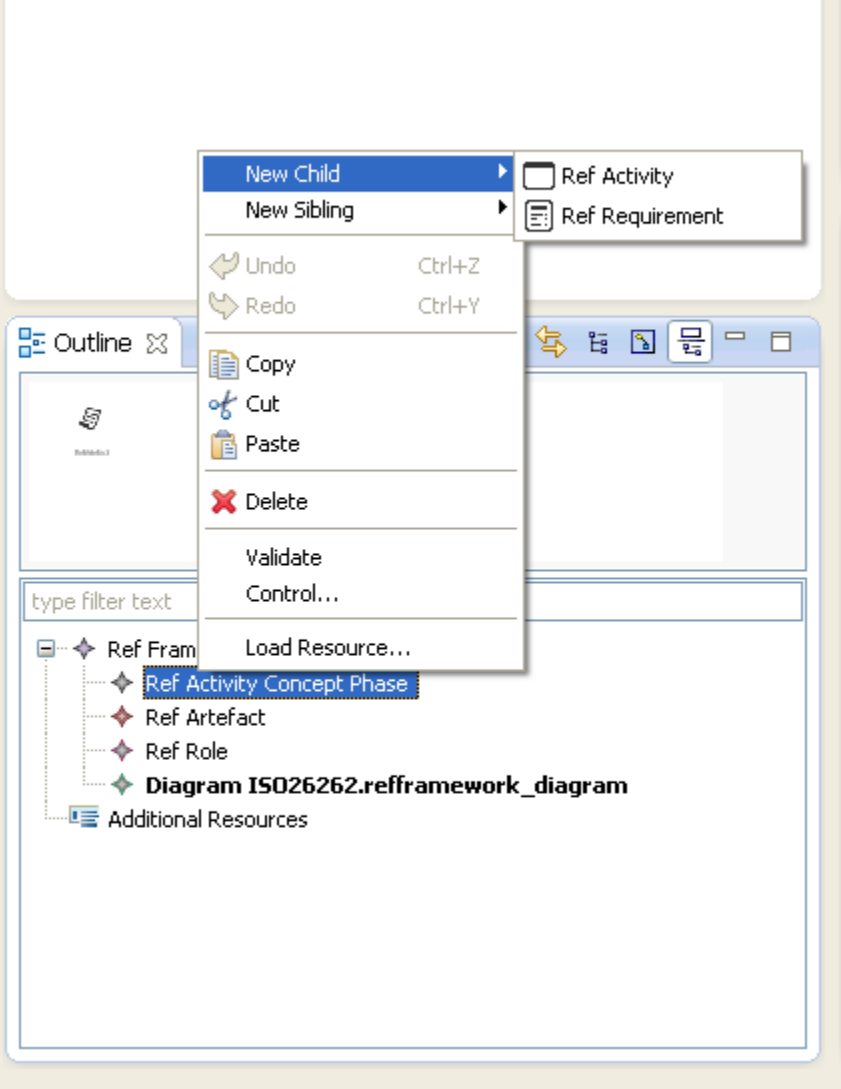


Figure 38 - Edit model from Outline

5.3 Creating Equivalence Maps

It's possible to create Equivalence Maps in two ways:

- One way, using the editor,
- Another way, using a tailored functionality for it.

5.3.1 Equivalence Map using the editor.

To create Equivalence Maps using the editor, it's necessary to load two CDO resources: the reference framework model (.reframework) and the mapping model (.mapping).

So, press the editing window and select "Load Resource" in the context menu.

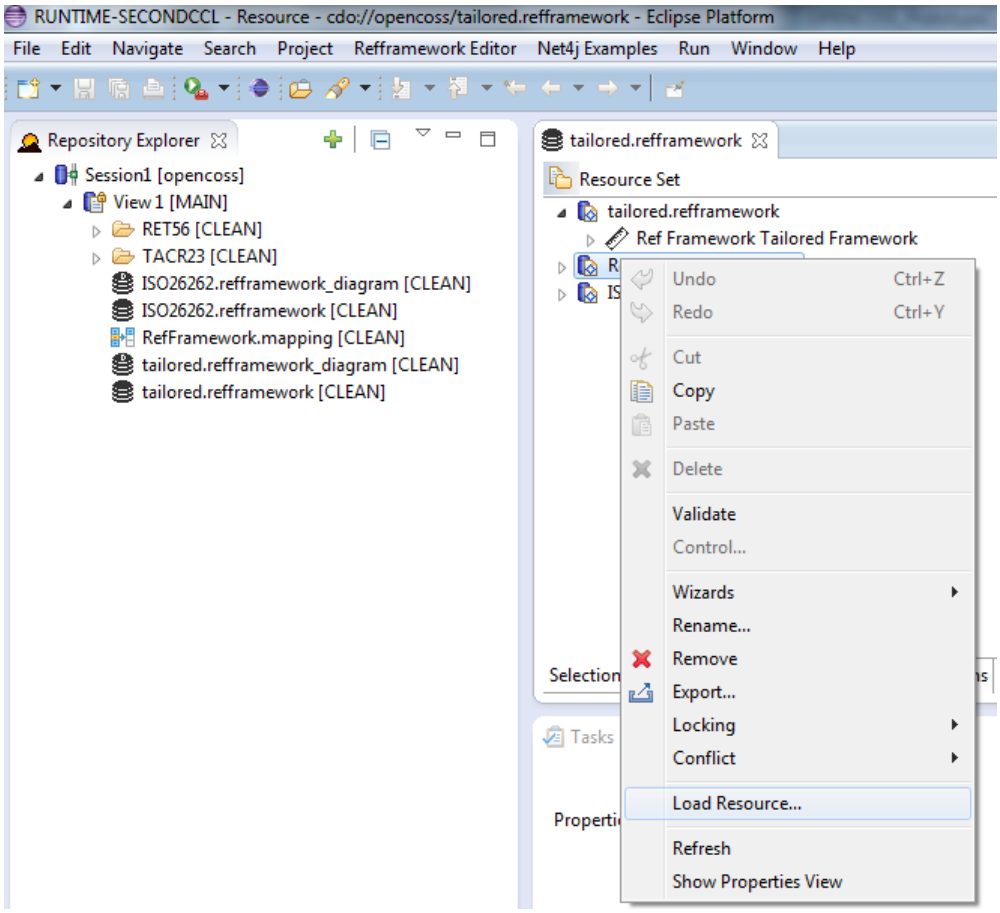


Figure 39 - Load Resource I

Then select the reframework model and mapping model using the “Browse Repository” button to obtain the URI of any model stored in the repository.

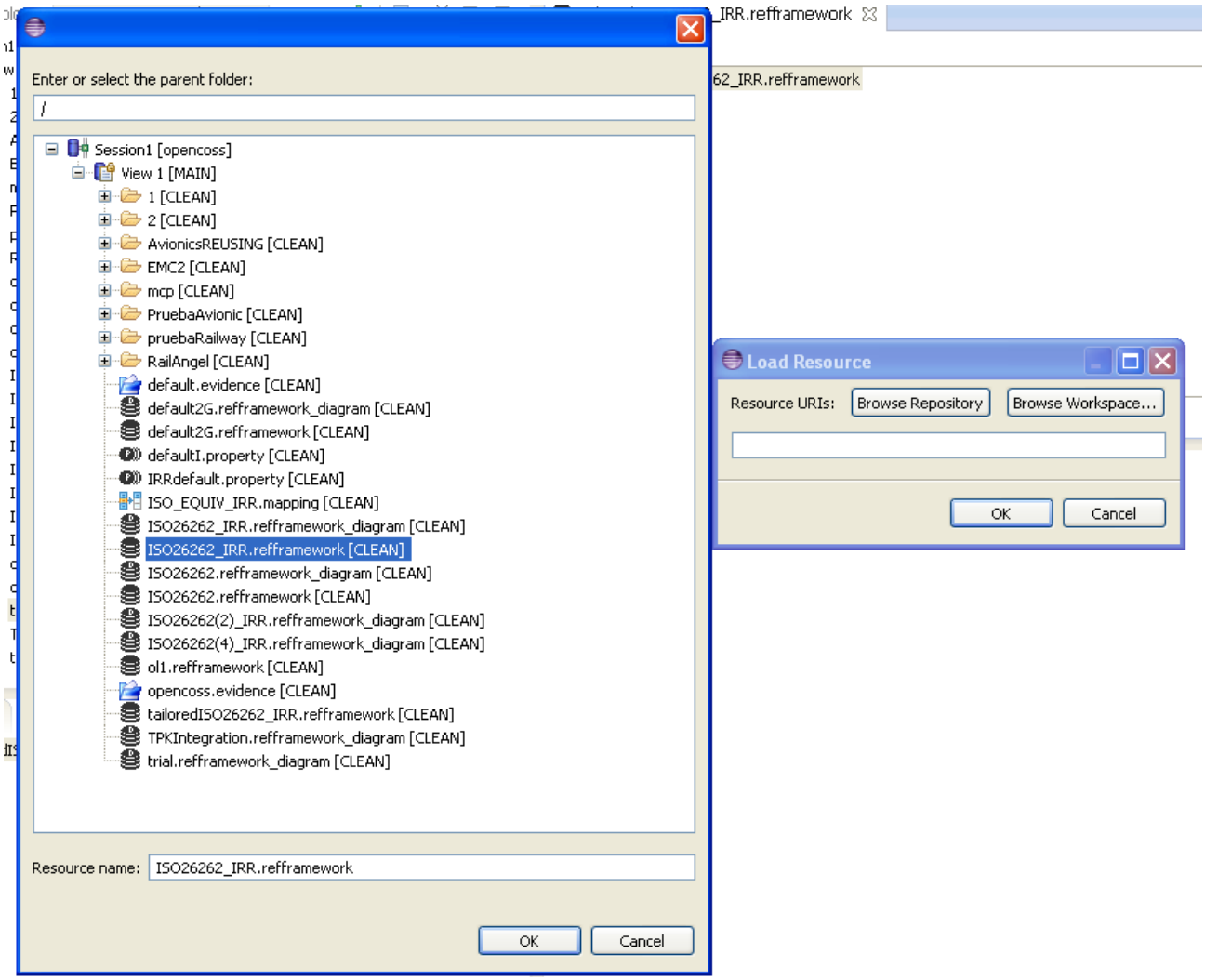


Figure 40 - Load Resource Reference Framework II

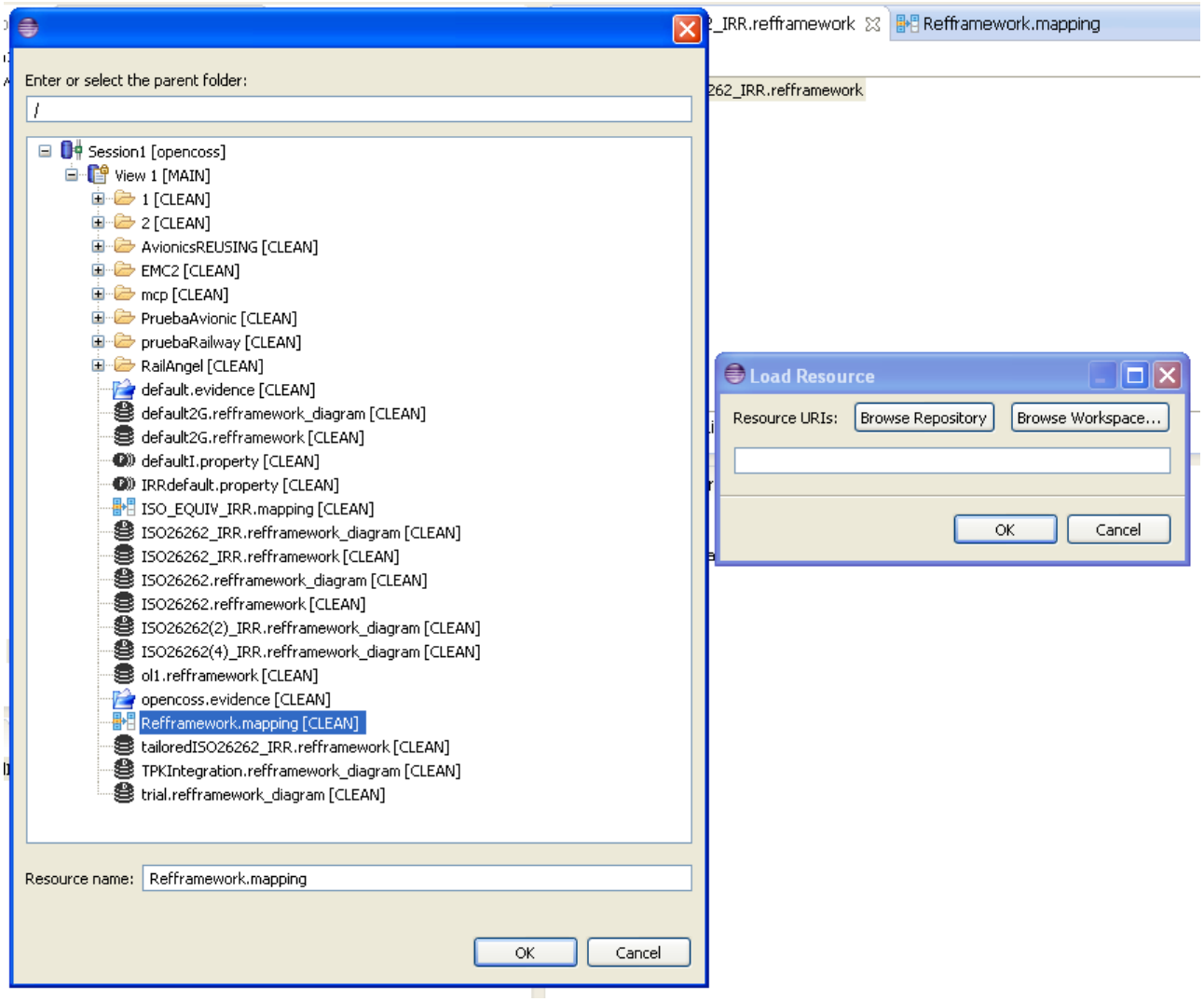


Figure 41 - Load Resource Map Group III

It's possible to create equivalence maps for activities, artefacts, requirements, roles and techniques. Then, first select the object in the tree and after click on the tab "ActivityEquivalence Map" and press the button "Add"

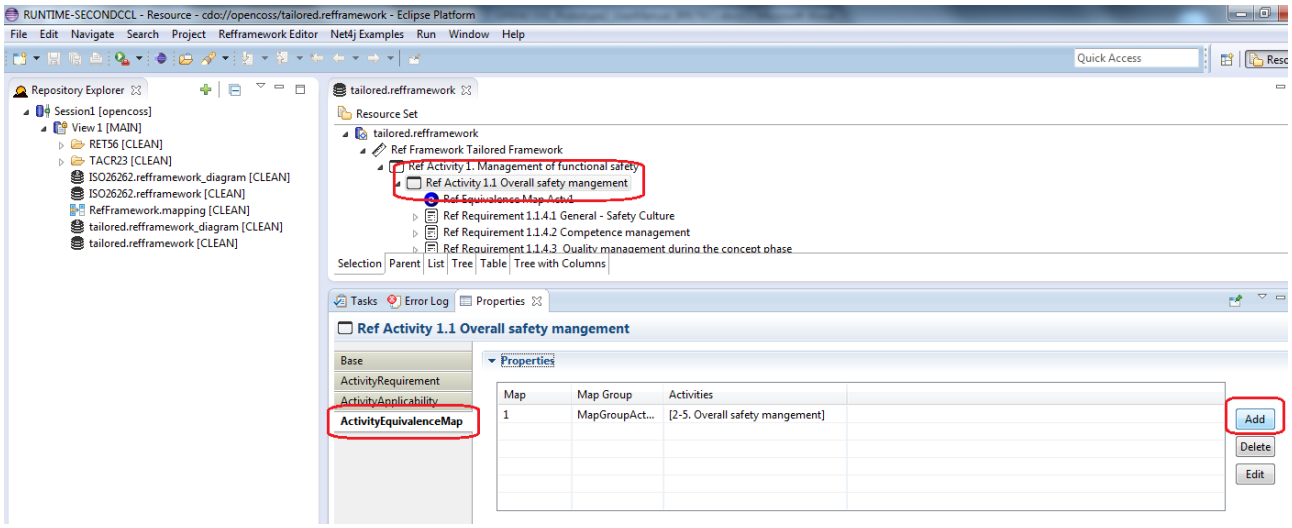


Figure 42 - Activity Equivalence Map

Finally, enter the information requested:

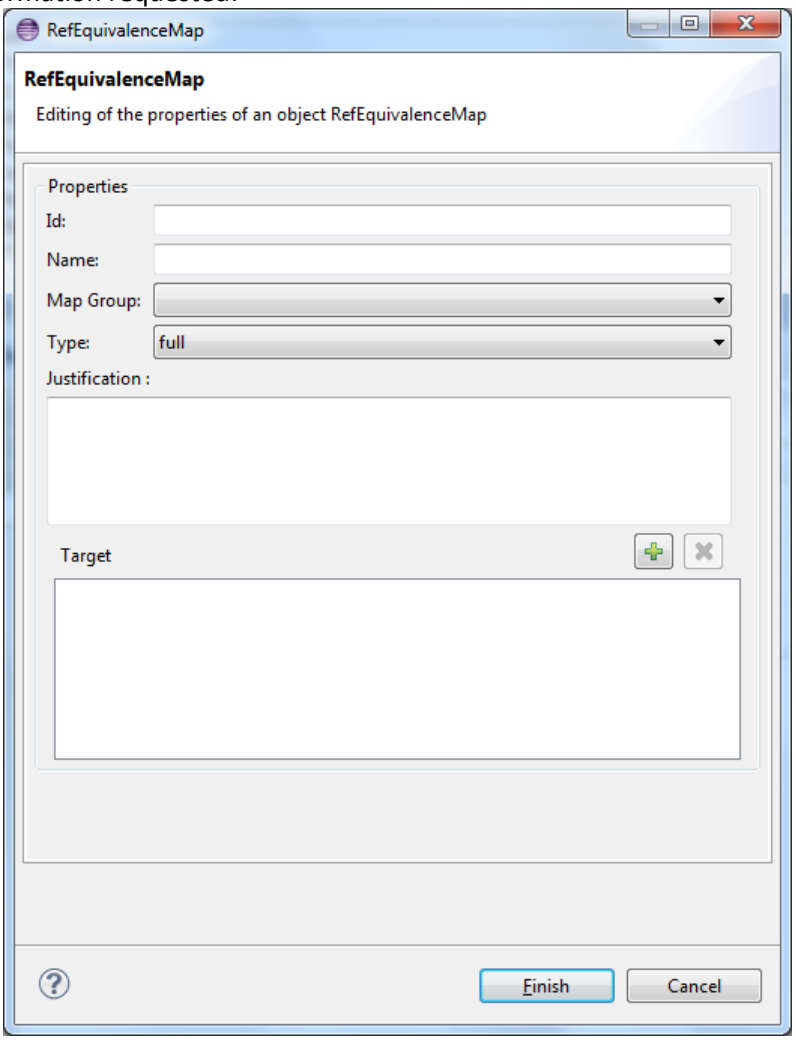


Figure 43 - Equivalence Map

If the user adds as target of the equivalence an element of the source reframework, this target element will be considered as postCondition. The postConditions are mandatory extra activities, not included in the standard, that must be performed in case of reusing the target element from one assurance project based in the target reframework in another assurance project based in the source reframework using the Cross-Domain functionality that will be explained in the section put section reference.

5.3.2 Equivalence Map using a tailored functionality.

To create Equivalence Maps using the tailored functionality, first of all, it's necessary to press the button "Mapping Set" on the properties form of the reference framework using the tree view editor (not available using the diagram editor). This window automatically saves the mappings when checking or unchecking elements of the target reframework tree.

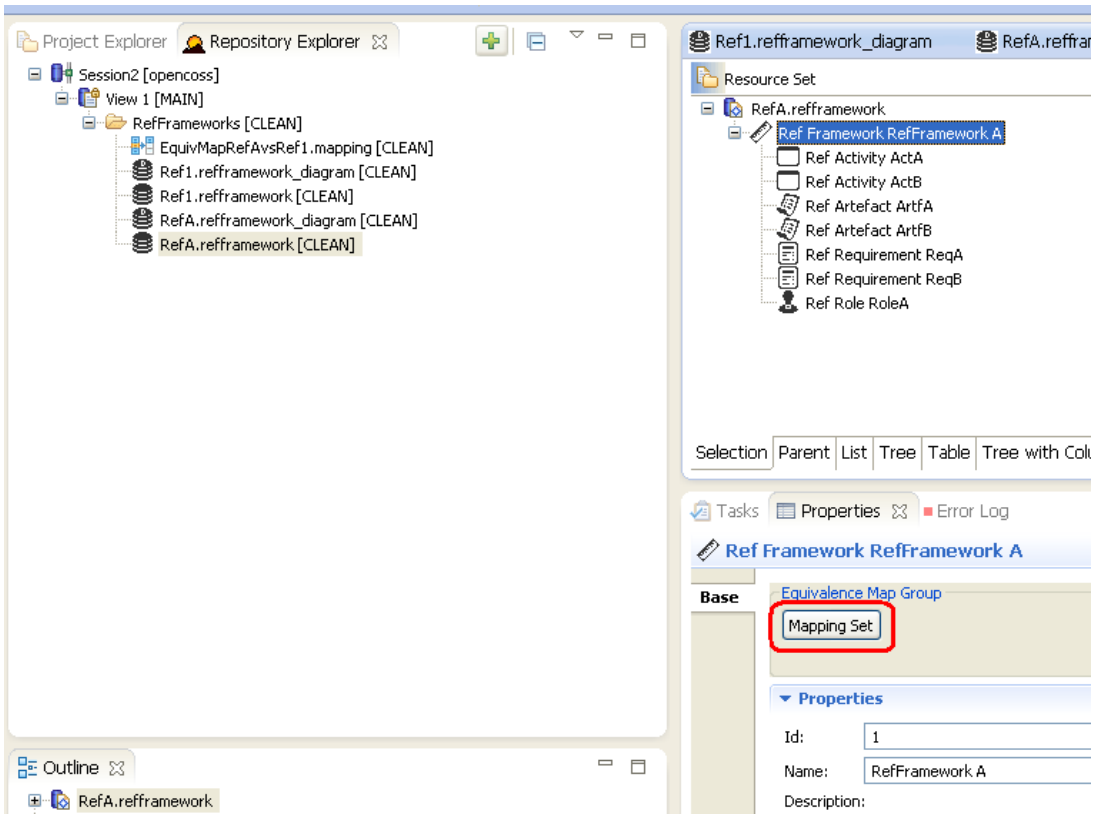


Figure 44 - How to create Equivalence Map.

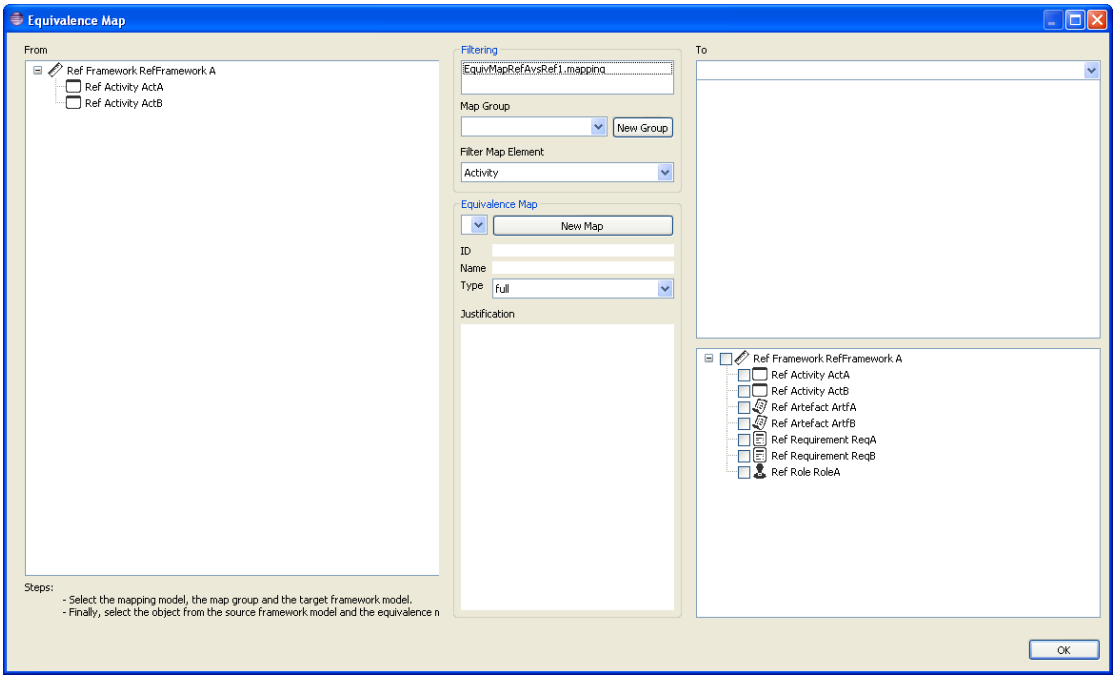


Figure 45 - Equivalence Map form.

The Equivalence Map form is organized in three zones:

- The *left zone* shows the actual reference framework, and it loads the type of elements for which we want to make the equivalence maps. For default, activities.
- The *middle zone* allows to make different filters like:

- *Filter Mapping Model* lists all the mapping models stored in the database, and it will be necessary to select one of them and one group model. It's also possible to create a new map group pressing the button "New group".
- *Filter Map Element*. It's possible to create equivalence maps for activities, artefacts, requirements, roles and techniques. When these filter change, also it changes the information showed by the reference framework. For example:
  - If the filter "Requirement" is selected only the requirements of both reframeworks will be shown:

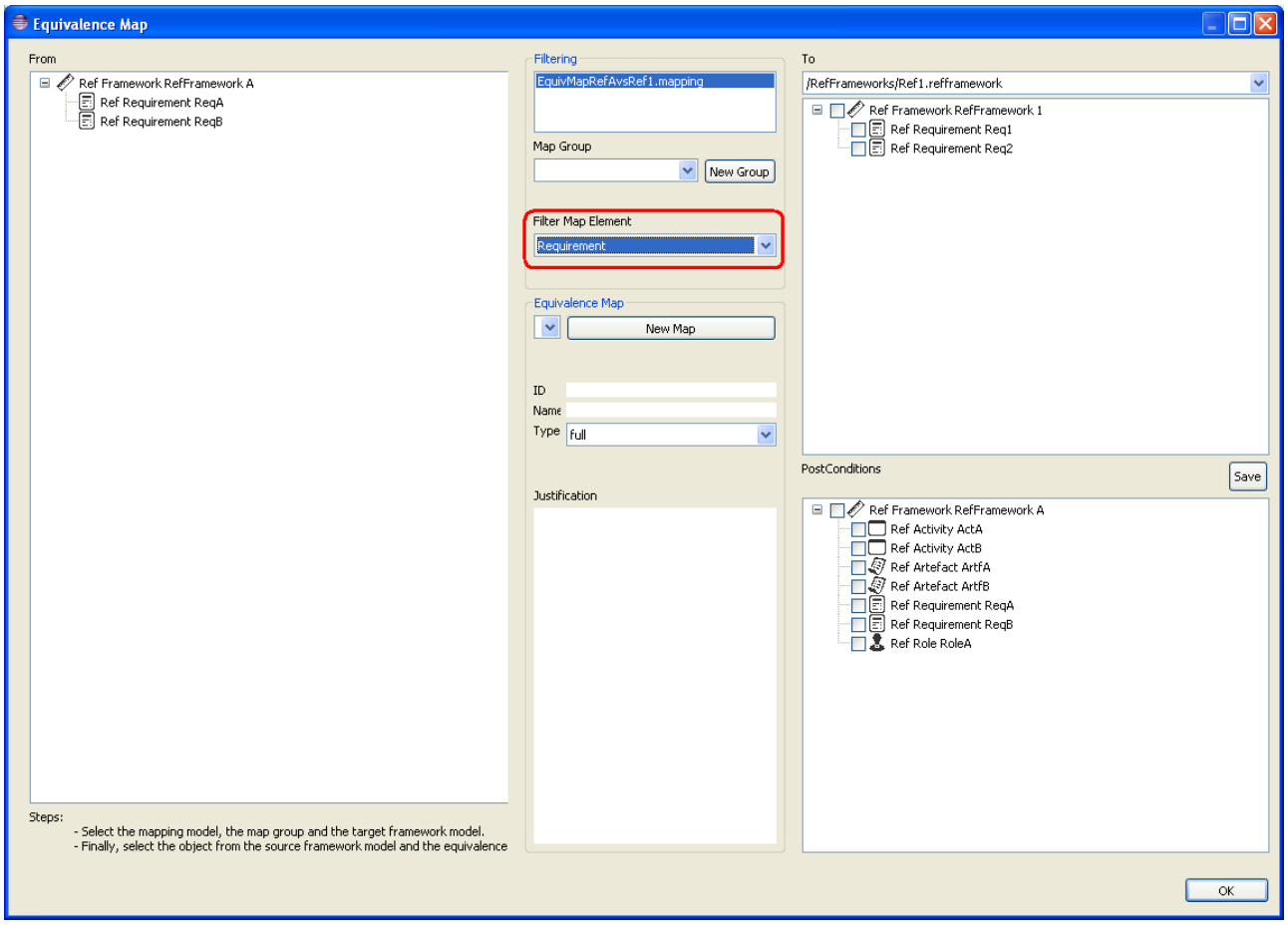


Figure 46 - Equivalence Map, select map element (I)

- *Filter Equivalence Map*. This filter allows making different equivalence maps for the same from reframework element.
- The mapping information must also be introduced in the middle part by the user; this information is the ID, the name, the type and a justification text.

- The *right zone* shows two lists and a combo box.
  - *The combo box*. It shows all the database reframeworks to select the reference framework that will be the target of the equivalence map to create.
  - The upper list loads the elements, according to the filter selected, of the reframework chosen in the combo box that will be the target of the equivalence map to create.

The lower list displays the full content (not filtered) of the source reframework that will be postConditions in case of reusing. The postConditions are mandatory extra activities, not included in the standard, that must be performed in case of reusing the target element from one assurance project based in the target

reframework in another assurance project based in the source reframework using the Cross-Domain functionality that will be explained in the section 6.5.

If the user double-clicks any element of this list, the source reframework could be modified to create new element to be used as postconditions (the save button must be pressed to save the changes).

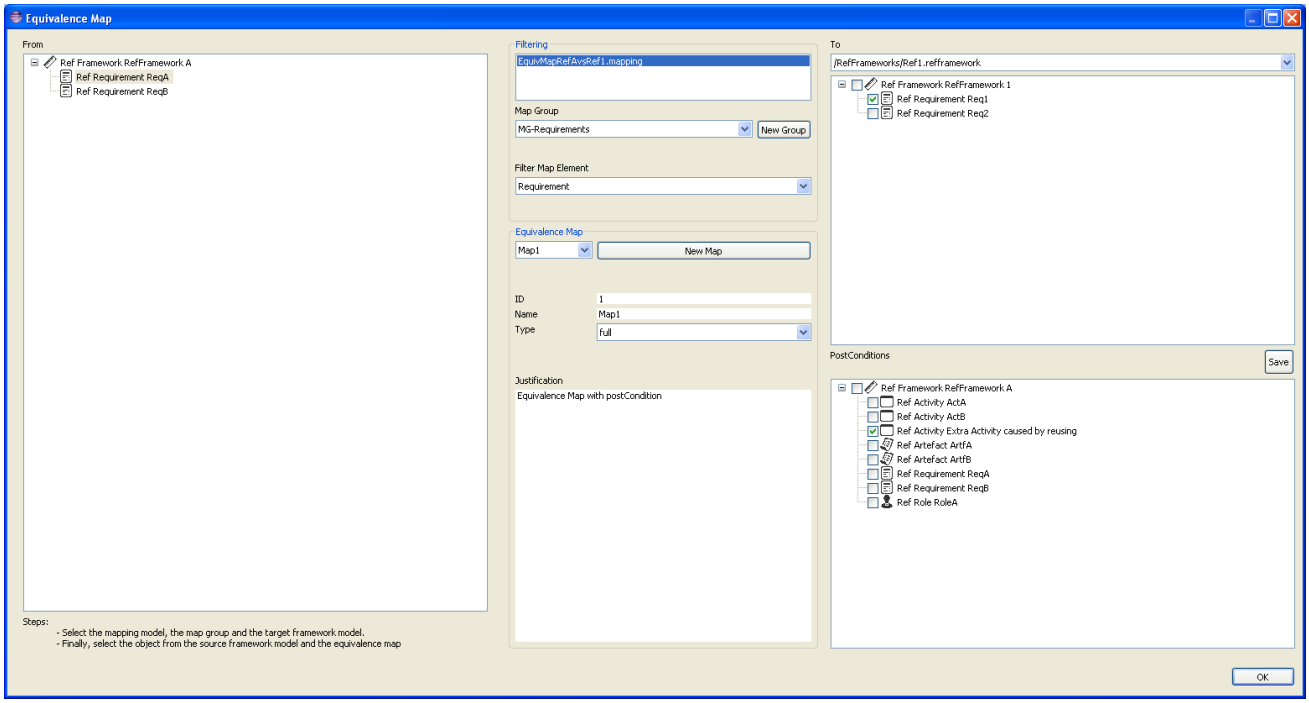


Figure 47 - Equivalence Map with Postcondition.

For making an equivalence map, this window saves automatically the mapping information, follow the next steps:

1. Select a mapping model and a map group (or create it if needed).
2. Select the target reference framework.
3. Select the filter map element.
4. Select the element from the source reference framework.
5. Select or create the equivalence map and introduce the mapping information (ID, name, type and justification).
6. Check or uncheck the element from the target reference framework.
7. Create the postconditions if needed and check or uncheck the postconditions elements.

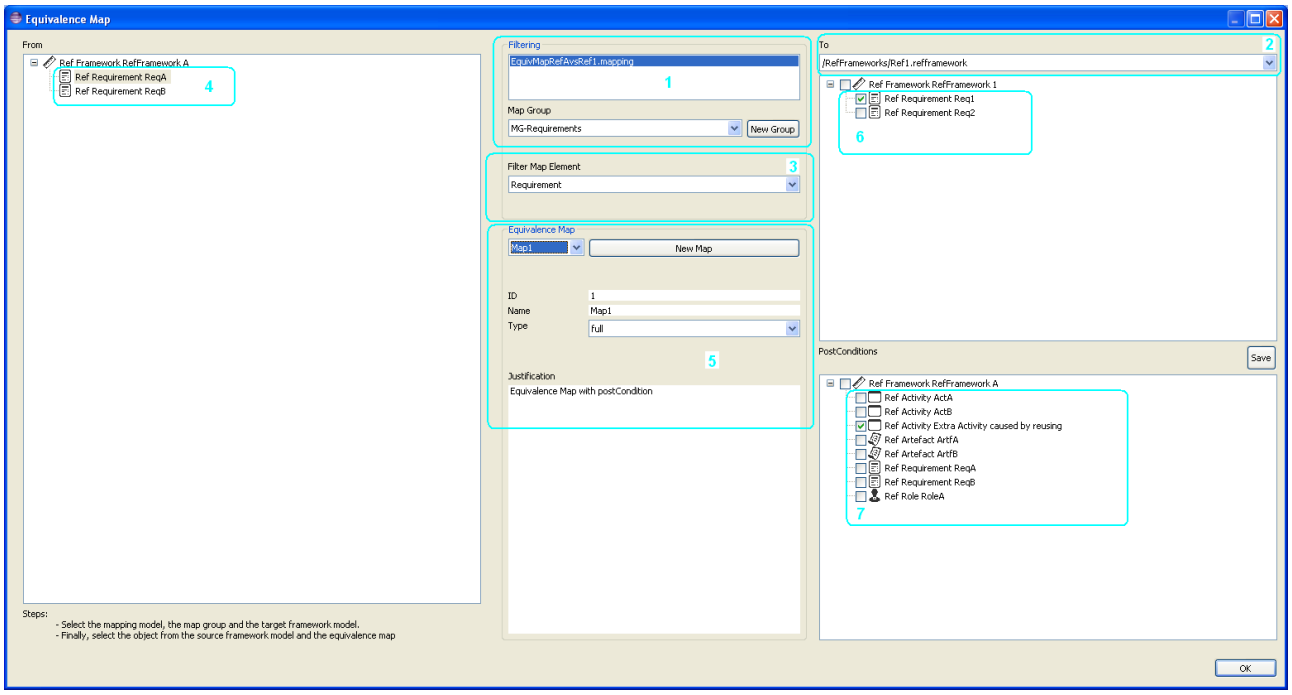


Figure 48 - Steps for making Equivalence Map.

5.4 Creating Applicability Tables

To create Applicability Tables (the naming is to cover various standards, for IEC 61508 derived standards would be Recommendation Tables) such as this one:

Table 1 — System design analysis

	Methods	ASIL			
		A	B	C	D
1	Deductive analysis[a]	o	+	++	++
2	Inductive analysis[b]	++	++	++	++

[a] Deductive analysis methods include FTA, reliability block diagrams, Ishikawa diagram.

[b] Inductive analysis methods include FMEA, ETA, Markov modelling.

Figure 49 - Applicability Table ISO 26262

For ISO26262 and IEC 61508 standards, you must select the desired “Requirement” from the Standard, go to Tab called RequirementApplicability (to go to a Requirement Form, you must first select the Activity which contains the Requirement, and double click on the Requirement properties), and you will see:

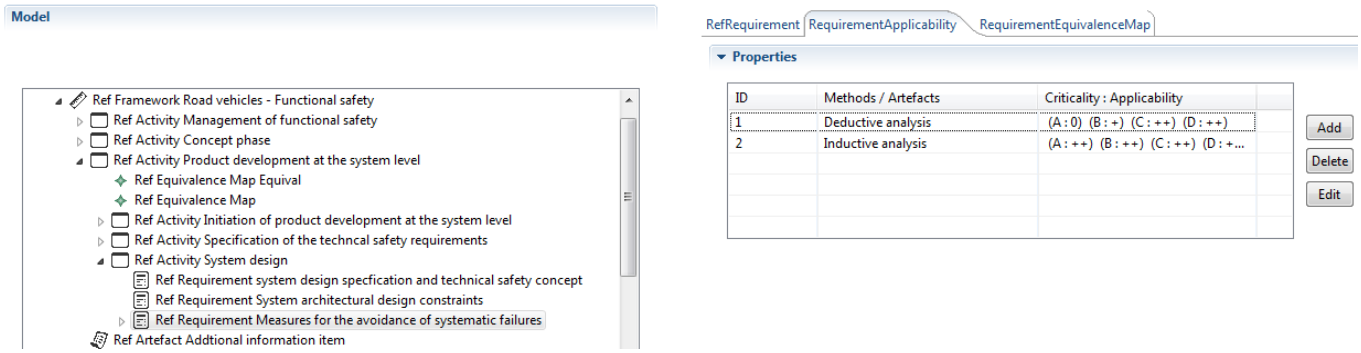


Figure 50 - Requirement Applicability Table ISO 26262

Then you can add rows by defining the **target Method (technique)**, and select the Criticality Level and the Recommendation level.

NOTE: You must first create Criticality Levels (SIL) and Recommendation Levels (++, +, o, or others as required) by clicking on the Diagram (in some blank space) where you will see Properties of the RefFramework model element (the Standard).

In the case of the DO-178C standard, to create Applicability Table such as the next figure:

	Objective		Activity	Applicability by Software Level				Output		Control Category by Software Level			
	Description	Ref	Ref	A	B	C	D	Data Item	Ref	A	B	C	D
1	The activities of the software life cycle processes are defined.	4.1.a	4.2.a 4.2.c 4.2.d 4.2.e 4.2.g 4.2.l 4.2.l 4.3.c	O	O	O	O	PSAC SDP SVP SCM Plan SQA Plan	11.1 11.2 11.3 11.4 11.5	① ① ① ① ① ① ② ② ① ① ② ② ① ① ② ② ① ① ② ②			
2	The software life cycle(s), including the inter-relationships between the processes, their sequencing, feedback mechanisms, and transition criteria, is defined.	4.1.b	4.2l 4.3.b	O	O	O		PSAC SDP SVP SCM Plan SQA Plan	11.1 11.2 11.3 11.4 11.5	① ① ① ① ① ② ① ① ② ① ① ② ① ① ②			
3	Software life cycle environment is selected and defined.	4.1.c	4.4.1 4.4.2.a 4.4.2.b 4.4.2.c 4.4.3	O	O	O		PSAC SDP SVP SCM Plan SQA Plan	11.1 11.2 11.3 11.4 11.5	① ① ① ① ① ② ① ① ② ① ① ② ① ① ②			
4	Additional considerations are addressed.	4.1.d	4.2.f 4.2.h 4.2.l 4.2.j 4.2.k	O	O	O	O	PSAC SDP SVP SCM Plan SQA Plan	11.1 11.2 11.3 11.4 11.5	① ① ① ① ① ① ② ② ① ① ② ② ① ① ② ② ① ① ② ②			
5	Software development standards are defined.	4.1.e	4.2.b 4.2.g 4.5	O	O	O		SW Requirements Standards SW Design Standards SW Code Standards	11.6 11.7 11.8	① ① ② ① ① ② ① ① ②			
6	Software plans comply with this document.	4.1.f	4.3.a 4.6	O	O	O		Software Verification Results	11.14	② ② ②			
7	Development and revision of software plans are coordinated.	4.1.g	4.2.g 4.6	O	O	O		Software Verification Results	11.14	② ② ②			

Figure 51 - Applicability Table DO-178C

It needs to make two steps:

1. First, it needs to define the applicability table for the activities: select the desired “Activity” from the Standard, go to Tab called ActivityApplicability.

   Then you can add rows by defining the **target Requirement**, and select the Criticality Level and the Recommendation level.

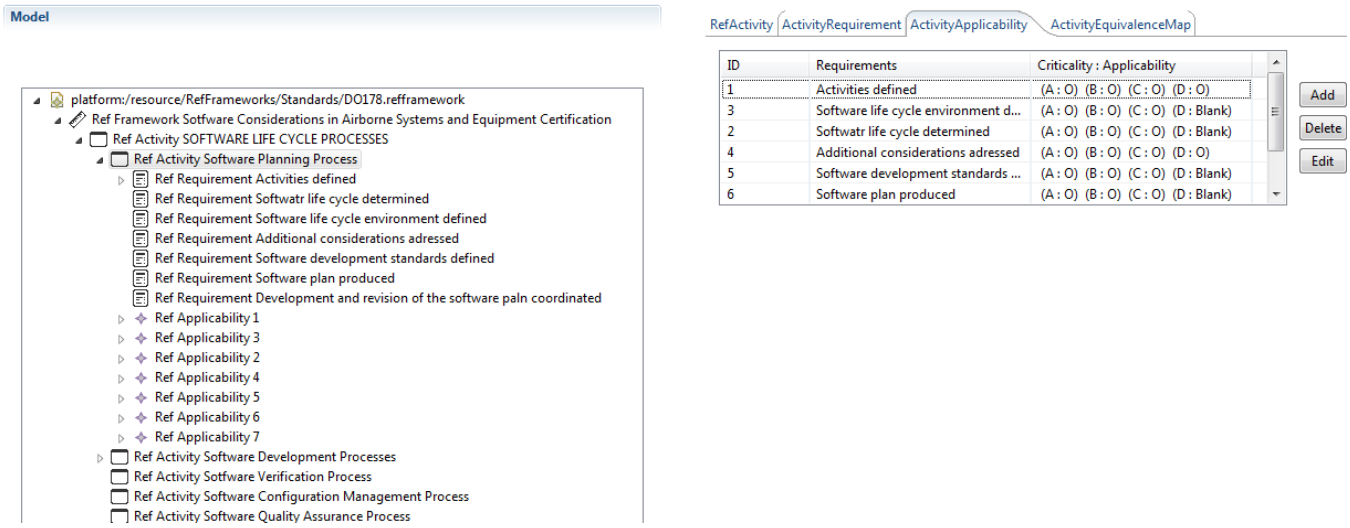


Figure 52 - Activity Applicability Table DO-178C

1. And finally, it needs to define the applicability table for the requirements: select the desired “Requirement” from the Standard, go to Tab called RequirementApplicability.

   Then you can add rows by defining **the target Artefact**, and select the Criticality Level and the Recommendation level.

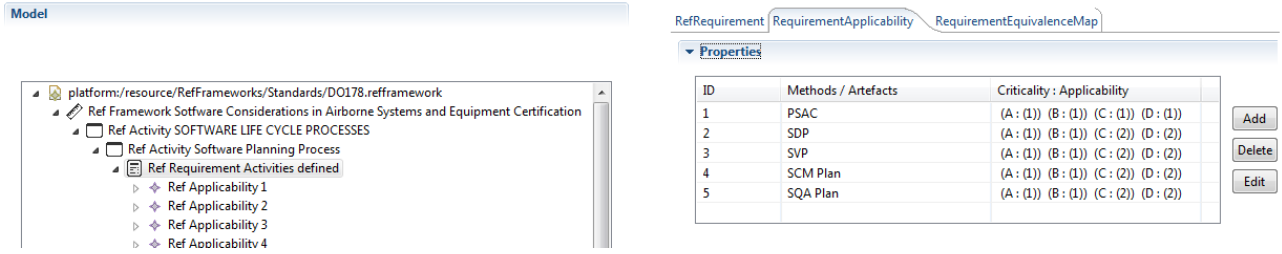


Figure 53 - Requirement Applicability Table DO-178C

In summary:

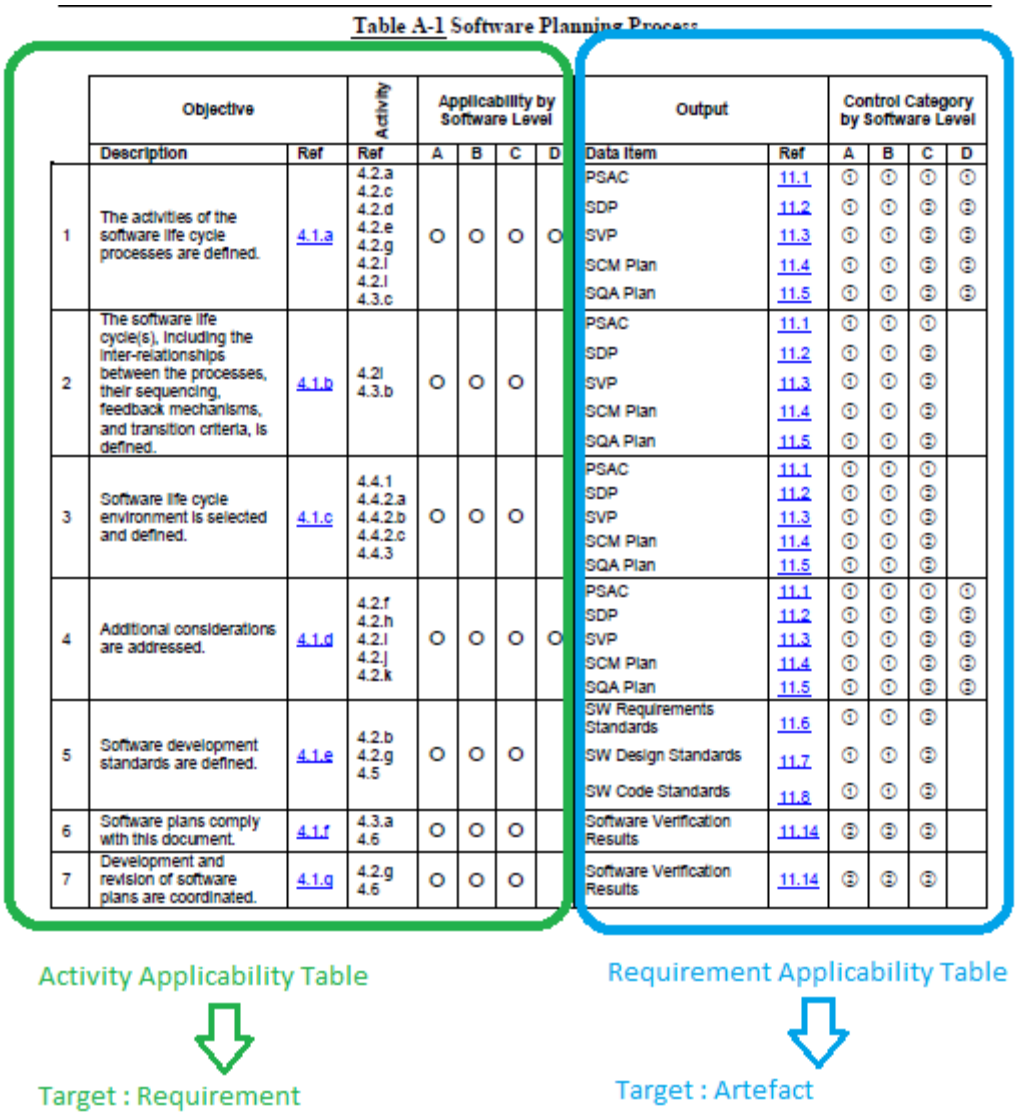


Table A-1 Software Planning Process

	Objective		Activity	Applicability by Software Level				Output		Control Category by Software Level			
	Description	Ref	Ref	A	B	C	D	Data Item	Ref	A	B	C	D
1	The activities of the software life cycle processes are defined.	4.1.a	4.2.a 4.2.c 4.2.d 4.2.e 4.2.g 4.2.i 4.2.l 4.3.c	O	O	O	O	PSAC	11.1	①	①	①	①
								SDP	11.2	①	①	②	②
								SVP	11.3	①	①	②	②
								SCM Plan	11.4	①	①	②	②
								SQA Plan	11.5	①	①	②	②
2	The software life cycle(s), including the inter-relationships between the processes, their sequencing, feedback mechanisms, and transition criteria, is defined.	4.1.b	4.2l 4.3.b	O	O	O		PSAC	11.1	①	①	①	
								SDP	11.2	①	①	②	
								SVP	11.3	①	①	②	
								SCM Plan	11.4	①	①	②	
								SQA Plan	11.5	①	①	②	
3	Software life cycle environment is selected and defined.	4.1.c	4.4.1 4.4.2.a 4.4.2.b 4.4.2.c 4.4.3	O	O	O		PSAC	11.1	①	①	①	
								SDP	11.2	①	①	②	
								SVP	11.3	①	①	②	
								SCM Plan	11.4	①	①	②	
								SQA Plan	11.5	①	①	②	
4	Additional considerations are addressed.	4.1.d	4.2.f 4.2.h 4.2.l 4.2.j 4.2.k	O	O	O	O	PSAC	11.1	①	①	①	①
								SDP	11.2	①	①	②	②
								SVP	11.3	①	①	②	②
								SCM Plan	11.4	①	①	②	②
								SQA Plan	11.5	①	①	②	②
5	Software development standards are defined.	4.1.e	4.2.b 4.2.g 4.5	O	O	O		SW Requirements Standards	11.6	①	①	②	
								SW Design Standards	11.7	①	①	②	
								SW Code Standards	11.8	①	①	②	
6	Software plans comply with this document.	4.1.f	4.3.a 4.6	O	O	O		Software Verification Results	11.14	②	②	②	
7	Development and revision of software plans are coordinated.	4.1.g	4.2.g 4.6	O	O	O		Software Verification Results	11.14	②	②	②	

Figure 54 - Summary Applicability Table DO-178C

6 Assurance Project Management

As described in Section 2.1, users can maintain the lifecycle of projects by creating Assurance Projects. An Assurance Project can have multiple Baseline Configurations, Permissions Configurations and Assurance Assets Package, but only one is active at once.

6.1 Create Assurance Project and Baseline

To create a new assurance project go to the menu File→ New → Project or use the button in the top button bar and select New Assurance Project inside the AMASS category.

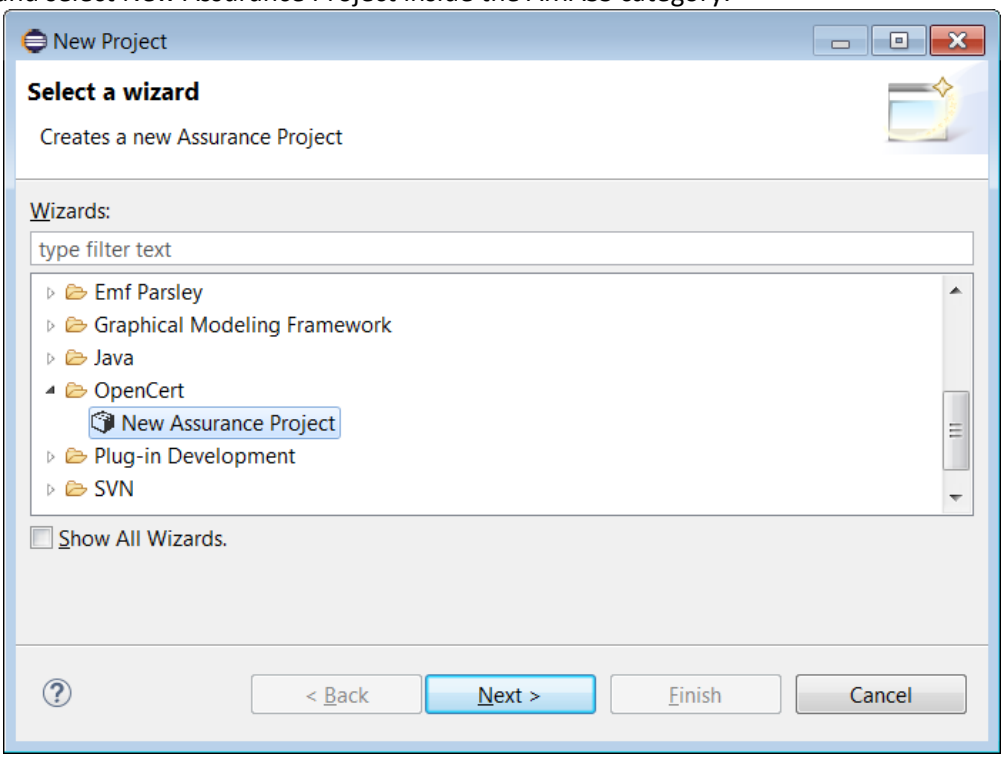


Figure 55 - New Assurance Project wizard

The first page is to enter the name of Assurance project and to choose if the user wants to automatically generate an Argumentation model based on the standard parts selected in the next wizard page.

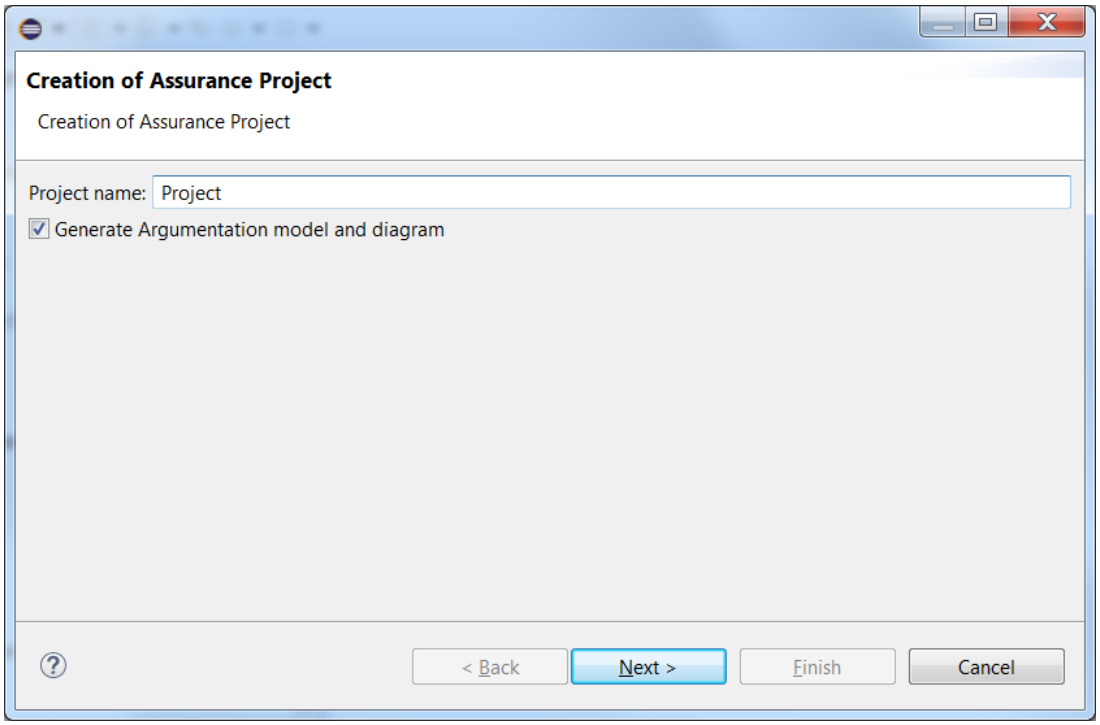


Figure 56 - Assurance Project name page

The next page of the wizard will show in the left the list of reference framework model(the standard) in the whole repository. Select the desired reframework and in the bottom right list will appear its contents in form of checkable tree for the generation of the baseline.

If the user wants to create an assurance project to accomplish with a certain criticality and/or applicability level, the user can select them from the upper right lists and the parts the mandatory parts of the standard to comply with the selected levels will be automatically selected, and also the other parts of the standard that don't have relation with any criticality or applicability level(see section 5.4 Creating Applicability Tables). The text "**** Selected in Assurance Project Generation Wizard ****" will be added to the description of the selected criticality and/or applicability levels in the generated baseline model.

The user can also select directly the contents of the standard that will be applied to the project that is going to be created using the bottom tree.

After giving a name to the baseline and clicking the Finish button, the project information generation process will start (This process could take several seconds). Only is possible to Finish the process if you give a name and select almost one concept.

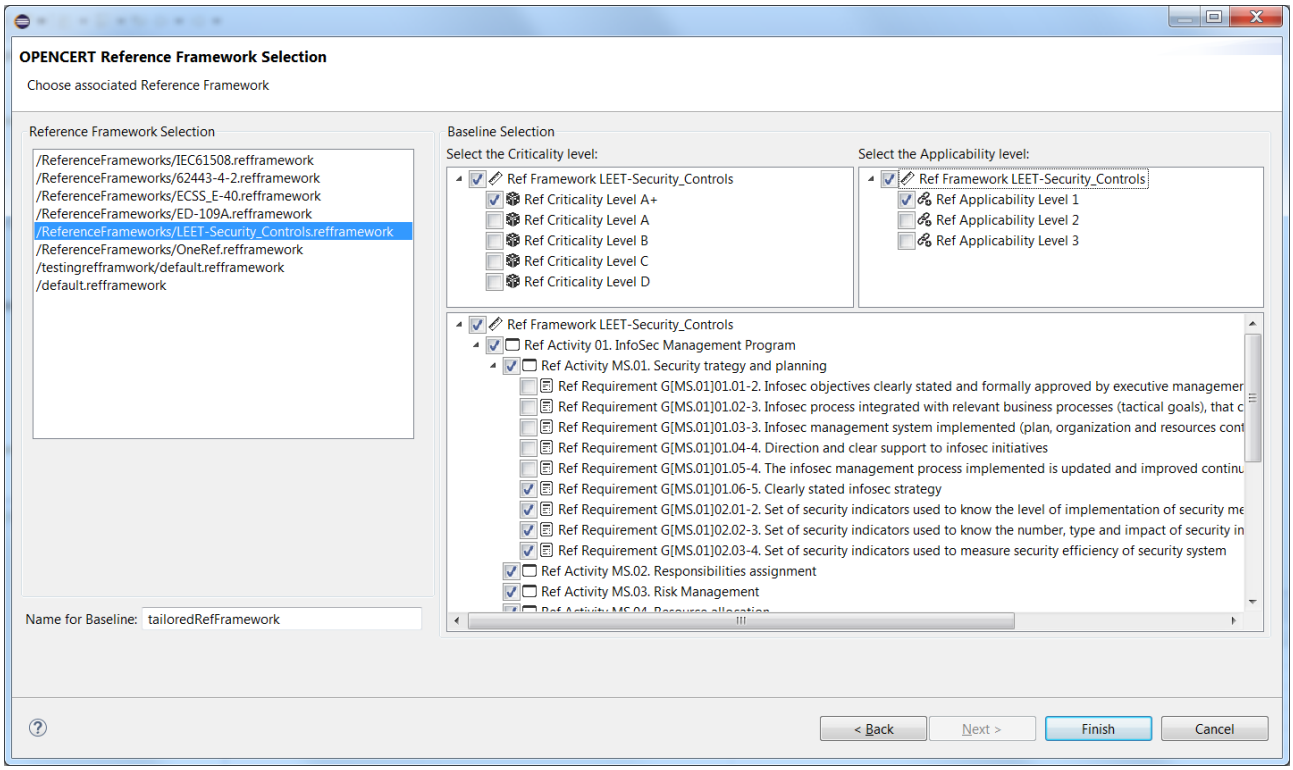


Figure 57 - Reframework selection

Now in the Repository Explorer the new project will be displayed.

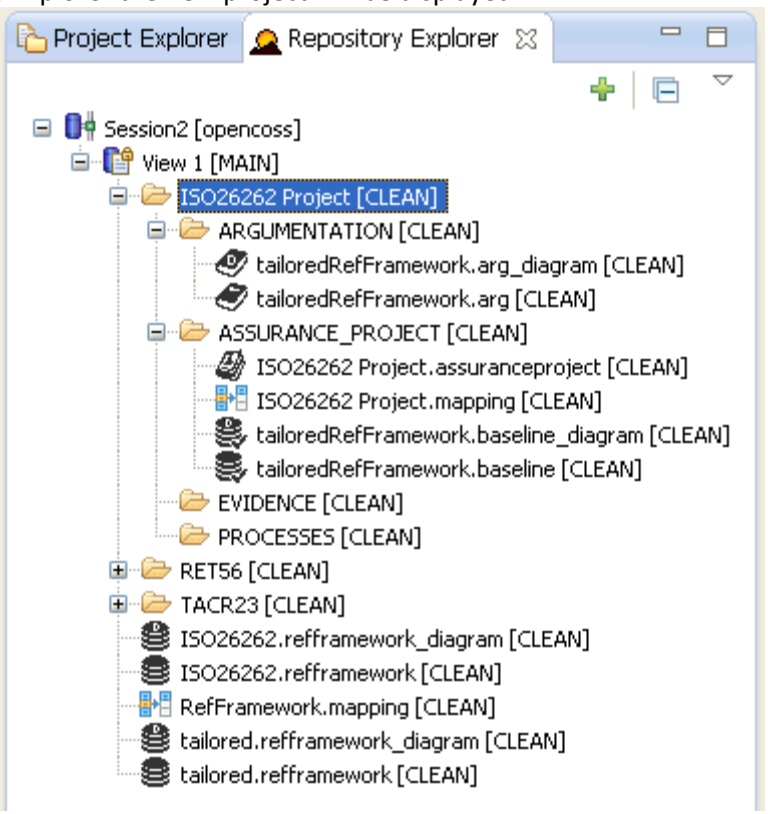


Figure 58 - Assurance Project structure

The project is composed by 4 folders:

- Argumentation for storing the argumentation models with and argumentation model (with diagram) generated automatically based on the baseline's selected entities. (For details, see section 7.2.2 Creating a Diagram at the project creation time)
- Assurance Project folder that has the project information in .assuranceproject model, the baseline information in .baseline model(with diagram) and the .mapping model to store the compliance mapping information.
- Evidence for saving the evidence models
- Process for the processes execution.

To edit the Assurance Project information double clicks over the model and its editor will appear. By default the assurance project has related all the models generated automatically, the baseline and mapping models in the active BaselineConfig and the argumentation model in the active AssetsPackage.
If the user generates new models related to the assurance project, for example evidence model, he must select the right folder (EVIDENCE following with the example) of the assurance project as destination for the new model and update manually the assurance project model to reference the new models inside the project (AssesuranceAssents following with the example).

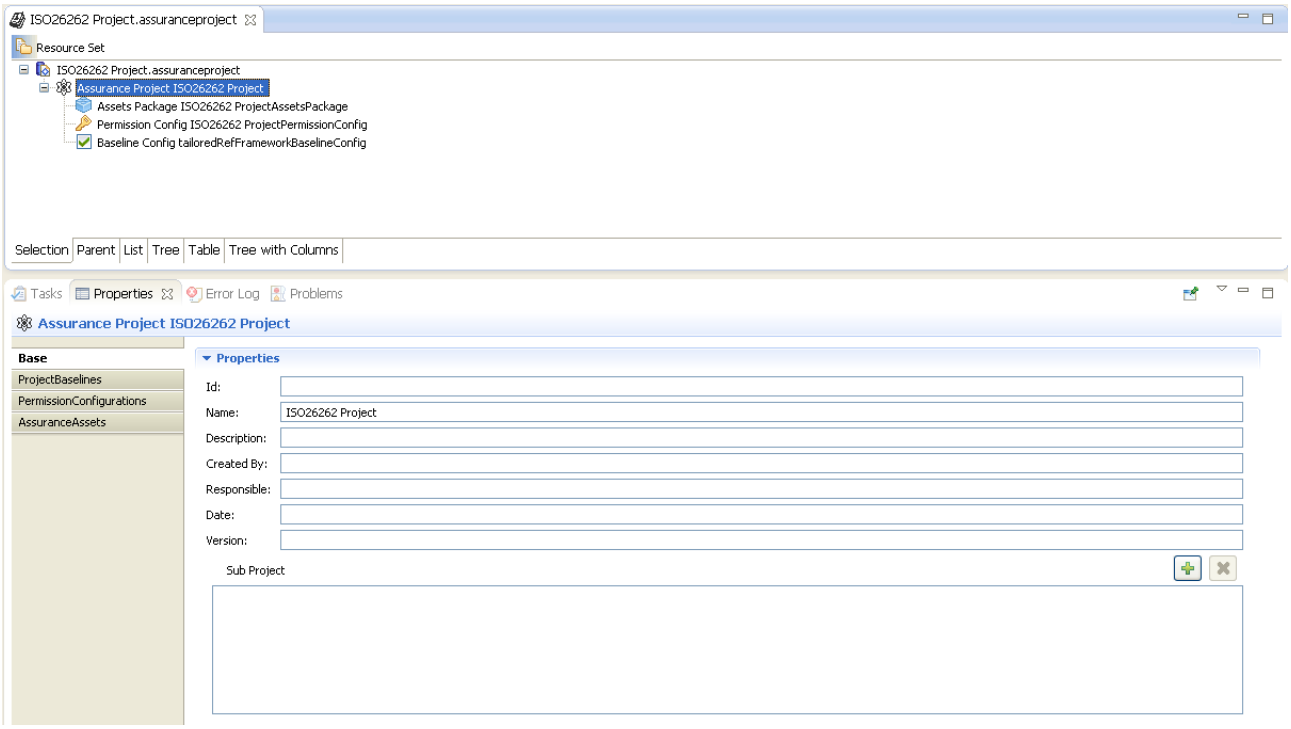


Figure 59 - Assurance Project editor

6.2 Create or update Project Baseline

To create a new assurance project baseline or update and exiting one select menu File→New → Other or use the arrow in the right of the button in the top button bar and select Other.

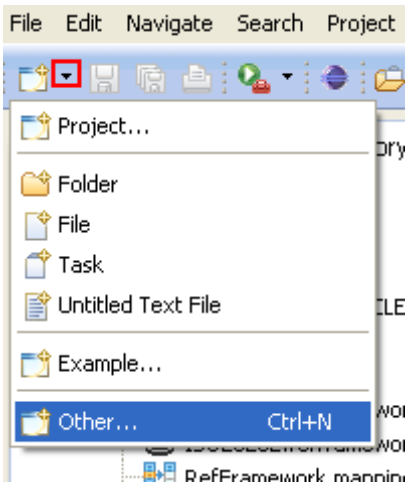


Figure 60 - Other kind of projects option

Choose the wizard Creates or Updates Baseline behind the AMASS category.

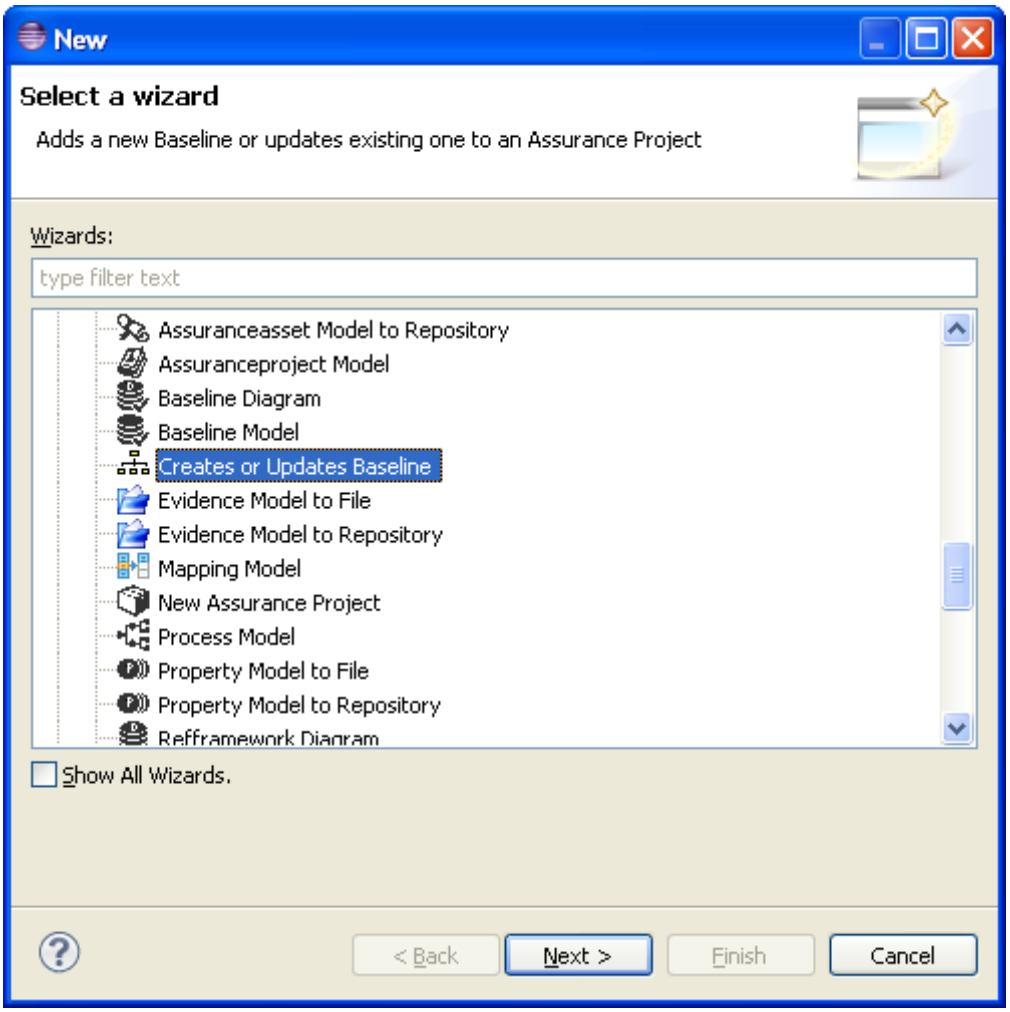


Figure 61 - Creates or Updates Baseline wizard

The first page of the wizard requests the selection of the assurance project model to update.

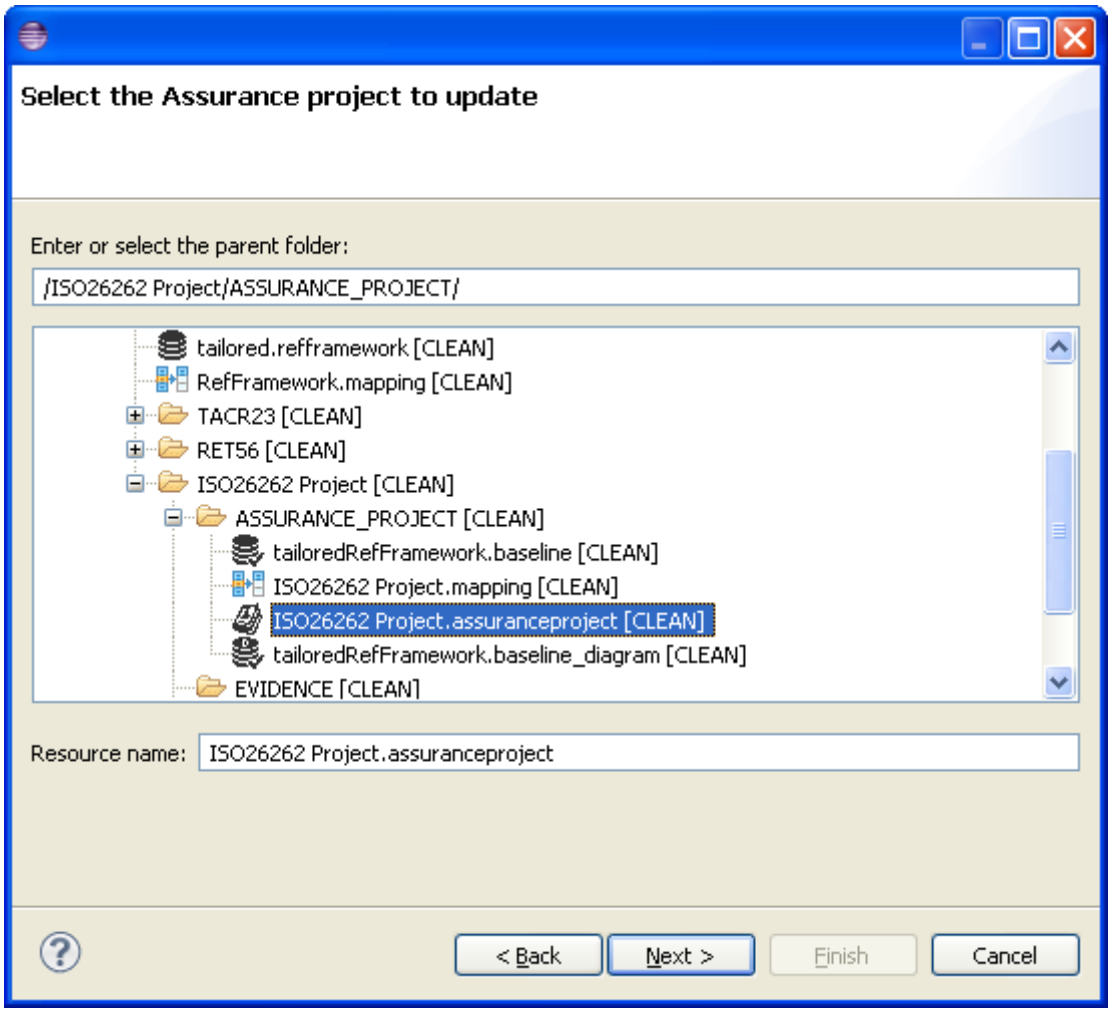


Figure 62 - Selection of the Assurance Project to update

The following steps are exactly the same than for the generation of a new assurance project.
Select the desired reference framework model to be used as source for the generation of the baseline in the left list, then in the right list will appear its contents in form of checkable tree for the generation of the baseline. Select the nodes of the tree that will be applied to this baseline and give a name to the baseline taking into account that if the given name is the same as previous existing baseline, the contents of the previous one will be replaced with the information selected and the same will occur with the argumentation model.
Finally click the Finish button to generate the new baseline and argumentation models that will be added to the assurance project model and stored in the appropriate Assurance Project Folders.

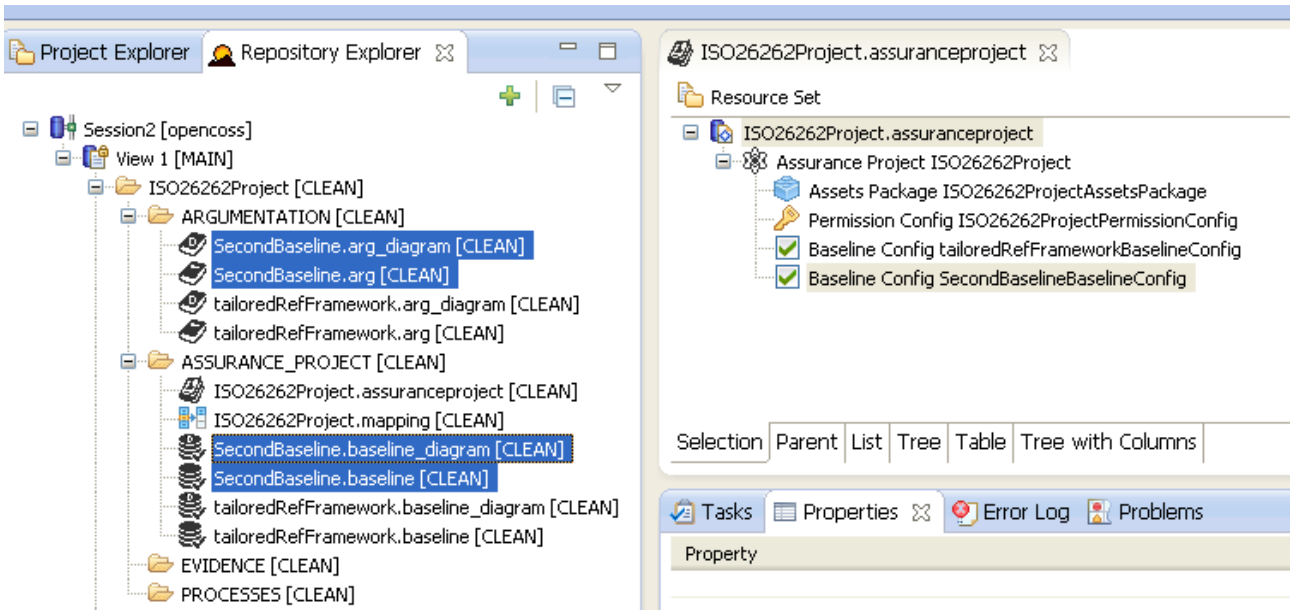


Figure 63 - Assurance Project with new baseline

6.3 Edit Project Baseline

To edit the baseline information double click over the .baseline model and its editor View will appear. The not selected elements will be displayed in the upper tree with a different icon.

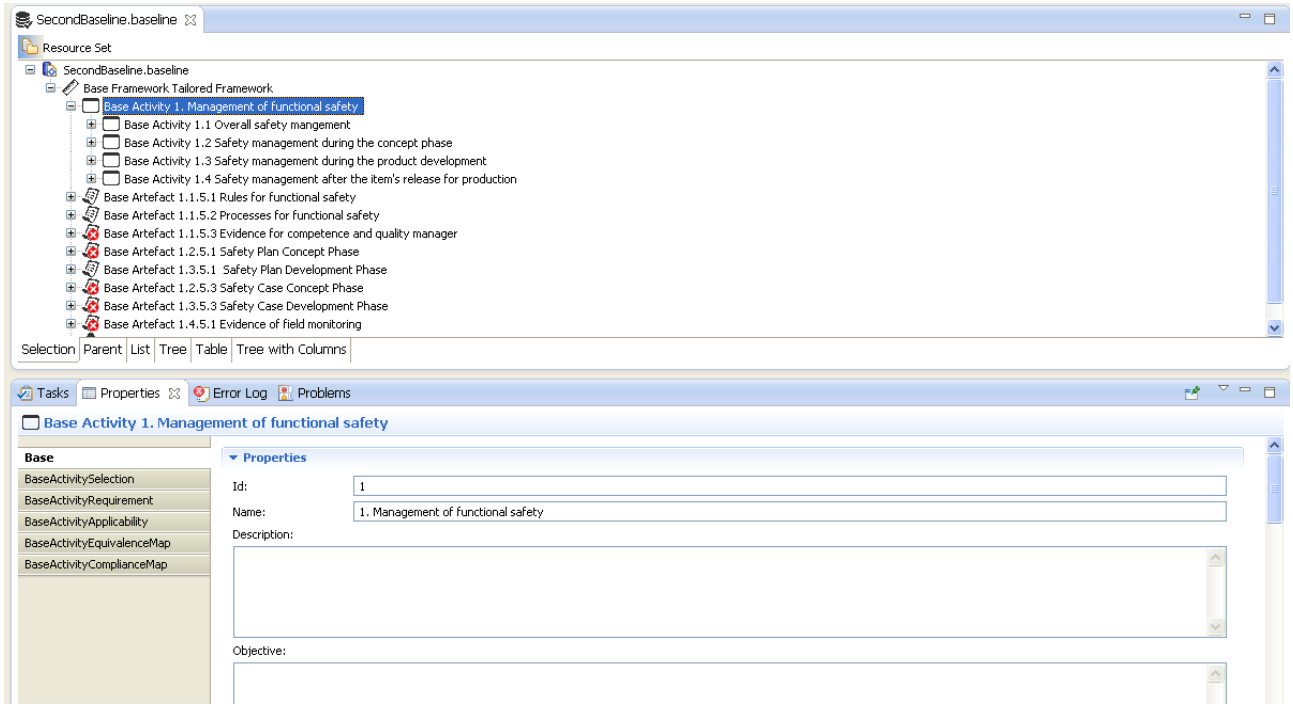


Figure 64 - Baseline editor

The baseline model can be edited also be means of a graphical editor, to use it double click in the .baseline_diagram model. The way of using this editor is exactly equal than the reffframework's editor explained in the chapter 4.2

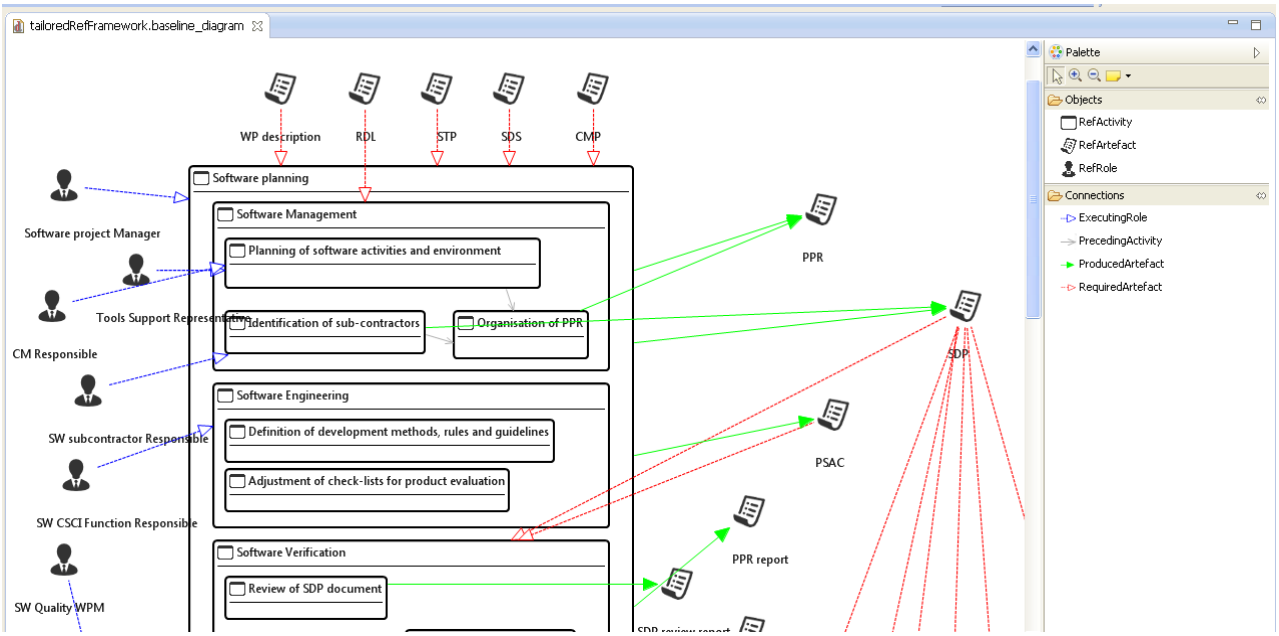


Figure 65 - Baseline graphical editor

6.4 Edit Compliance Maps

It is possible to create Compliance Maps in two ways:

- One way using the editor,
- Another way using a tailored functionality for it.

6.4.1 Compliance Map using the editor

To create Compliance Maps using the editor, we must load four CDO resources: the artefact model (.evidence), process model (.process), argumentation model (.arg) and the mapping model (.mapping).

It is important remind that these models have to be part of the active BaselineConfig and AssetsPackage of the Assurance Project, in other words,

- The artefact model, process model and argumentation model have to be part of the active AssetsPackage of the project:

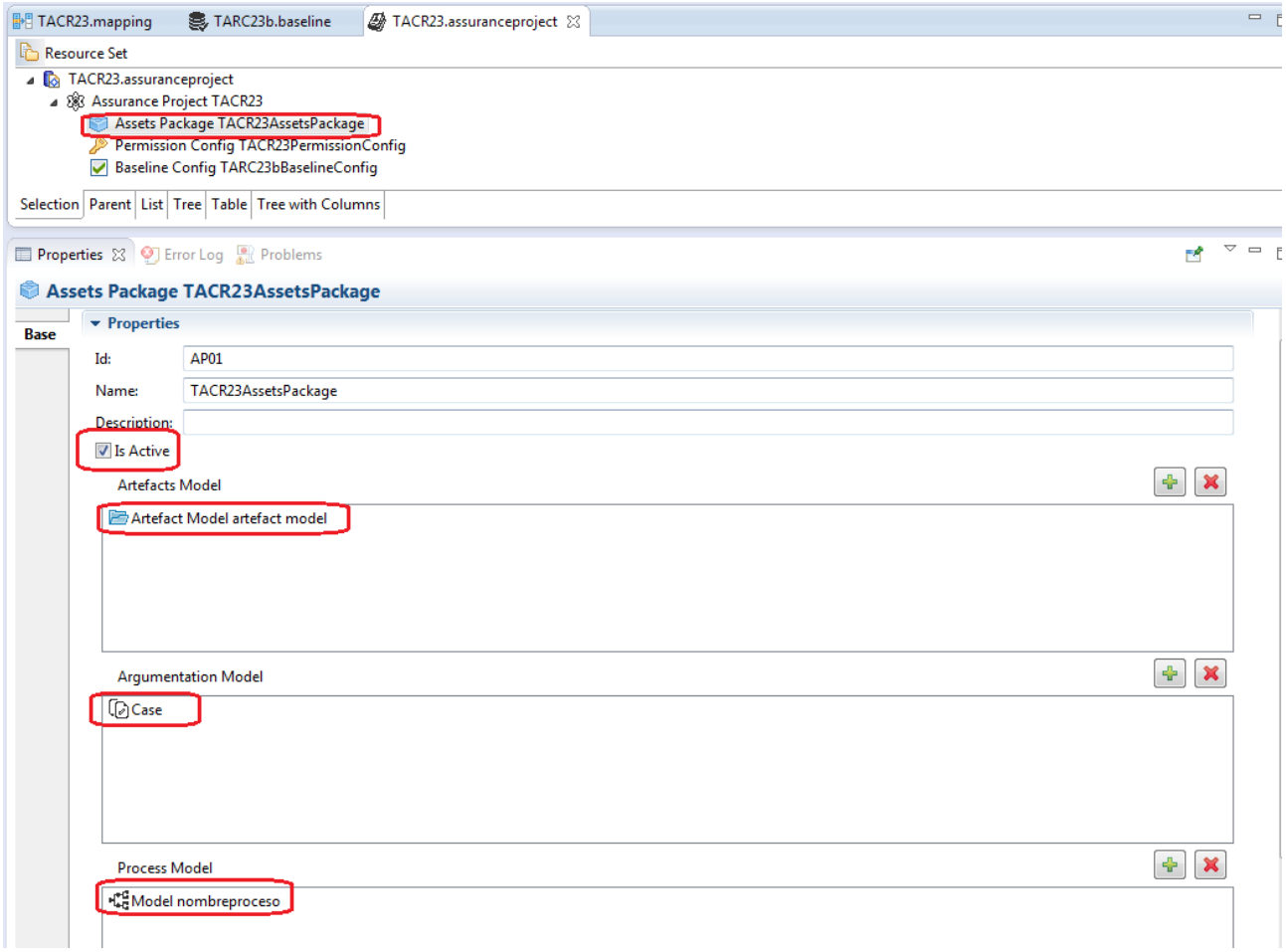


Figure 66 - Assets Package active

- The map group of the mapping model has to be part of the active BaselineConfig of the project:

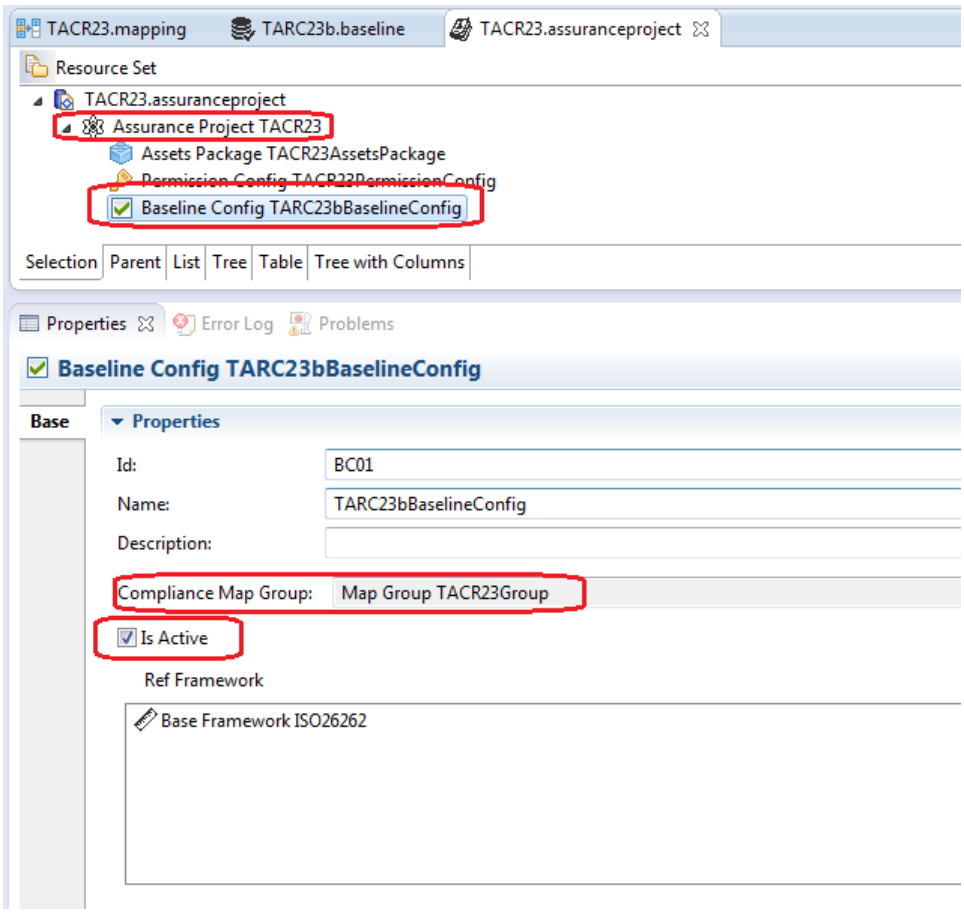


Figure 67 - Baseline Config active

So, press the editing window and select “Load Resource” in the context menu.

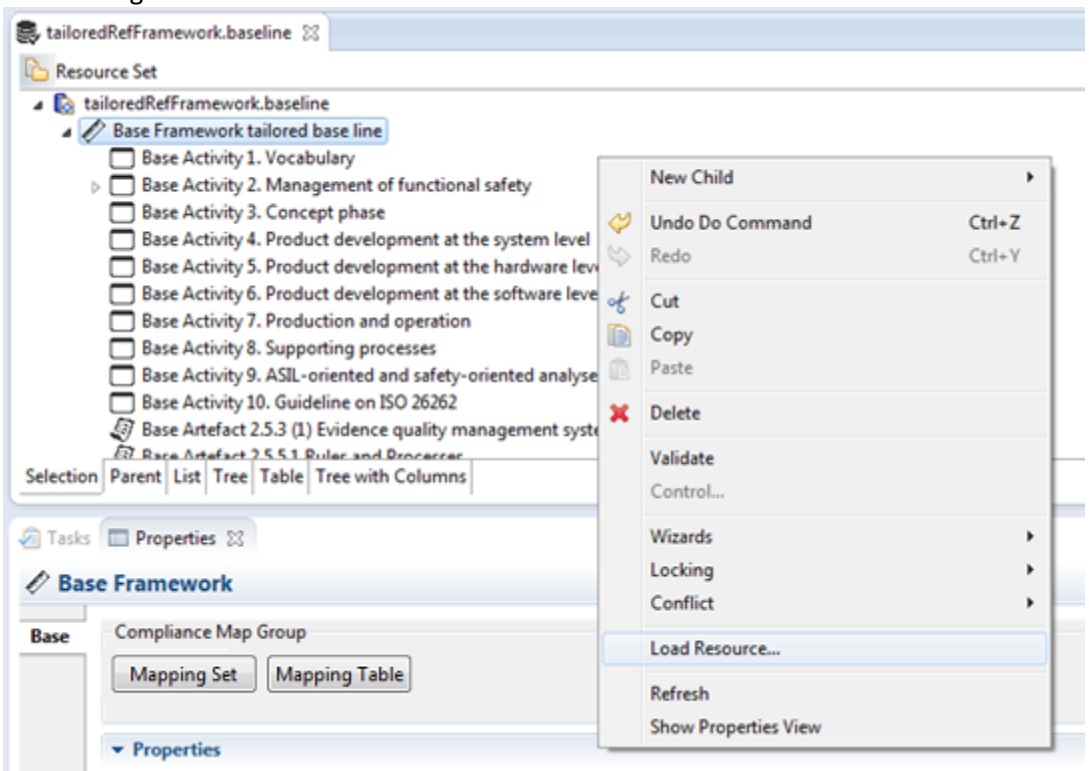


Figure 68 - Load Resource

Other way is select the entry of the menu Baseline *Editor ->Load Resource*

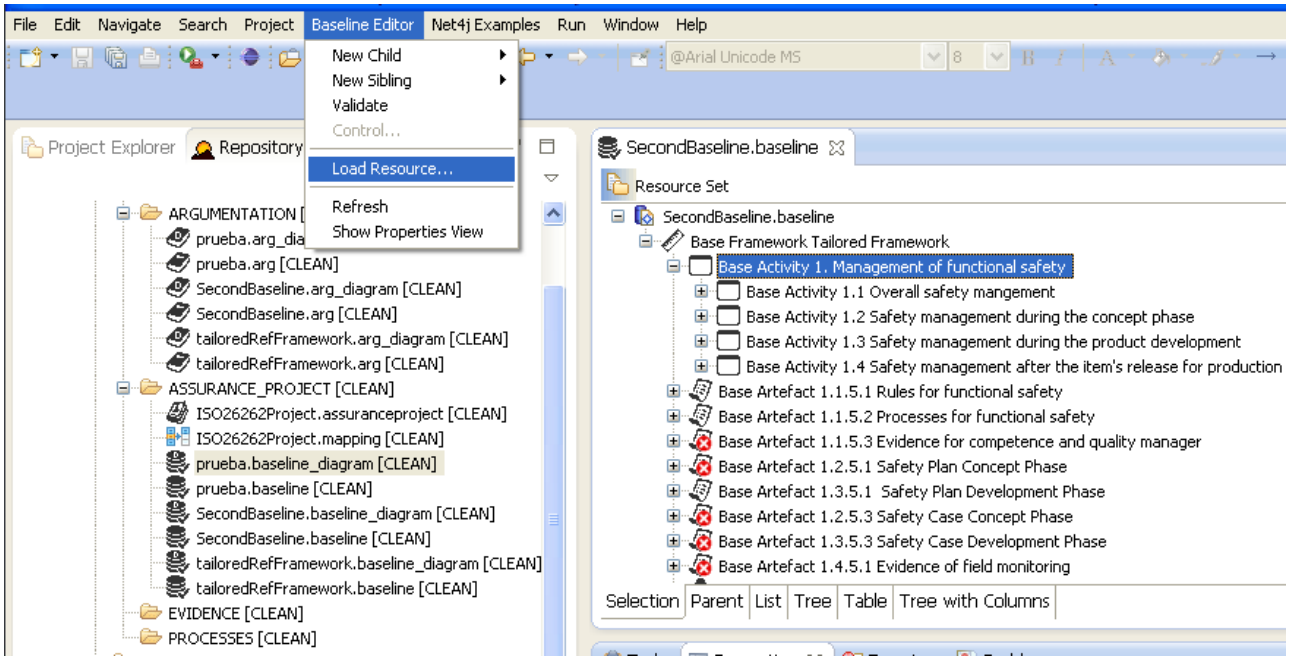


Figure 69 - Load Resource II

Then select the resource model browsing the repository using the “Browse Repository” button.

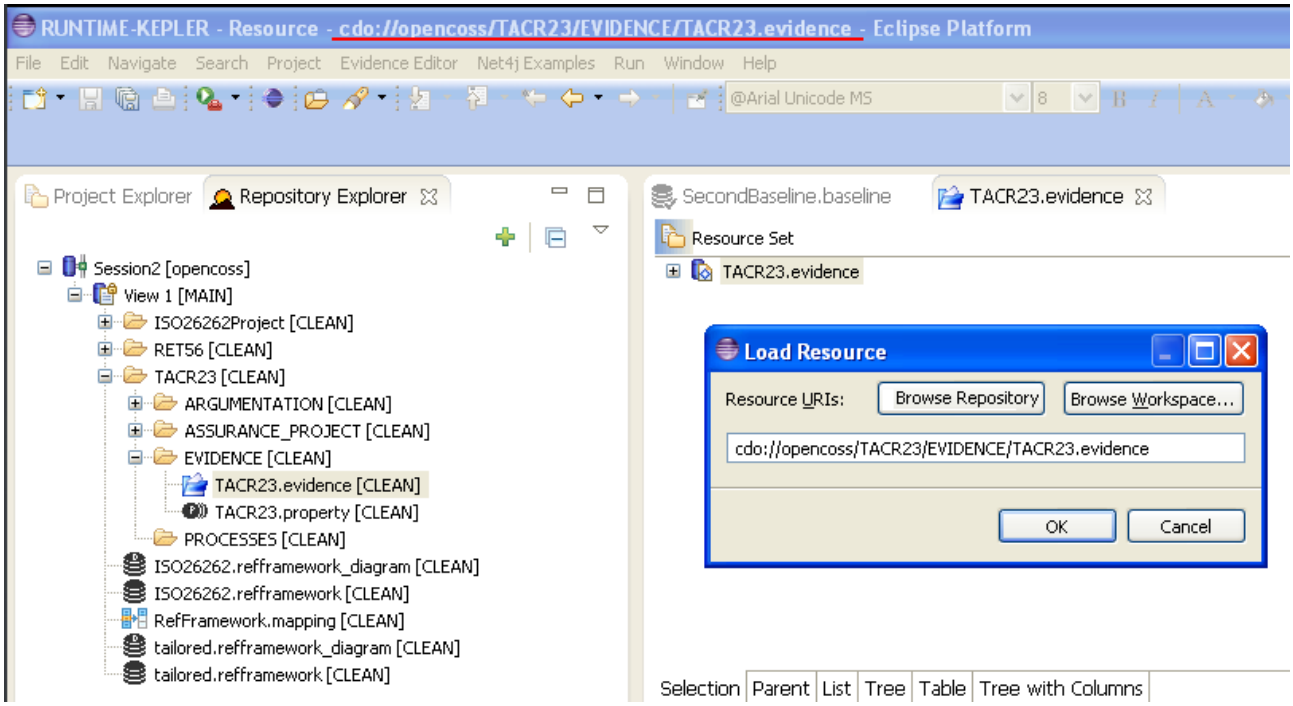


Figure 70 - Load Resource Evidence, Process or Argumentation model

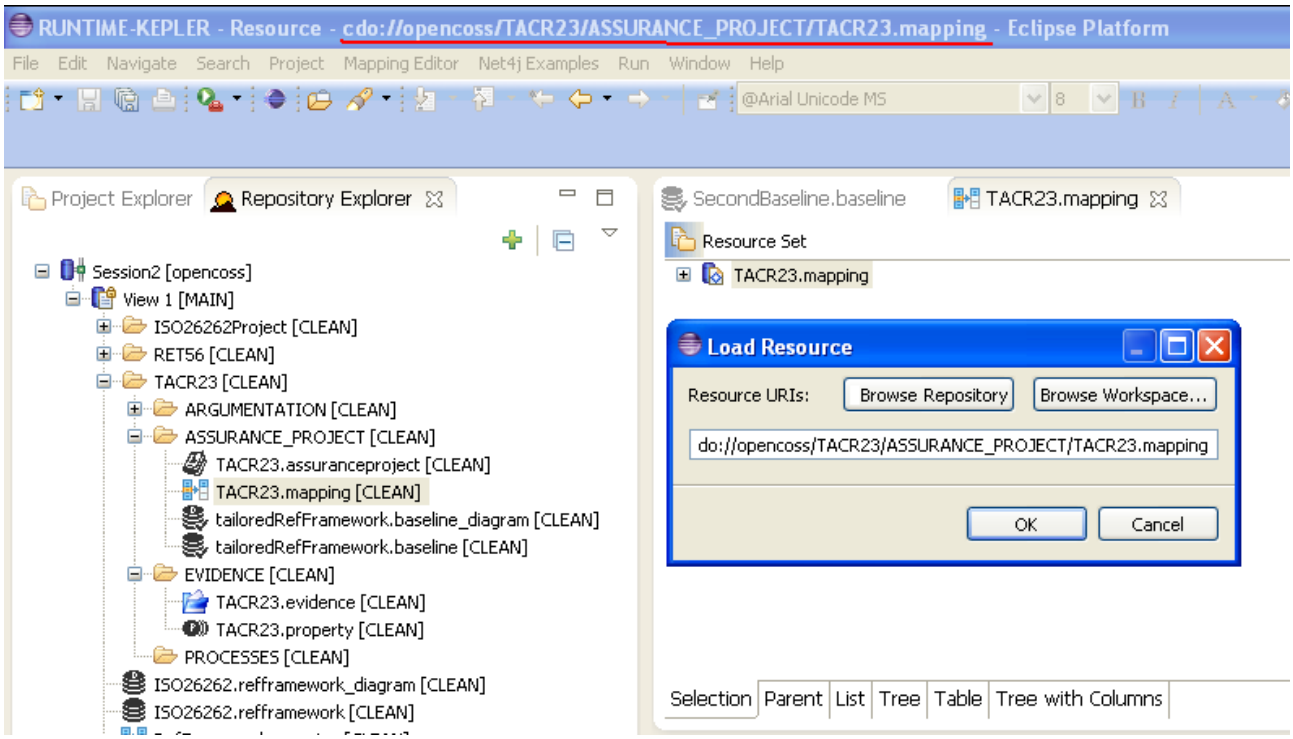


Figure 71 - Load Resource Mapping model

It is possible to create compliance maps for activities, artefacts, requirements, roles and techniques. To do so, first select the object in the tree and after click on the tab “Compliance Map” and press the button “Add”

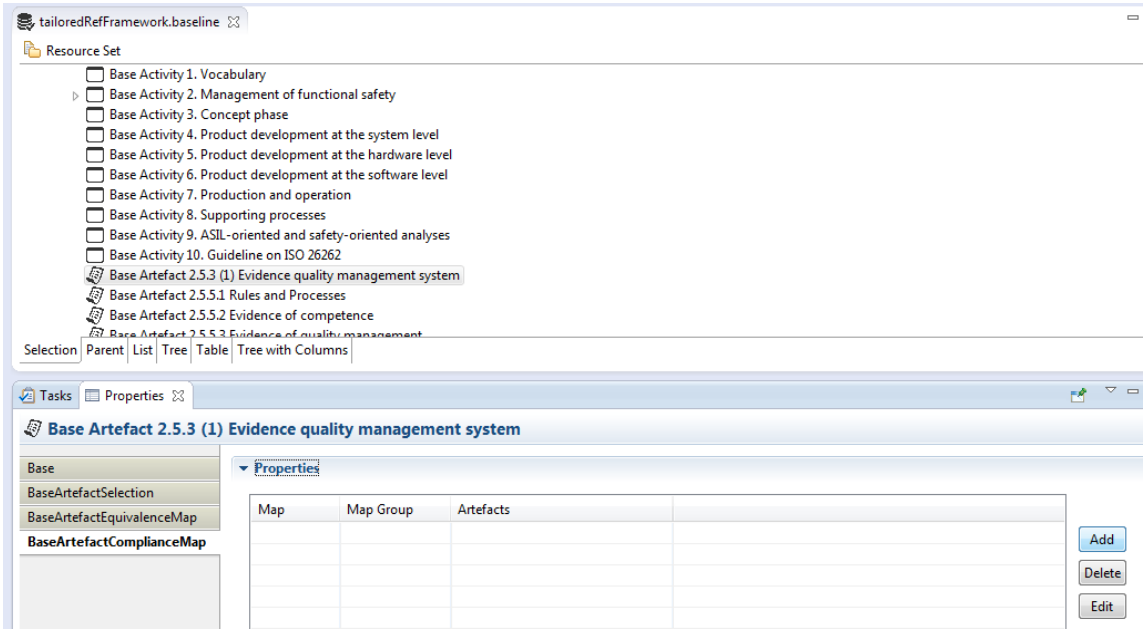


Figure 72 - Artefact Compliance Map

Finally, enter the information requested:

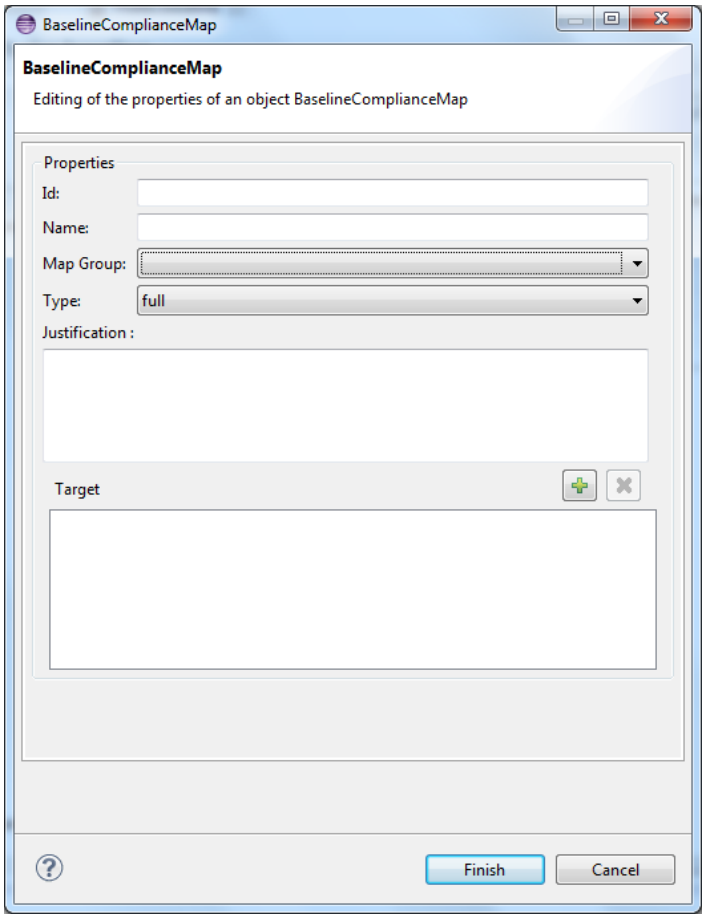


Figure 73 - Compliance Map

Remember that the map group has to be part of the active Baseline Config of the project.

6.4.2 Compliance Map using a tailored functionality.

To create Compliance Maps using the tailored functionality, , first of all, it is necessary to press the button “Mapping Set” on the properties form of the baseline using the tree view editor (not available using the diagram editor). This window automatically saves the mappings when checking or unchecking elements of the target baseline tree.

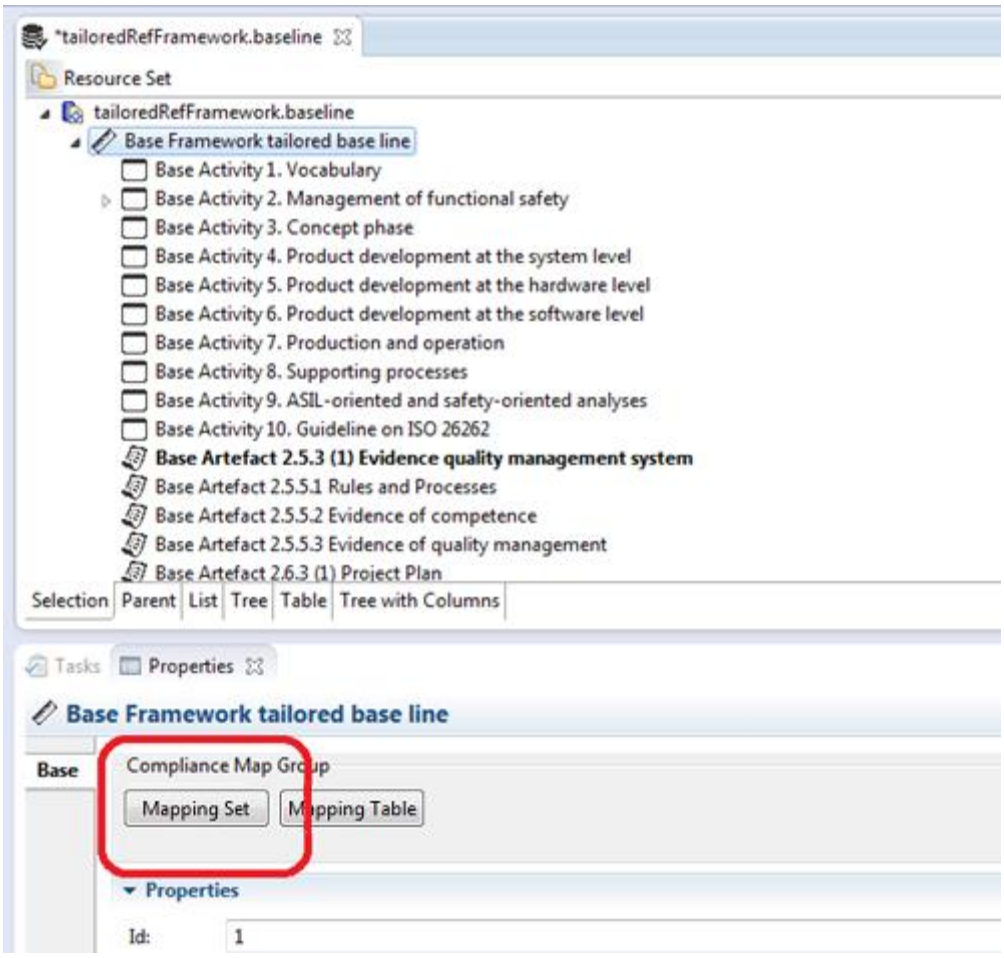


Figure 74 - How to create Compliance Map

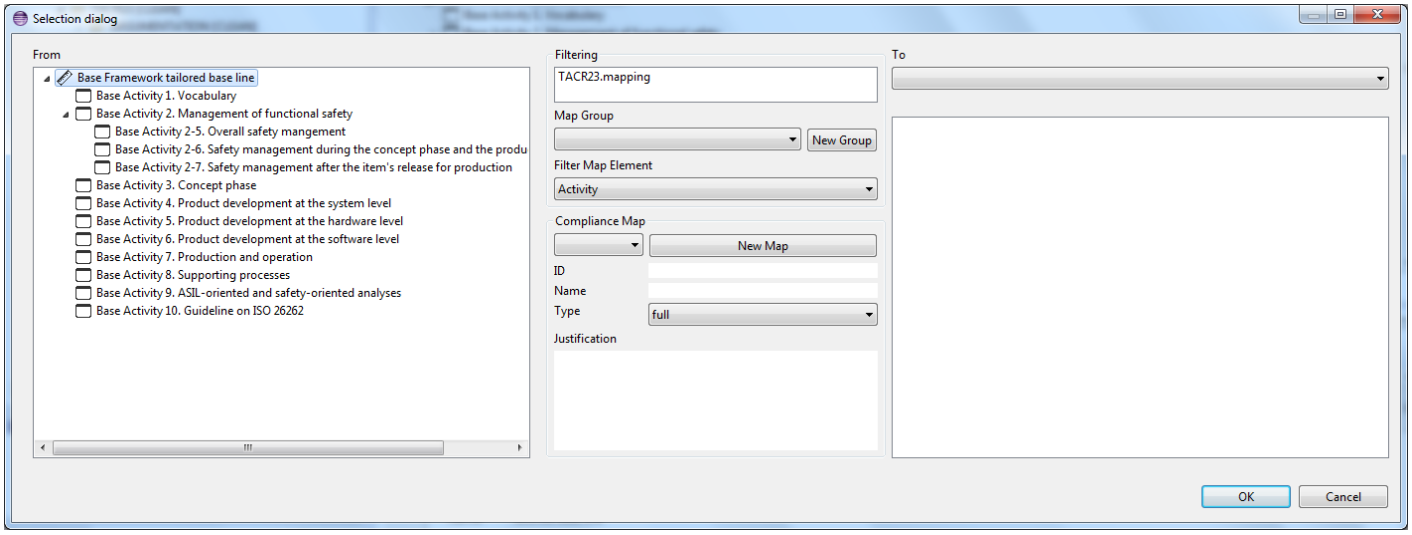


Figure 75 - Compliance Map form

The Compliance Map form is organized in three zones:

- The *left zone* shows the actual baseline, and it loads the type of elements for which we want to make the compliance maps. For default, activities.
- The *middle zone* allows to make different filters like:
    - *Filter Mapping Model* lists all the mapping models stored in the database, and it will be necessary to select one of them and one group model. It's also possible to create a new

map group pressing the button “New group”. This map group has to be part of the active Baseline Config of the project.

- *Filter Map Element*. It’s possible to create compliance maps for activities, artefacts, requirements, roles and techniques, and the allowed maps are:
  - BaseArtefact -> Artefact
  - BaseRequirement -> Artefact , Claim or Activity
  - BaseActivity -> Activity
  - BaseRole -> Participant
  - BaseTechnique -> Technique

  When the filter changes, also it changes the information showed by the reference framework. For example:
  - If the filter “Artefact” is selected:

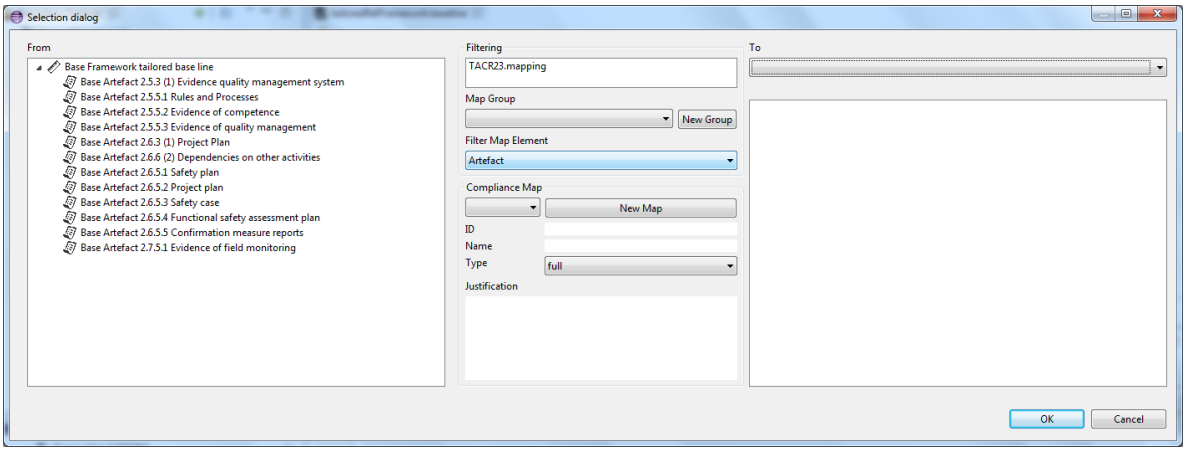


Figure 76 - Compliance Map, select map element

Remember that these models have to be part of the active Assets Package of the project.

- *Filter Compliance Map*. This filter allows making different compliance maps for the same element.

- The *right zone* shows the list of models, depend on the map filter, stored in our database. We should select one of them. This selected model will be the target of the compliance map to create.

For making a compliance map, follow the next steps:

1. Select a mapping model and a map group.
2. Select the target reference framework.
3. Select the filter map element.
4. Select the element from the source reference framework.
5. Select o create the compliance map.
6. And for last, check or uncheck the element from the target model.

6.4.3 Compliance Mapping Table

The Compliance Mapping table allows monitoring the compliance status of one assurance project’s active baseline with the possibility of filtering by several criteria.
To access this window it is necessary to press the button “Mapping Table” on the properties view of the Base Framework element of one baseline using the tree view editor (not available using the diagram editor).

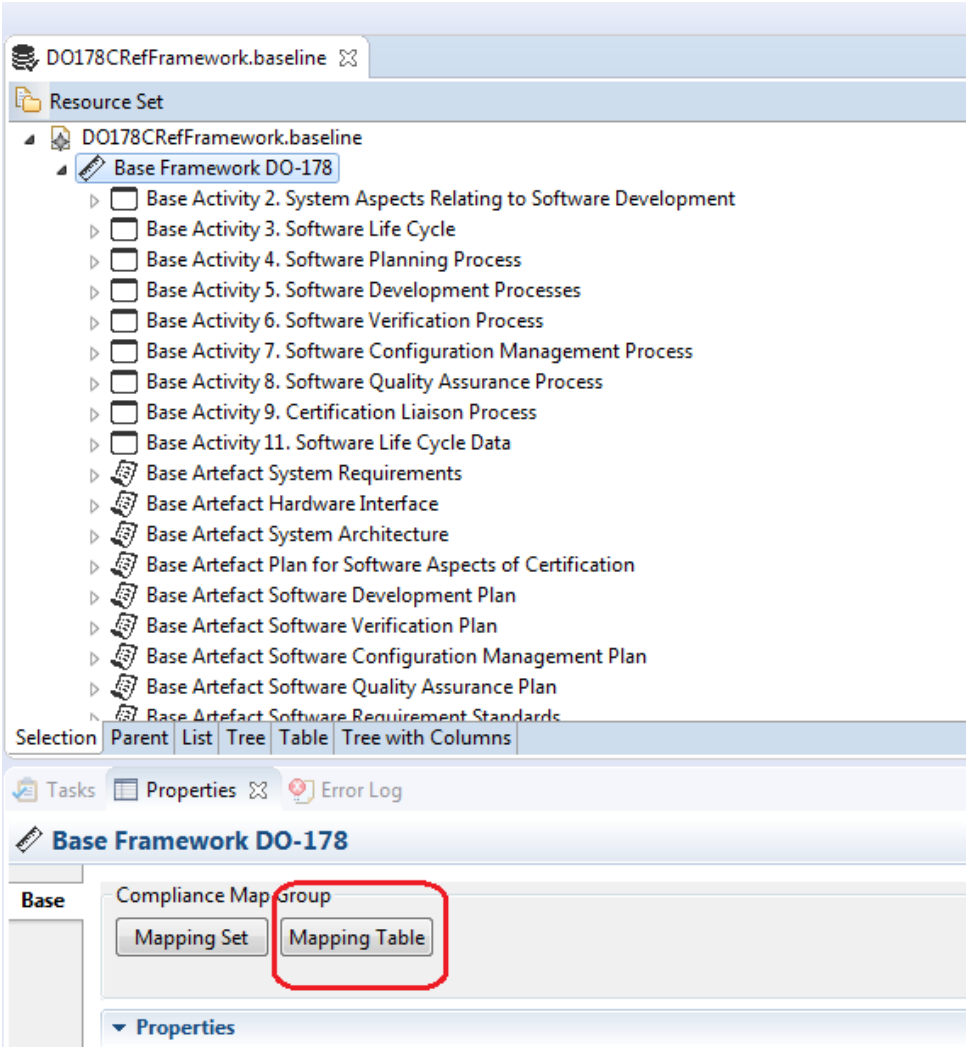


Figure 77 - How to access Compliance Mapping Table

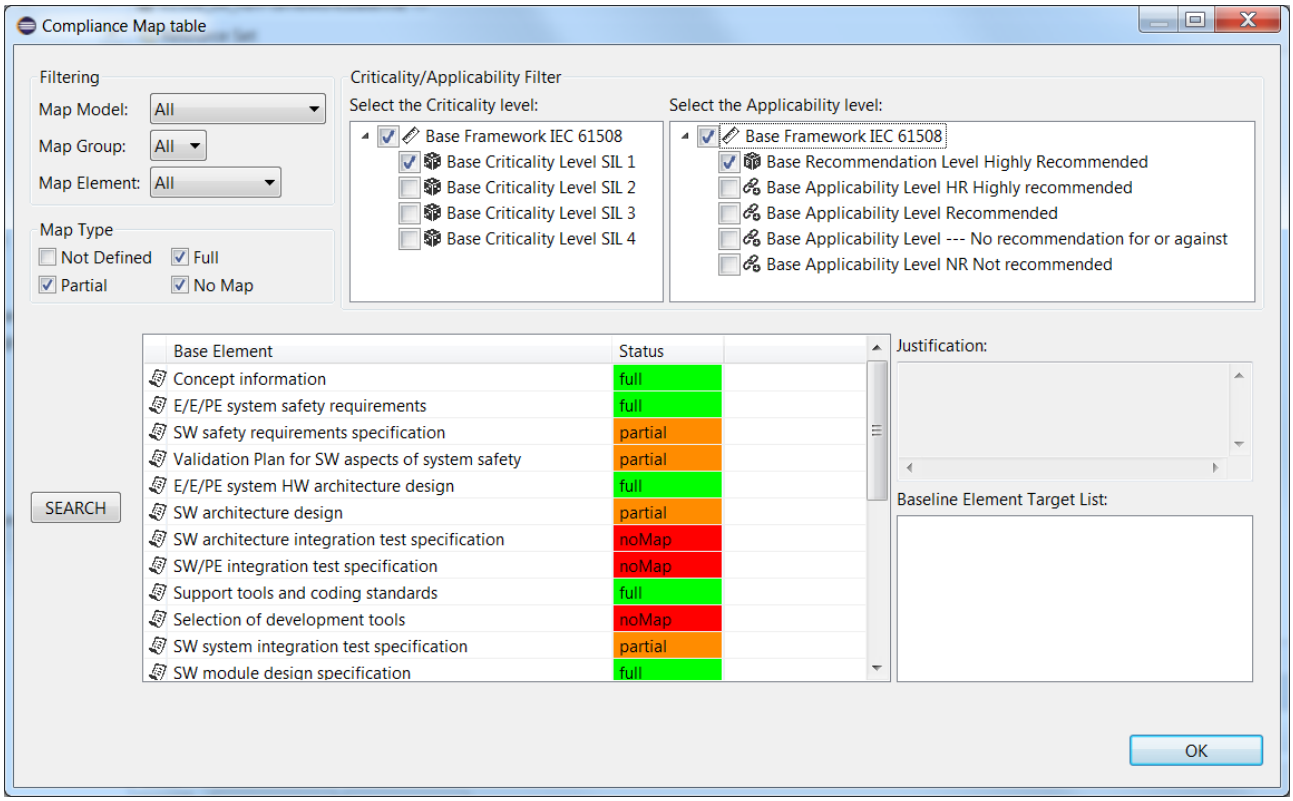


Figure 78 - Mapping Table window

The Compliance Map window is organized in two zones:

- The upper part has controls to allow filtering. It's possible to filter by criticality level, applicability level, map model, a map group of the selected map model, a type of element that could be mapped and the mapping type. The "*Not Defined*" option is to include in the table all the elements of the baseline that has not mapping established, in this way is possible to see the compliance Gap. It's necessary to click the Search button to begin the search process based in the filter options selected that will fill the table.
- The *button part showing three controls*:
    - The compliance mapping table that shows all the baseline elements that accomplish with the searching criteria selected by the user. By default, all the baseline elements that has compliance map are shown in the table.
    - A text box that shows the compliance map justification introduced for the base element selected in the bottom left table with a simple left click.
    - A list that shows all the target elements of the base element selected in the table with a simple left click.

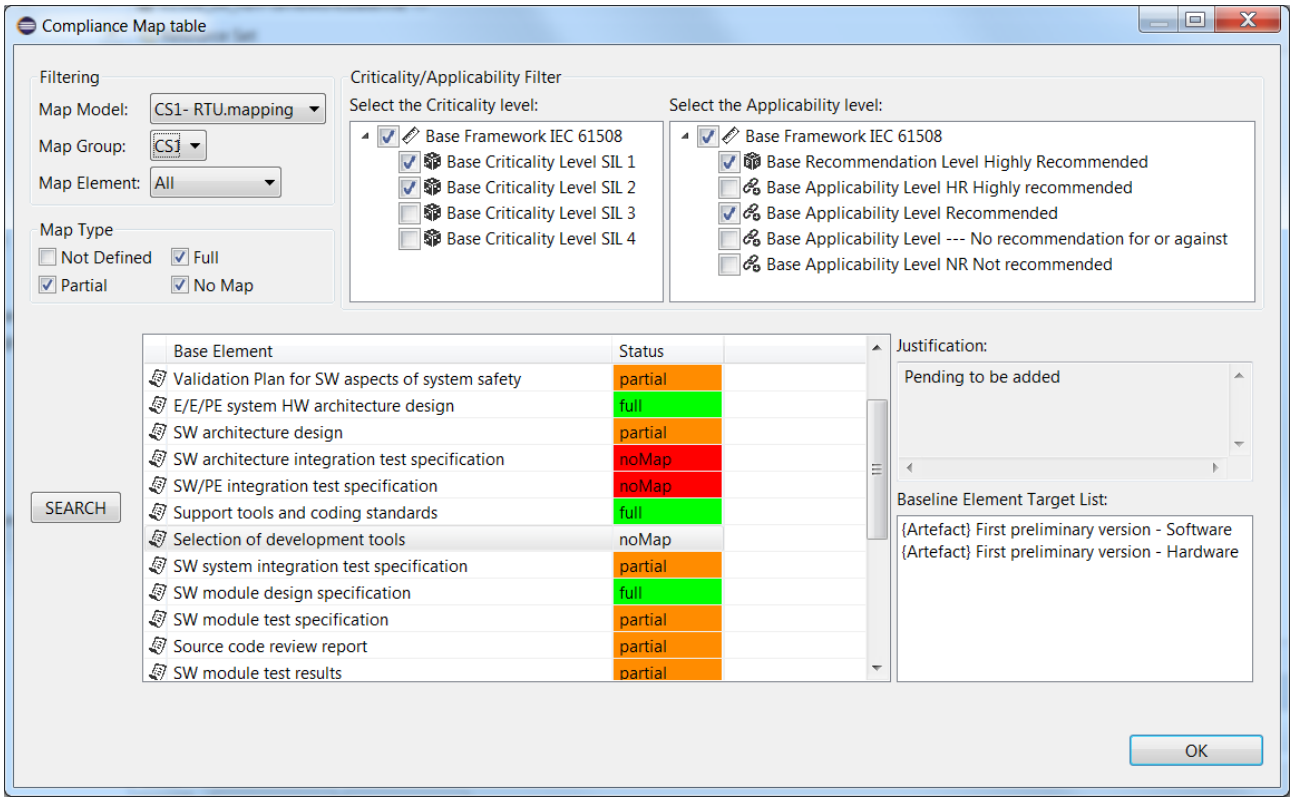


Figure 79 - Showing the target list of the Base element selected

Double clicking in any element of table will give access to the Compliance Map tailored functionality explained in the previous 6.4.2 section to allow the user create or modify the compliance map information of the double-clicked element.

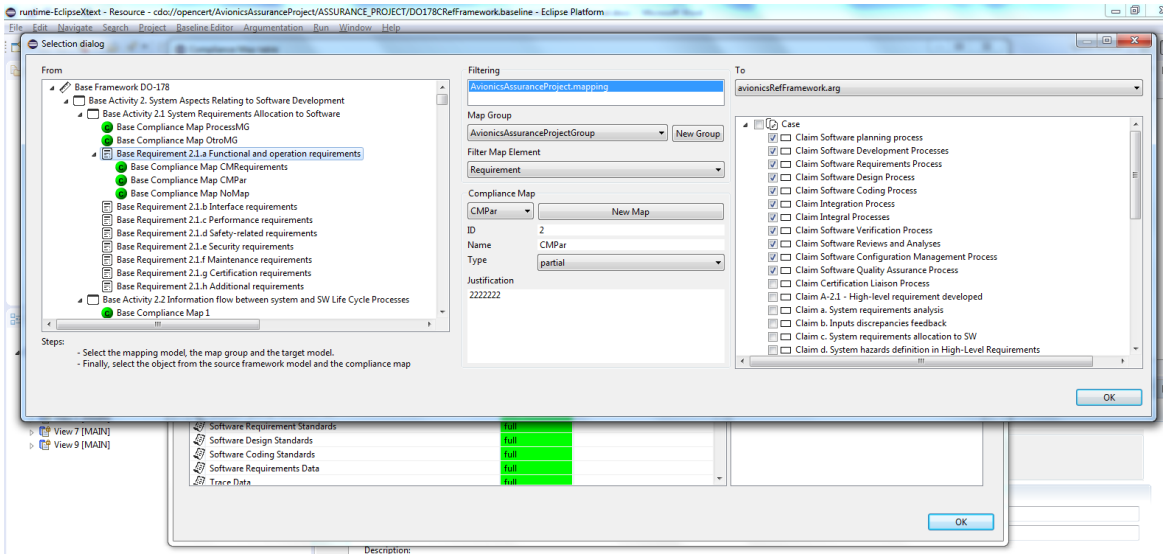


Figure 80 - Compliance editor accessed via compliance mapping table

Finally it's possible to double click in one element of the target list to access to detailed information of the select target element.

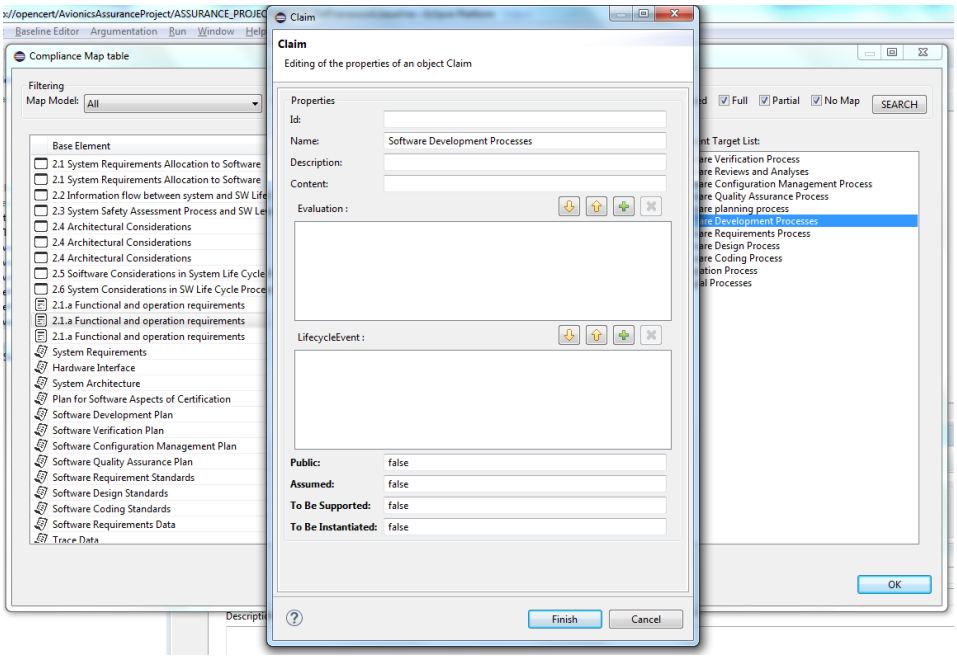


Figure 81 - Compliance map target element details accessed from compliance mapping table

6.5 Cross-Domain reuse

The cross-domain window objective is reusing the evidences from one source assurance project of one domain in a target assurance project of other domain. It's mandatory that the target assurance project is based in a reframework with equivalence maps with the reframework in which is based the source assurance project and, logically, the source project must have and evidence model.

To access this functionality open the target assurance project model and press the button "Cross Domain" on the properties form of the Assurance Project element of the model.

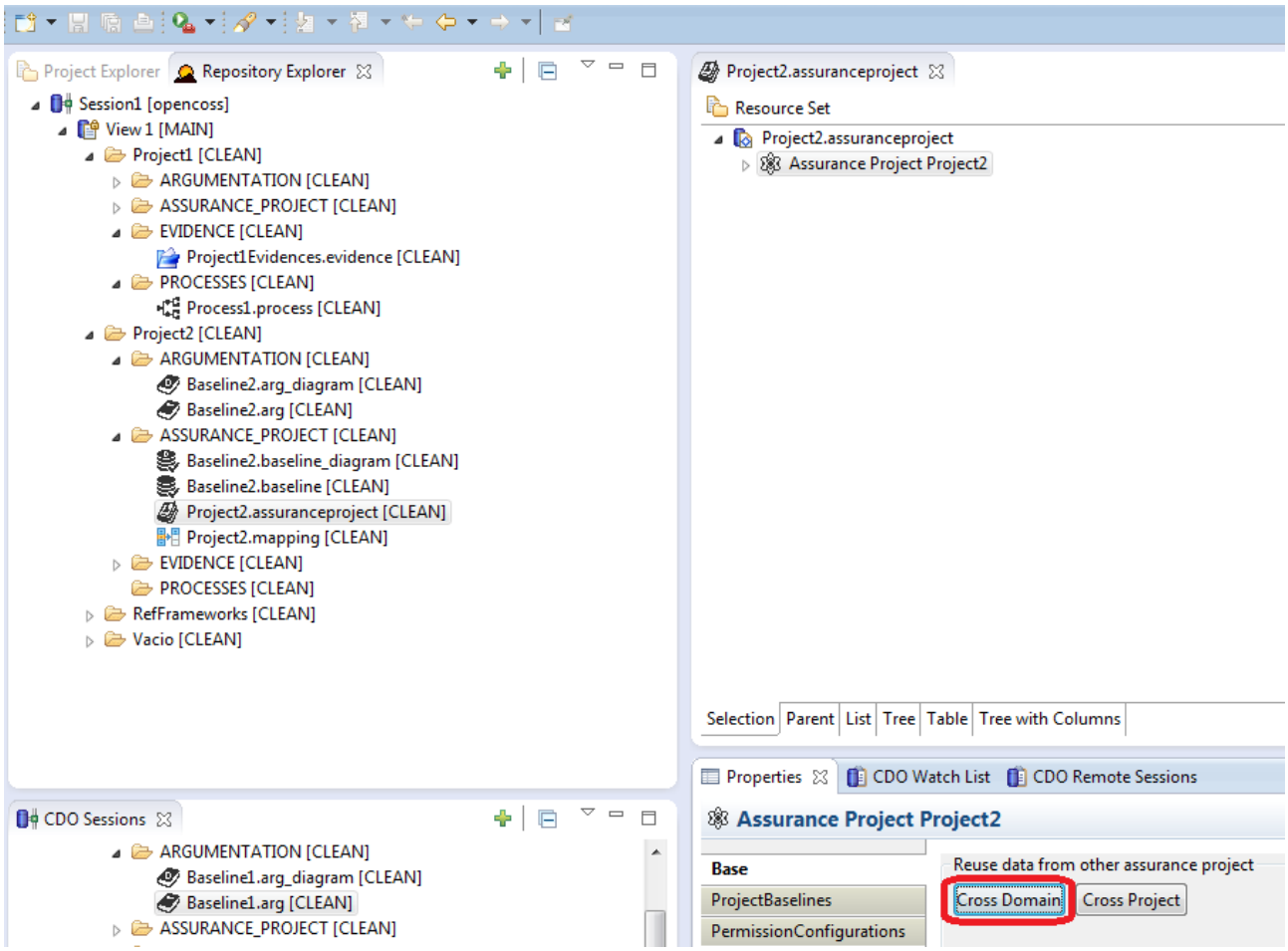


Figure 82 - Cross Domain button

If the target project doesn't have an evidence model a confirmation message will be displayed asking the user confirmation to create it.
If the user is agree, a new evidence model will be generated automatically based in the contents of the target assurance project baseline (for each BaseArtefact in the target baseline, one Artefact Definition with one Artefact will be included in the target evidence model), the evidence model will be related with the target assurance project and also the compliance maps between the baseline and the evidences are automatically created. After this the data generation process, the cross domain window will be opened.
If the user refuses, the data is not created and the cross domain window is not opened because is mandatory to have and evidence model as destination model of the reuse.

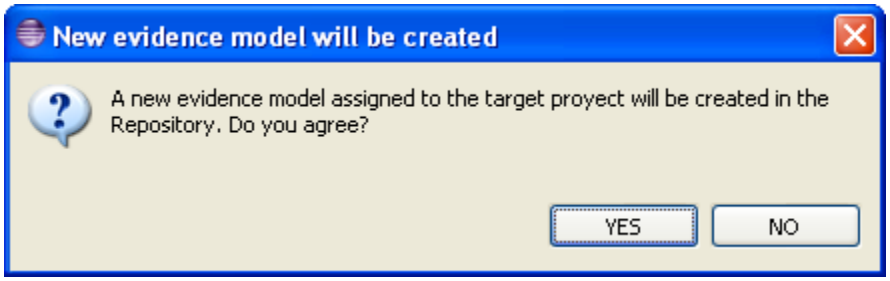


Figure 83 - Create a new evidence model message

If the target model already has an evidence model, the user is asked if he wants to use it as destination model of source Artefacts to reuse. If the user answers "Yes" the existing evidence model will be used as the target model of the reuse and the cross domain window will be opened. If the user says "No" the previously explained message will appear (see Figure above).

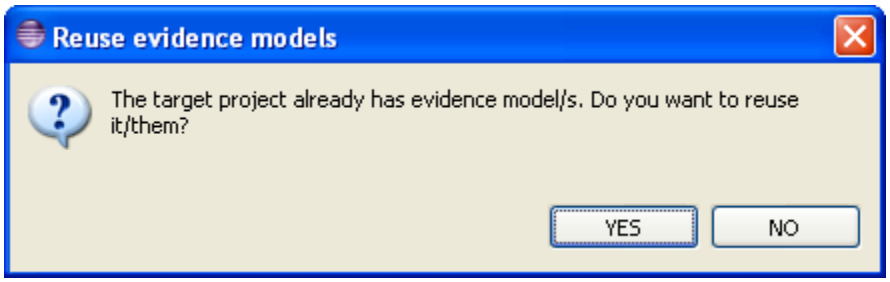


Figure 84 - Use existing evidence model message

The Cross Domain window is organized in three zones:

- The *left zone* shows information about the target project. In the top part the URL of the target assurance project, above a tree with the target baseline contents, above the compliance map information of the target baseline element selected and the contents of the target evidence model in other tree.
- The *middle zone* displays equivalence map information. It includes controls to select the equivalence mapping model and the equivalence map group, display the equivalence map details of the target baseline element selected and its postconditions in a list (to see the ID, Name and description one postcondition must be selected).
- The *right zone* presents information about the source project. In the top part the URL of the source assurance project, above a tree with the source baseline contents, above the compliance map information of the source baseline element selected and the contents of the source evidence model.

The user can obtain detailed information of any element displayed in the trees in a popup window by double-clicking over it. In the case of the target evidence model, the model can be edited directly in the popup window and the changes will be saved after clicking the “Save target evidences” button.

Also is possible to create new Compliance Maps between the target baseline and the target evidence model using the “New CM” button connected with the tailored Compliance map window explained in Section 5.4.2

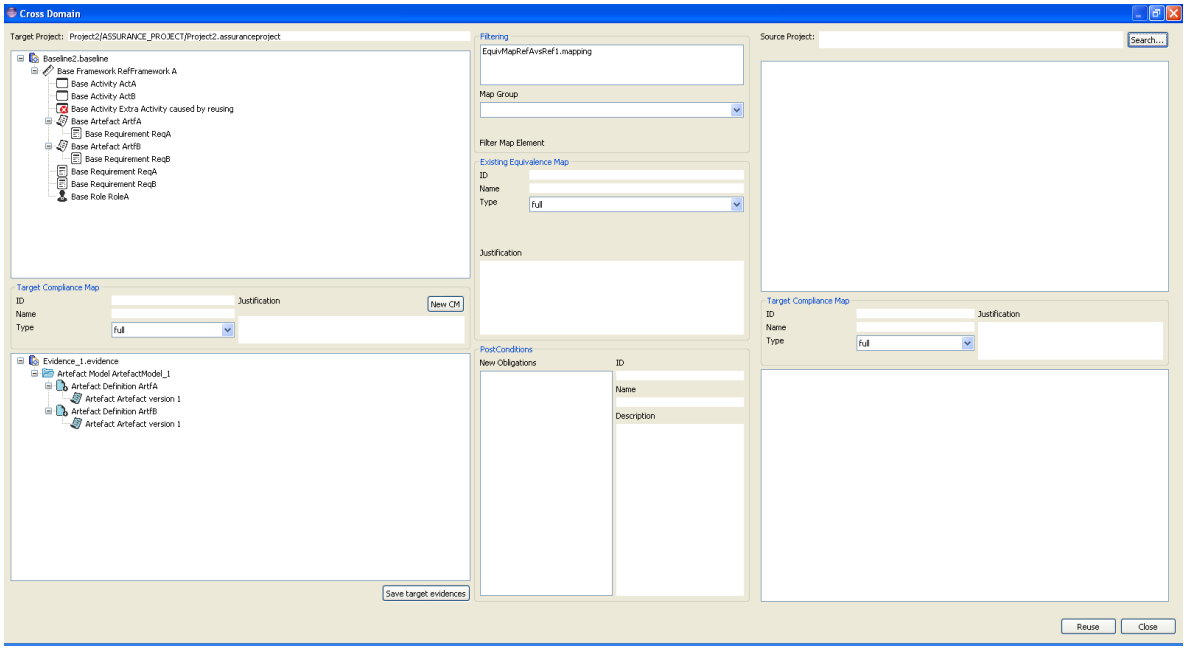


Figure 85 - Cross domain window

The user has to choose the source project of the reuse using the “Search button” and the source baseline and evidence model tree will be loaded. After this, has to select the equivalence model and the equivalence group. The next step is to select the target base element that will receive the evidences to be reused and its compliance and equivalence map information will be loaded, highlighting in green its target elements in the trees. Finally the user has to select the target Artefact and press the “Reuse” button to start the copy of the checked source Artefact/s to the target selected Artefact (only one can be selected).

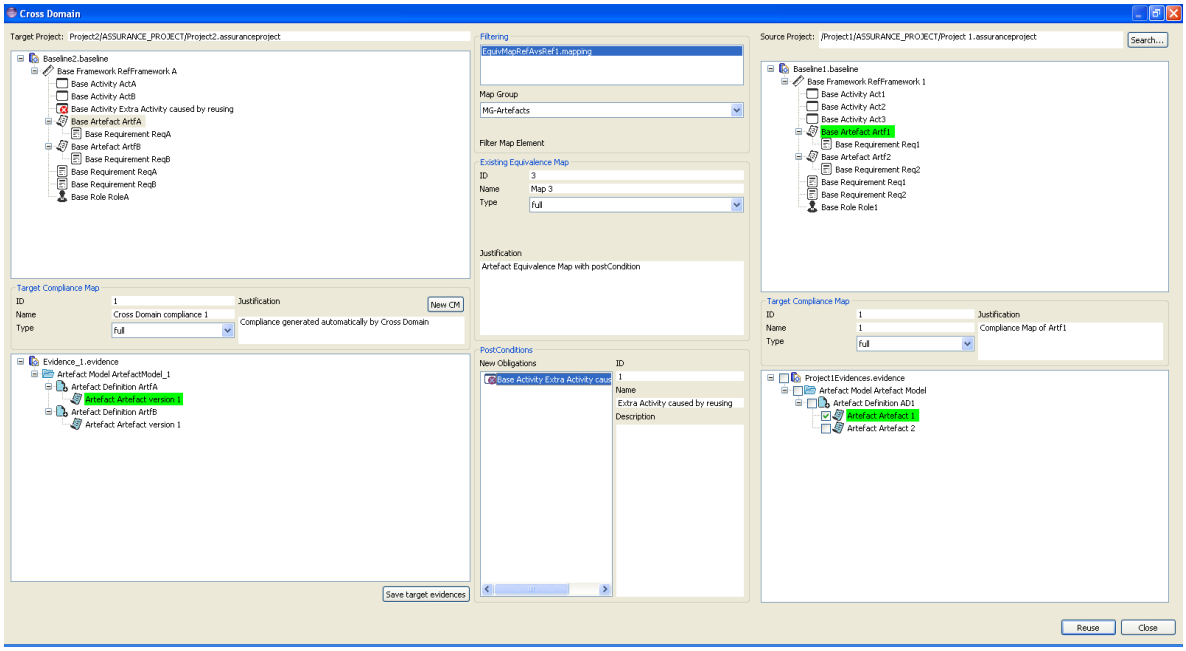


Figure 86 - Cross domain window with base element selected

As example the screenshot above shows that the selected BaseArtefact “ArtfA” of the target project “Project2” has a Compliance Map with the target Artefact “Artefact version 1” of the evidence model of the target project. Also has an equivalence map, inside the map group “MG-Artefacts”, with the source baseline BaseArtefact “Artf1” of the source assurance project “Project1” and as postcondition the BaseActivity “Extra Activity caused by reusing” (the equivalence maps are created at reffframework level and are copied to the baselines during the assurance project generation process). The source BaseArtefact “Artf1” has a Compliance Map with the target Artefact “Artefact 1” of the evidence model of the source project. Therefore, the “Artefact 1” is a good candidate to be reused, according to the existing equivalence and compliance mapping information, and appears checked and highlighted in green.

This window checks the integrity of the data before start the copy process, for example:

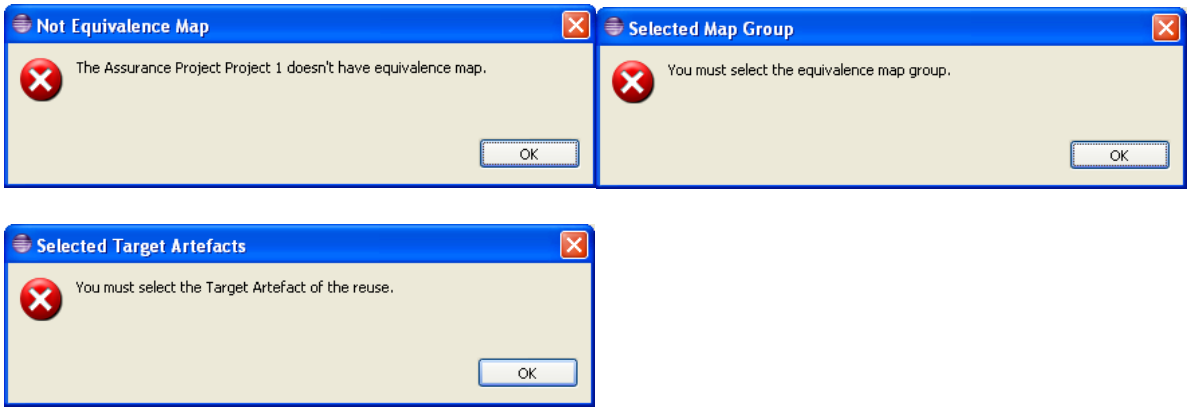


Figure 87 - Cross domain information messages about integrity

If the user wants to copy artefacts without equivalence between them, a confirmation message is showed.

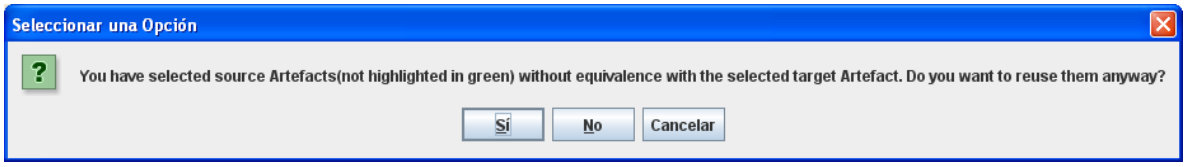


Figure 88 - Reuse not equivalence artefacts confirmation message

Finally, if all is correct another confirmation message with the resume of the data that will be copy is displayed.

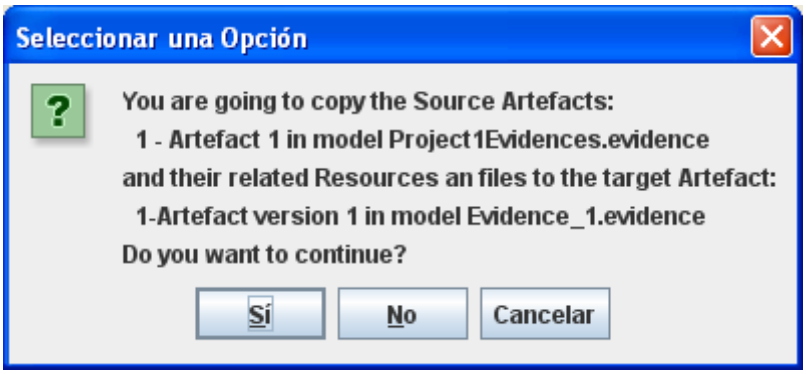


Figure 89 - Cross domain final confirmation message

If the user continues the copy process will begin. The source repository configuration information inside the Artefact Model Object, the Resource objects of the checked source Artefacts and the repository files related to these resources will be copied to the target evidence model. Additionally, the postconditions will be selected in the target baseline model.

6.6 Cross-Project reuse

The cross-project window objective is reusing models from one source assurance project to a target assurance project, and also the diagrams will be copied to the target project if exists.

This window allows reusing only the selected source evidence models associated to the active Assets Package, because evidences are not related to any other model of the project, or all the baselines associated to the active Baseline Config and all the evidence, argumentation and process models of the active Assets Package. In this second option all the models will be cloned to assure the integrity of the data, for example, a baseline could be related with argumentations, evidences and/or processes and in this way we are sure all the related information is copied avoiding inconsistencies.

To access this functionality open the target assurance project model and press the button “Cross Project” on the properties form of the Assurance Project element of the model.

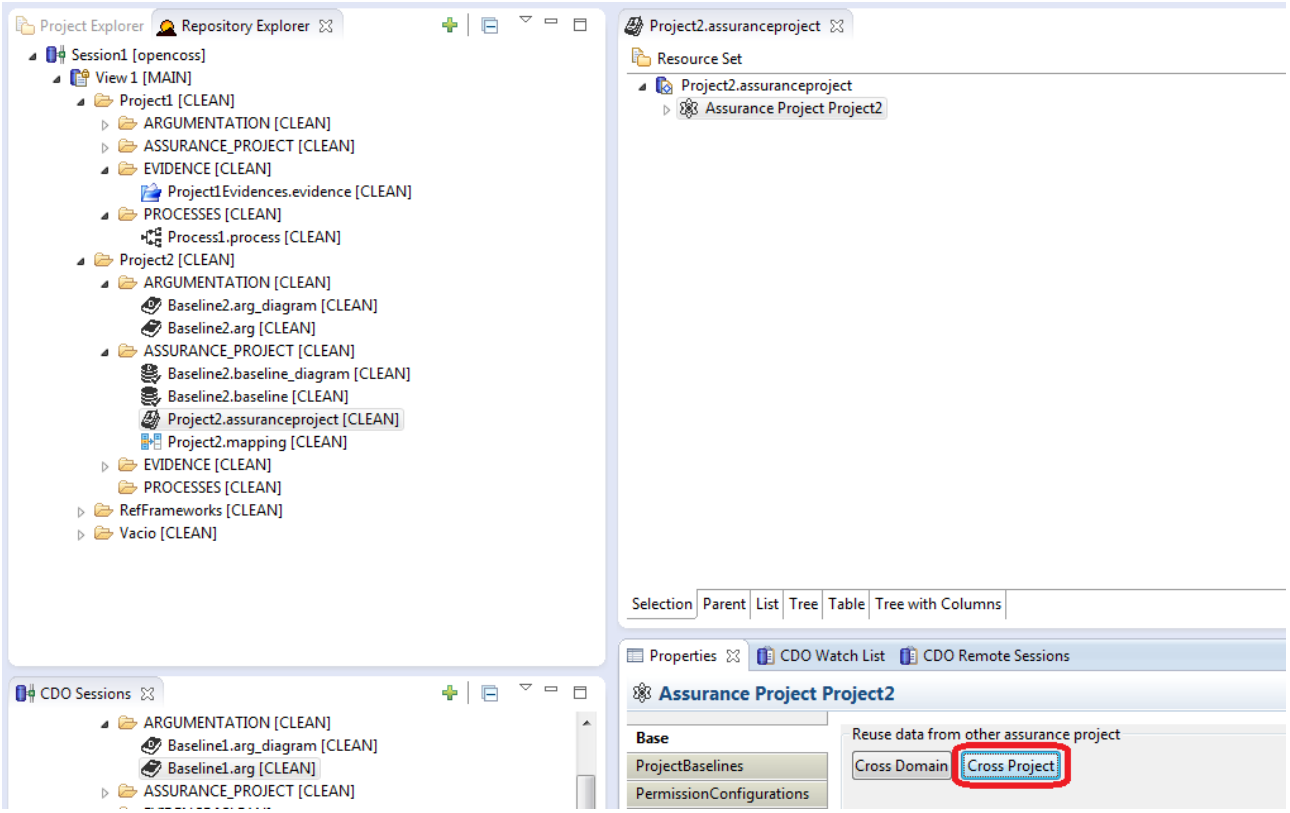


Figure 90 - Cross Project button

The user has to select the source project and after its models will be displayed in the window. As said before, only the models related to the active Baseline Config and active Assest Package of the source assurance project will appear.

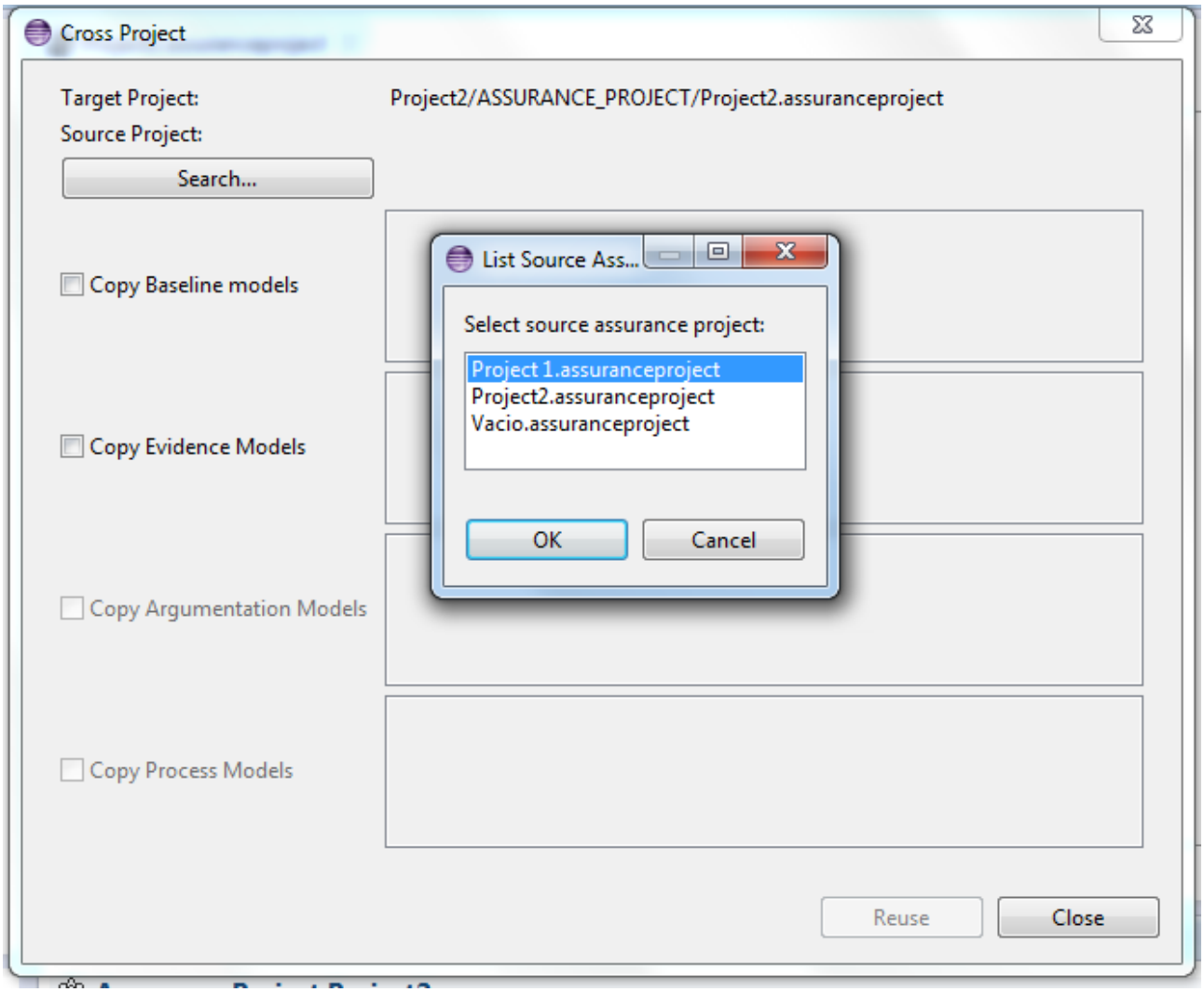


Figure 91 - Cross project: Source project selection

By default the “copy all models” option is selected but the user can uncheck the “copy baseline models” control to indicate that only wants to copy evidences. In this case, the user can select the desired models to copy from the evidence model list. To go back to the previous option only is necessary to check again the “copy baseline models” option.

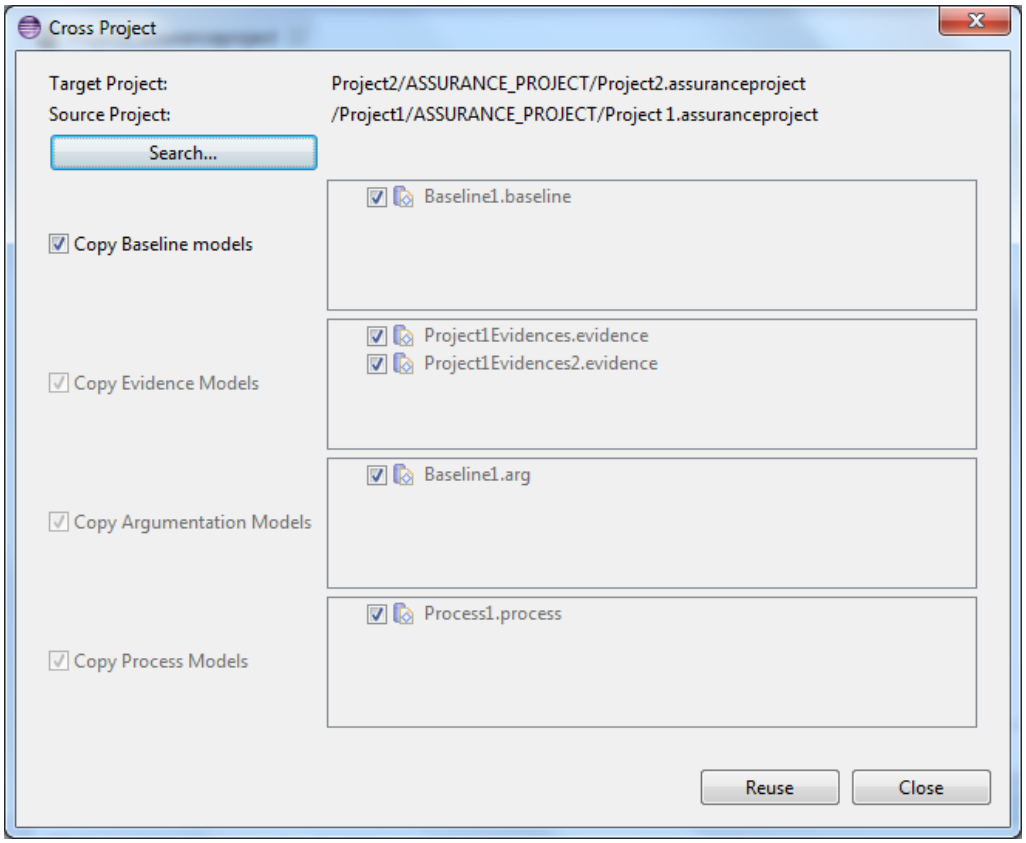


Figure 92 - Cross project: Copy all models

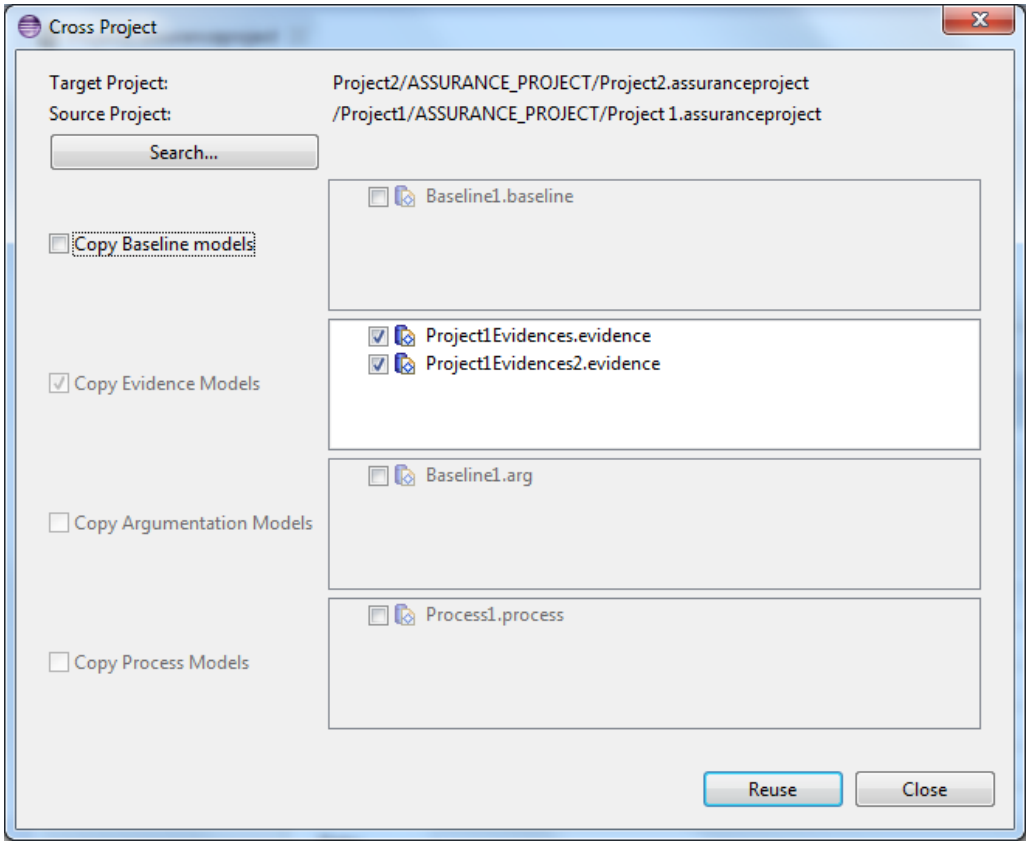


Figure 93 - Cross project: Copy only evidences

To begin the copy process the user must click over the Reuse button. In case of copying all the models, the information message in the screenshot below will be show to clarify to the user that the active Assets Package and Baseline Configuration of the target process will be changed. In case of reusing evidences, this message won't appear.

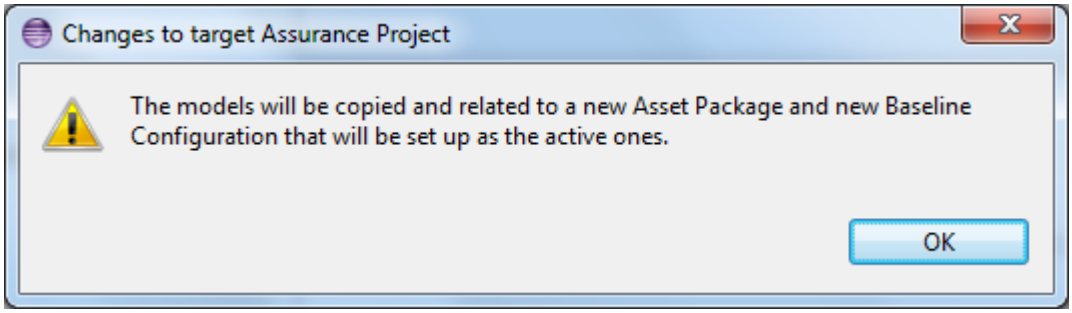


Figure 94 - Cross Project information message

When the copy process ends a message window will be shown.

In the next screenshot we can see, boxed in red, the new models copied and the new information added to the target assurance project model.

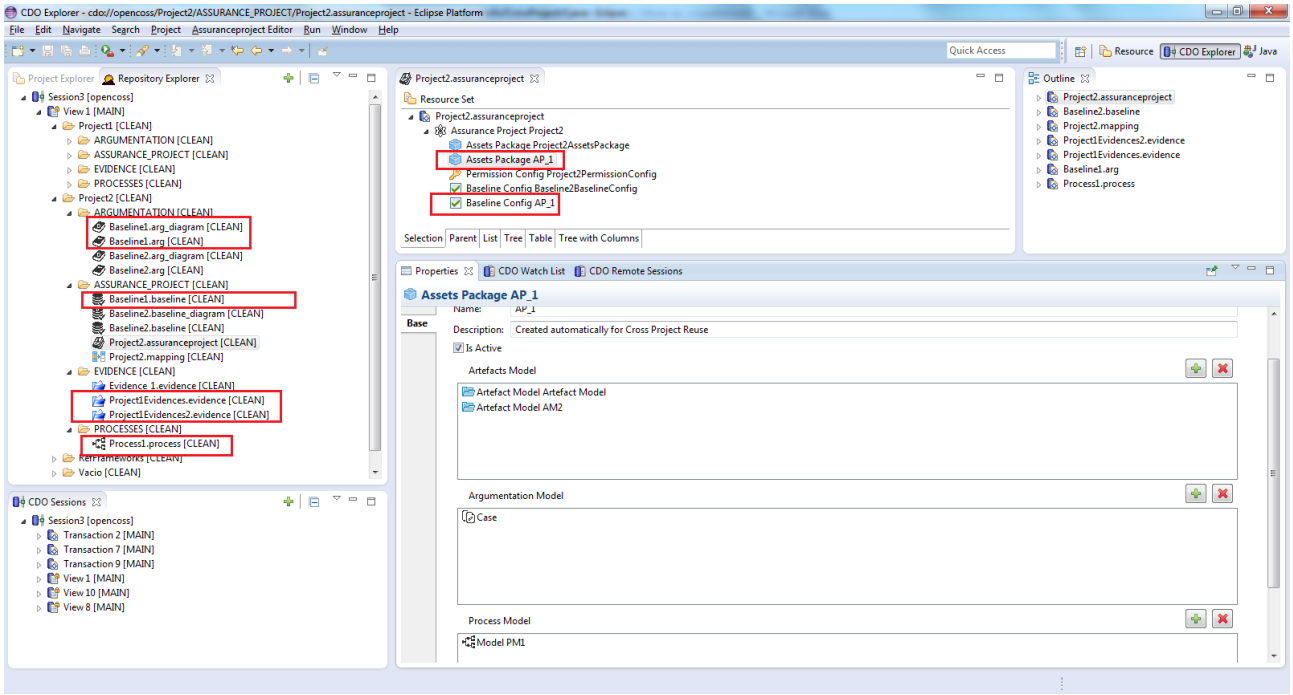


Figure 95 - Cross project reuse result

6.6.1 Cross Project Reuse View

It is also possible to reuse models from one source assurance project to a target assurance project using a specific view called "Reuse". This view is particularly useful to reuse a subset of model elements, which can be selected manually by the user. Future versions of the Reuse View will allow users to filter specific subset of model elements by different criteria such as those featured by the Reuse Discovery functionalities (see Section 12.3).

To open this view, the user must select it from the menu Windows-> Show View -> Reuse. It is also integrated in the set of OpenCert views and it will be already opened when the OpenCert perspective is activated.

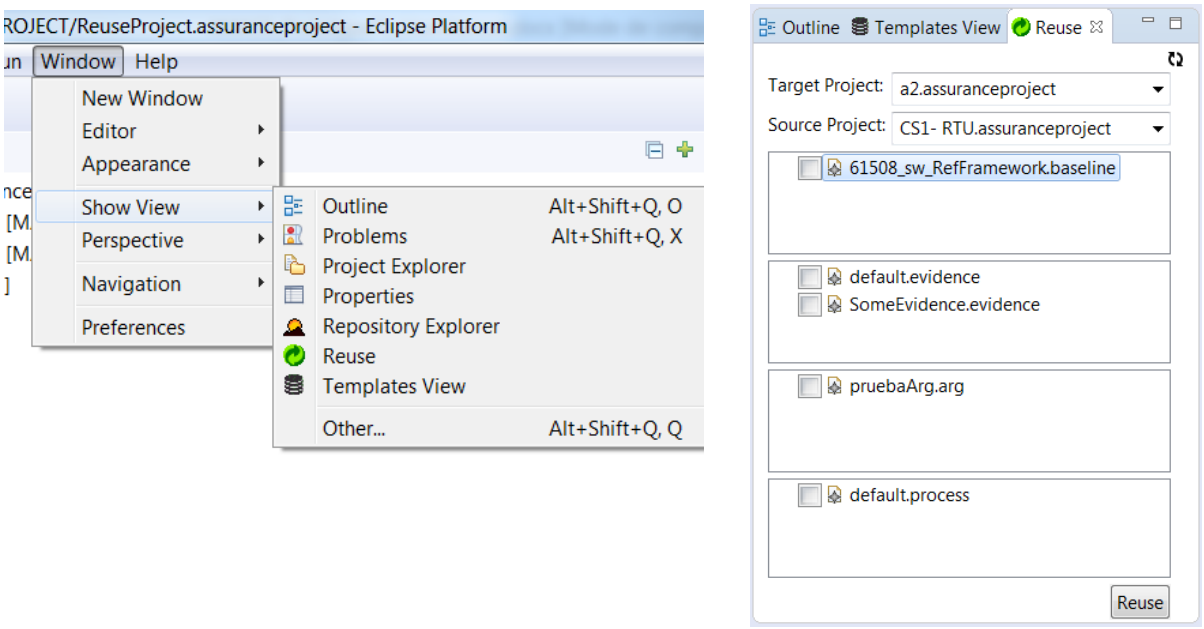


Figure 96 - Cross-project Reuse View

The procedure to perform cross-project reuse with this view is as follows:

1. The user must select the Target (Assurance) Project and the Source (Assurance) Project from the respective drop-down lists at the top of the Reuse View.
2. Once the Source Project is selected, a set of associated models are shown in the four lists below. Select the models to be reused by using the check option. The lists include the following four kind of models:
    a. Baseline models (name extension .baseline)
    b. Evidence models (name extension .evidence)
    c. Argumentation models (name extension .args)
    d. Process models (name extension .process)

    The list of models correspond to the active models only (Baseline Configuration and Assets Package elements from the Source Project with the active option checked.
3. Double clicking over any model will open its corresponding editor (see *Figure 97*).
4. Using the "Tree with Check Boxes" tab of the opened model allows users to select the model elements that will be reused.

    Important note: Since there is a bug in the application, and while we try to solve the issue, you will need to close and re-open the Reuse View to see this tab. We will solve this issue as soon as possible.
5. Finally pressing the Reuse button, the copy of data from source to target project will be performed.

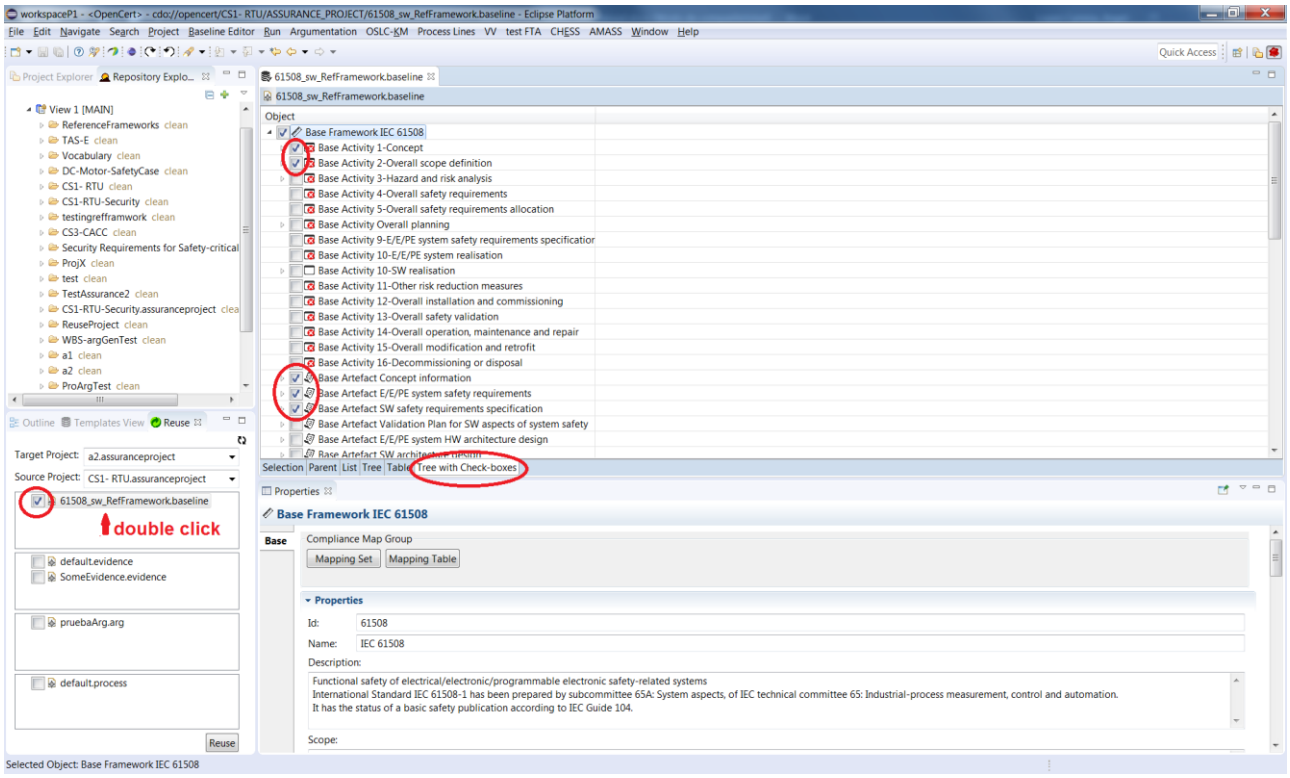


Figure 97 - Using the Reuse View

6.7 EPF to OpenCert Transformation

As described in Section 4, EPF can be used for modelling the definition and planning of processes. OpenCert also allows users to model processes (see Sections 9 and 9.9) but looking at post-planning phases. We can get benefit of the EPF process information to create a first view (which can evolve during an assurance project) of process models in OpenCert by importing EPF information into OpenCert. Specifically, the transformation process takes a delivery process modeled in EPF and generates an evidence model and a process model in Opencert. Models of the EPF composer are described according to the UMA metamodel [6], while process and evidence metamodels are part of CACM. The main mappings between these metamodels are described in Table 1 and Table 2.

A Delivery Process in EPF is contained in the metamodel class ProcessComponent which provides additional information to the process description like its version, authors or team profiles required for the execution of the process. The generated ProcessModel only contains information described in the context of the ProcessComponent. The user does not explicitly require creating a ProcessComponent in the EPF composer; they are automatically created each time a delivery process or capability pattern is created. Moreover, they are invisible in the tool.

UMA	CACM Process
ProcessComponent	ProcessModel
TaskDescriptor	Activity
Activity	Activity
CapabilityPattern	Activity
RoleDescriptor	Person
Guideline	Technique
Practice	Technique

ToolMentor	Tool
RoleSet	Organization
TeamProfile	Organization
WorkProductDescriptor	Artefact

Table 1. Main mappings between UMA and CACM Process

CACM and EPF distinguish between artifact definition and artifact usage by means of the use of corresponding metaclasses (Artefact and ArtefactDefinition in CACM, and Artifact and WorkProductDescriptor in EPF). However, the semantic of these concepts is slightly different in both metamodels. In the case of CACM, ArtefactDefinition is a template of a work product involved in an activity, and Artefact represents one specific work product involved in the activity that uses that template. On the other hand, in EPF, Artifact is an element that belongs to the Method Content package (i.e. the library of reusable process elements) and WorkdProductDescriptor is an instantiation of an artifact in the context of an activity. In order to match these concepts, we focus the transformation in WorkProductDescriptor (see Table 2). So, from a WorkProductDescriptor is generated an ArtefactDefinition and an initial version of Artefact.

UMA	CACM Evidence
ProcessComponent	ArtefactModel
WorkProductDescriptor	ArtefactDefinition
WorkProductDescriptor	Artefact

Table 2. Main mappings between UMA and CACM Evidence

This section describes how to import information contained in EPF models into OpenCert models by following two steps: (1) EPF Export to XML files, and (2) Import EPF XML Files into OpenCert.

6.7.1 EPF Export to XML files

Process modeled in EPF can be imported in OpenCert using the XML export functionalities available in this tool (See Section 10.7 of the EPF Manual [1]). Specifically, we need two documents to make this import possible, the Method Library and the Method Configuration that contains our process. In the following, we will explain how to get these documents.

In order to export our library in XML format, we select “File -> Export” and the dialog shown in Figure 98 appears. In this dialog, we select XML and click “Next”. In the next dialog, we select “Export the entire method library” and click “Next”. In the last dialog, we enter a name and location for the XML file and click “Finish”. The button “Browse” can be used to navigate to the location of our choice.

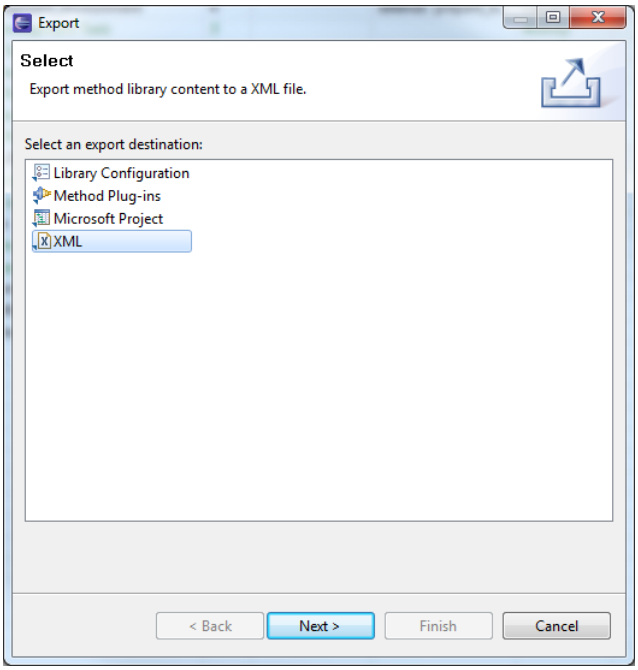


Figure 98 - Export wizard of the EPF composer.

In order to export our process, we need to define a configuration (detailed explanation in Section 4.2.11 of EPF manual). To do so, in our library, we right click “Configuration” and select “New -> Configuration” (see Figure 99) and a view for the modeling of the Method Configuration automatically appears.

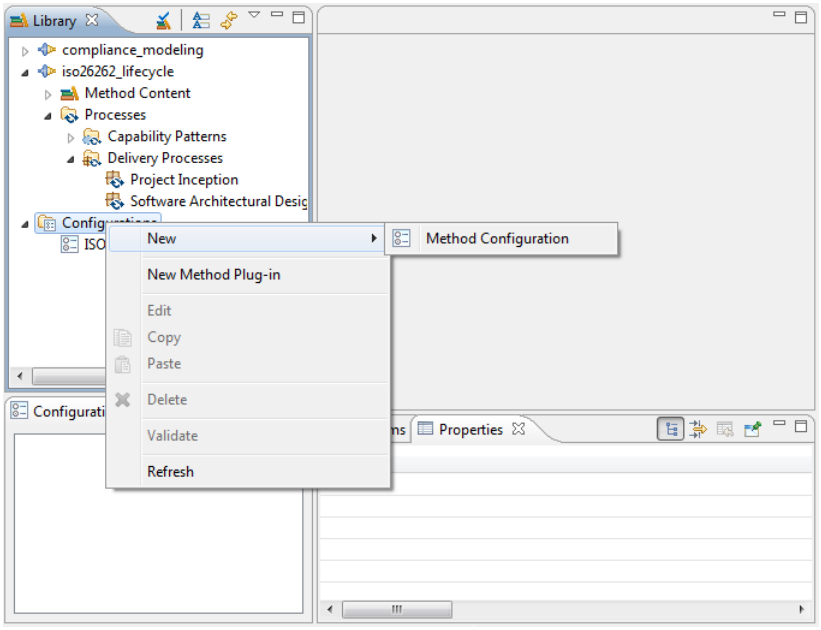


Figure 99 - Creation of Method Configuration in EPF

In the Method Configuration view, we give a name to the method configuration and click in the tab “Plug-in and Package Selection”. In the Content field of this form will appear all the plug-ins defined in our Method Library (see Figure 100). We expand the different plug-ins and folder to locate our Delivery Process and check just the Delivery Process that we want to import in OpenCert. In this case, we have imported “Project Inception”. Then, we check that there are not errors in the export by clicking in the button surrounded in red and we fix the errors by clicking in the button surrounded in black.

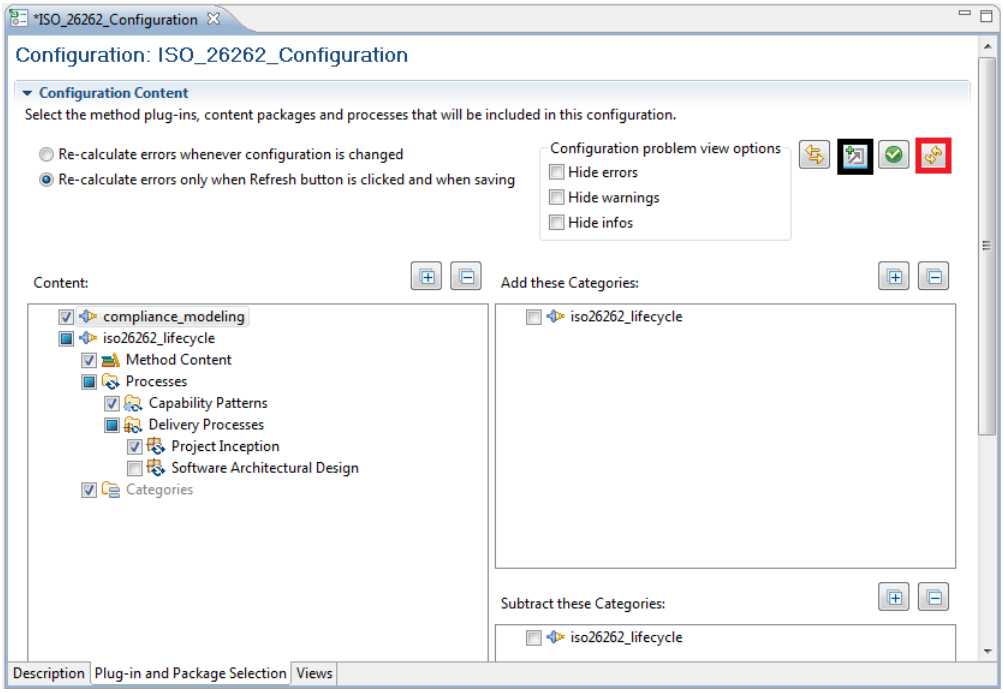
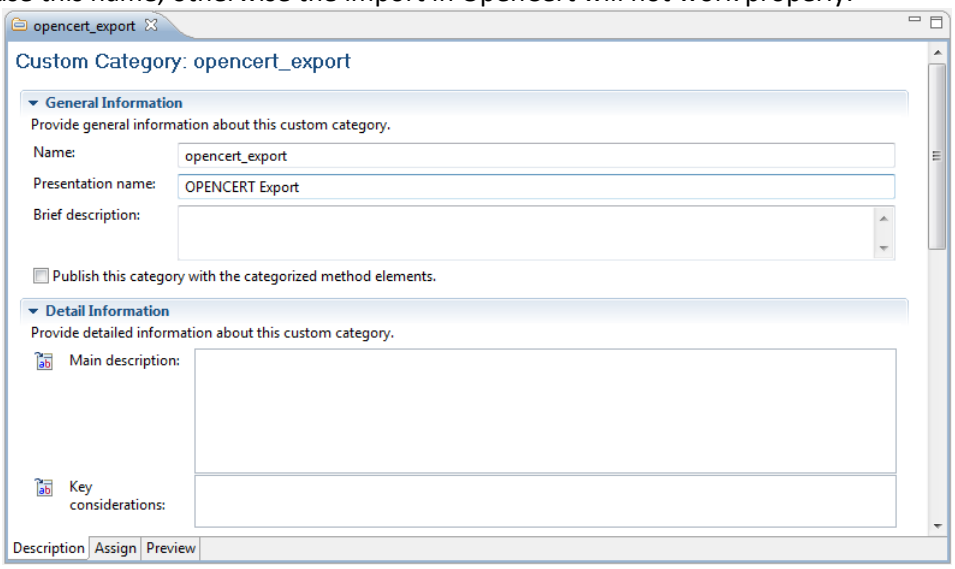


Figure 100 - Plug-in and Package selection of Method Configuration View.

In order to export the configuration, we follow the same steps as for the export of the Method Library but in the second step of the wizard we select "Export a Method Configuration" and in the next screen, we select the corresponding method configuration, in our case, ISO 26262 Configuration.

In case that we have more than one process in the Delivery Processes folder of our plug-in, the XML export functionality will export all of them. This is a bug of the EPF composer that has been reported. In order to overcome this problem in the meantime, we must define a View in the configuration. With this goal, we define a Custom Category in our plug-in (Section 4.5.3 of EPF Manual). To create a custom category, we expand the Method Content package of a plug-in, right click in the Custom Categories and select **"New -> Custom Category"**. In the Description tab, we set as a name of the category "opencert_export". It is very important to use this name; otherwise the import in OpenCert will not work properly.



In the Assign tab, we click "Assign…" and a dialog to select our delivery process to be exported will appear. We expand our plug-in and select the process to be exported, in this case "Software Architectural Design".

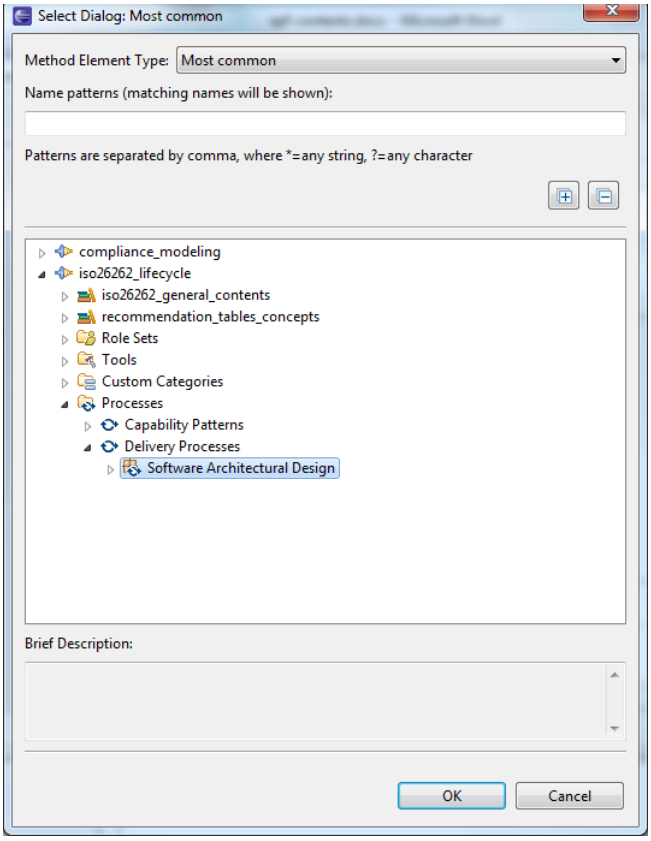


Figure 101 - Assign dialog for Custom Category.

Finally, in the tab Views of our Method Configuration (see Figure 102), we click “Add view…” and dialog will appear. In the Custom Categories folder, we select the custom category that contains our delivery process, “Opencert Export”. Then, we export the Method Configuration as it has been previously explained.

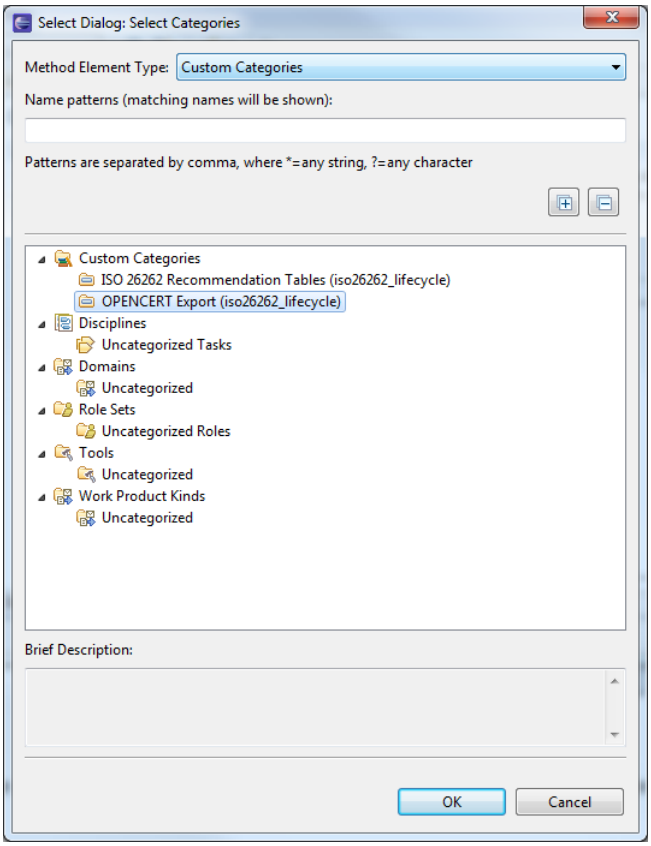


Figure 102 - Views tab of the Method Configuration view.

6.7.2 Import EPF XML Files into OpenCert

Once EPF models are exported to XML files, we are able to import that information into OpenCert. This functionality takes the two XML files exported in the previous step and creates two different models in OpenCert:

- **XML input files**: Method Library and Method Configuration.
- **OpenCert output models**: Evidence Model and Process Model.

To access this functionality, open the target assurance project model and press the button “Import from EPF” on the properties form of the Assurance Project element of the model.

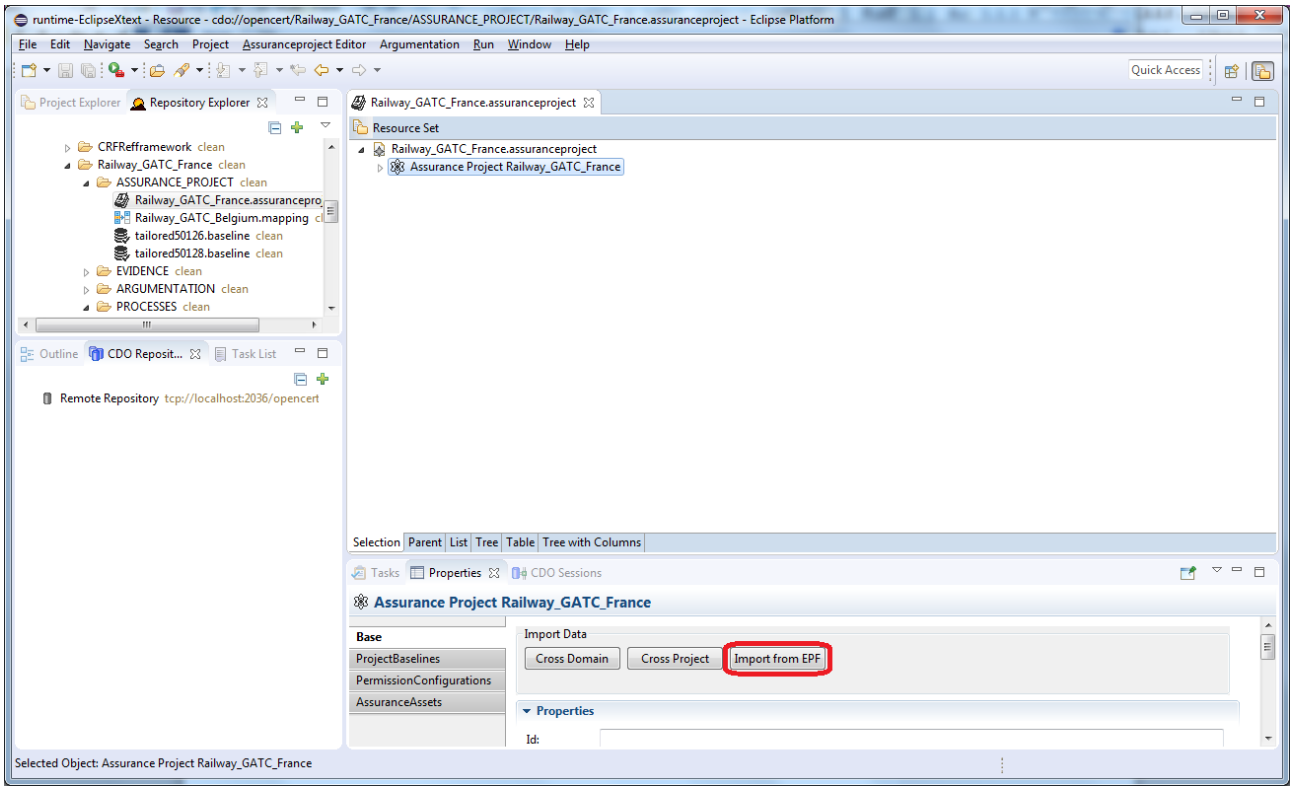


Figure 103 - Import from EPF button

The user must browse the EPF XML files that will be used as input for the importing operation. The browsing dialog filters the files with the ".xml" extension only.

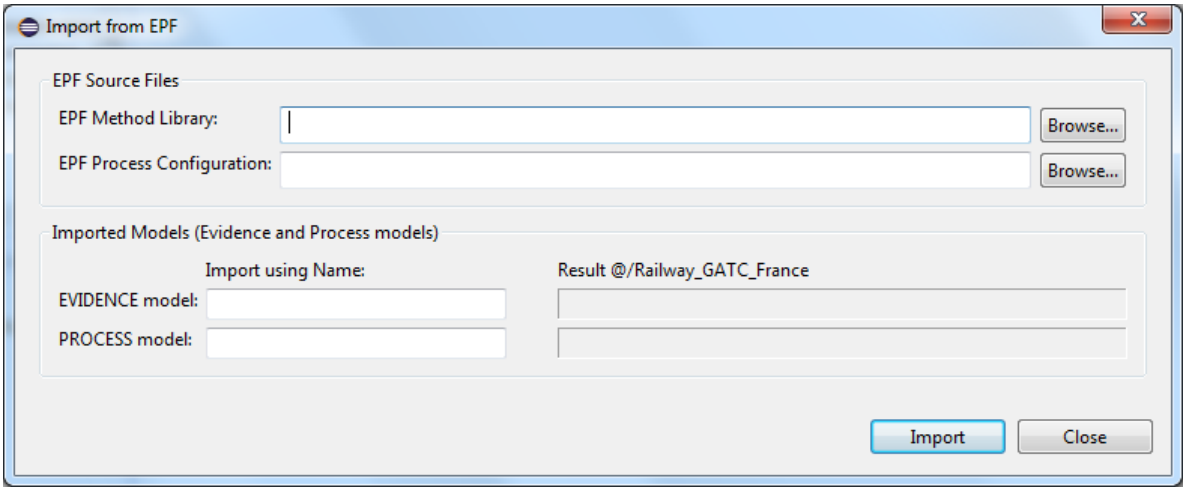


Figure 104 - Import from EPF: Source files selection

When the second file "EPF Process Configuration" is selected, the names for the models to be imported are proposed with the following format:

- EVIDENCE model: [EPF Process Configuration fine name] + "_Evid"
- PROCESS model: : [EPF Process Configuration fine name] + "_Proc"

These model names can be changed by editing the text boxes below the label “Import using Name:” before executing the Import operation.

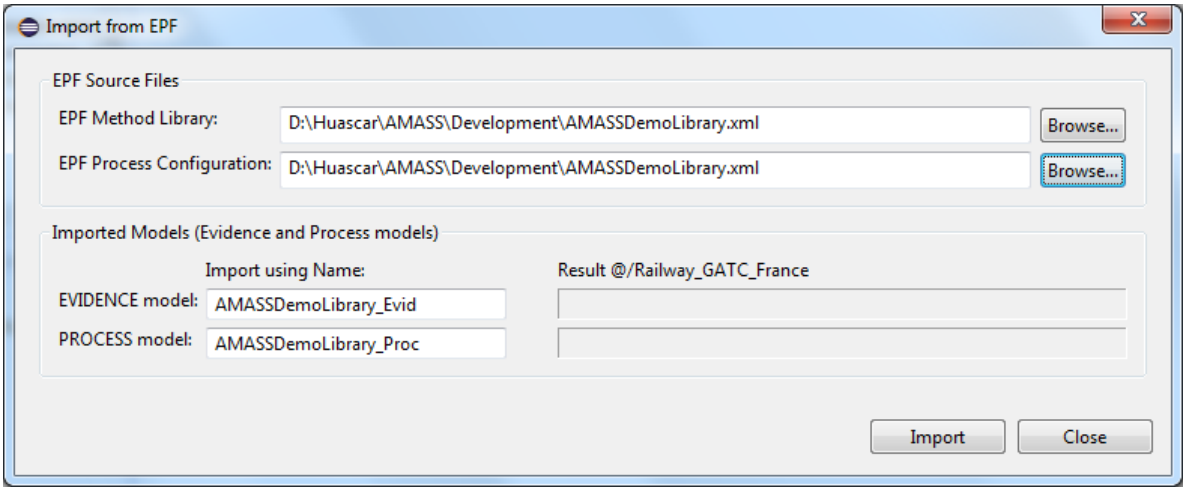


Figure 105 - Import from EPF: Imported model names specification

To perform the Import operation, push the “Import” button. At the end of the import operation, the generated models will be specified in the text boxes below the label “Result @ [Assurance Project folder]”. This completes the import process. If the transformation has errors, this will be indicated in the Result boxes. For the moment, we do not have a model transformation validation functionality to provide a diagnosis of the problems.

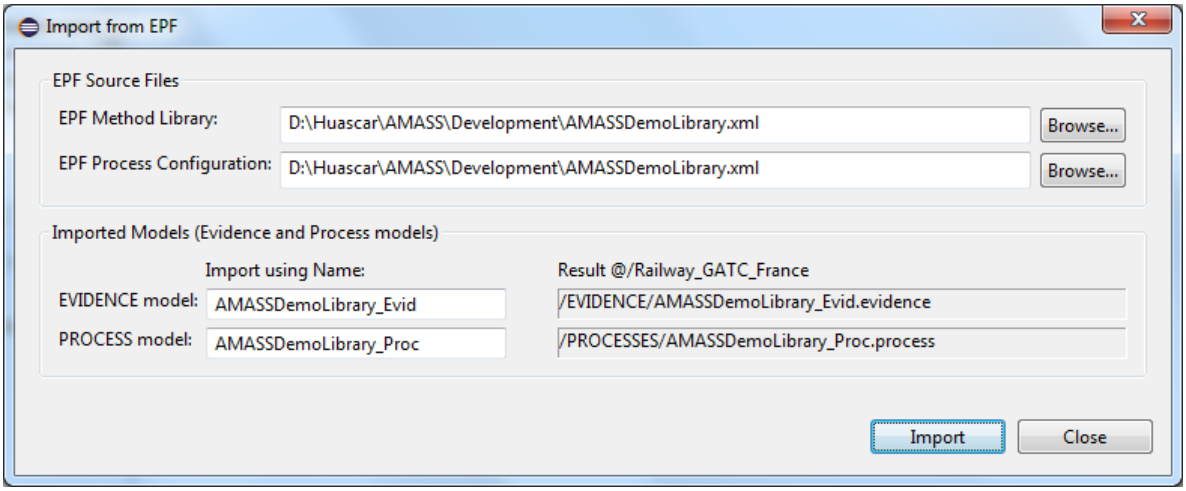


Figure 106 - Import from EPF: Result of the import operation

Please note that the operation can be executed again if the user selects other EPF input files. To finish the process, press the button “Close”.

Please note that the imported EVIDENCE and PROCESS models will be automatically linked to the assurance project by means of the Assurance Package element as shown in the following figure.

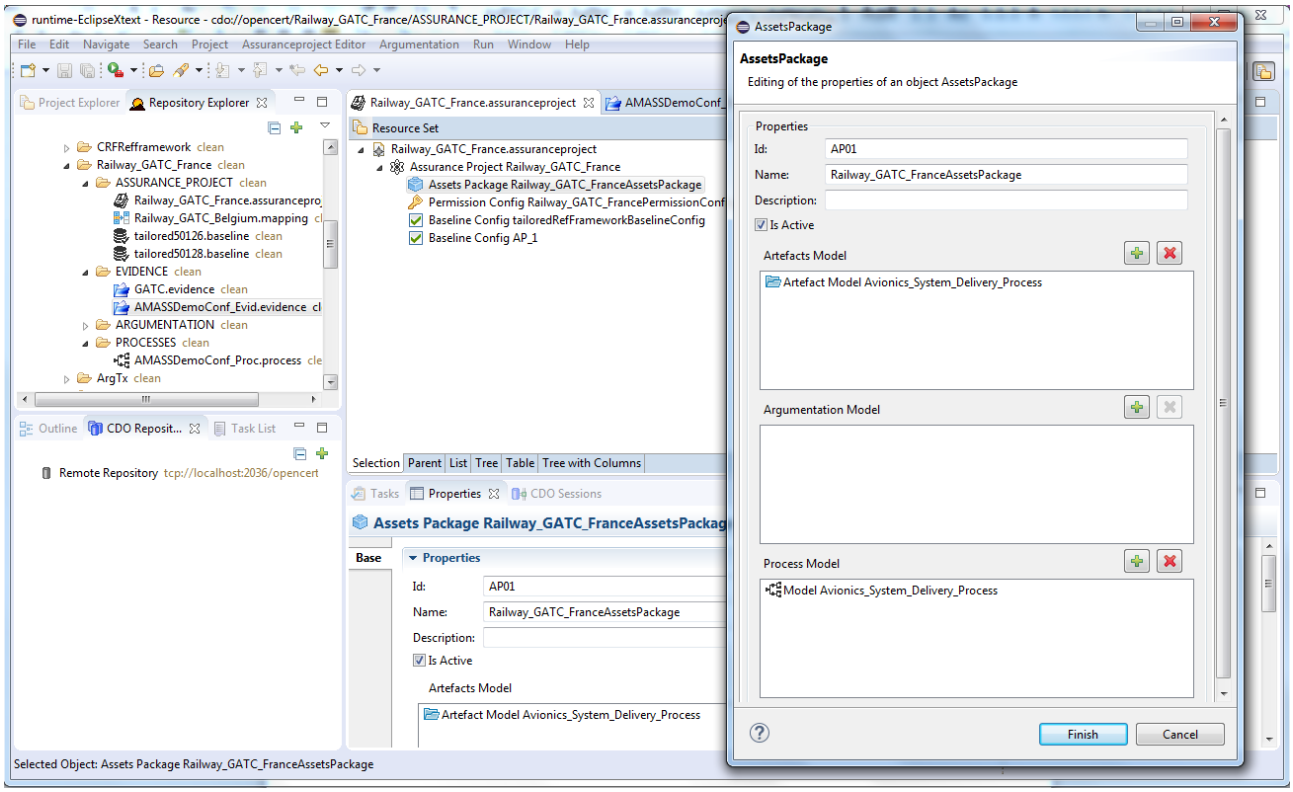


Figure 107 - Import from EPF: Imported models automatically linked to the Assurance Project

6.8 Creation of Mapping Model

The management of Mapping must be made through the creation of a new model of the type Mapping **Model**.

In order to generate a new Mapping Model, the following steps need to be done:

- First, select the entry of the menu *File -> New -> Other*:

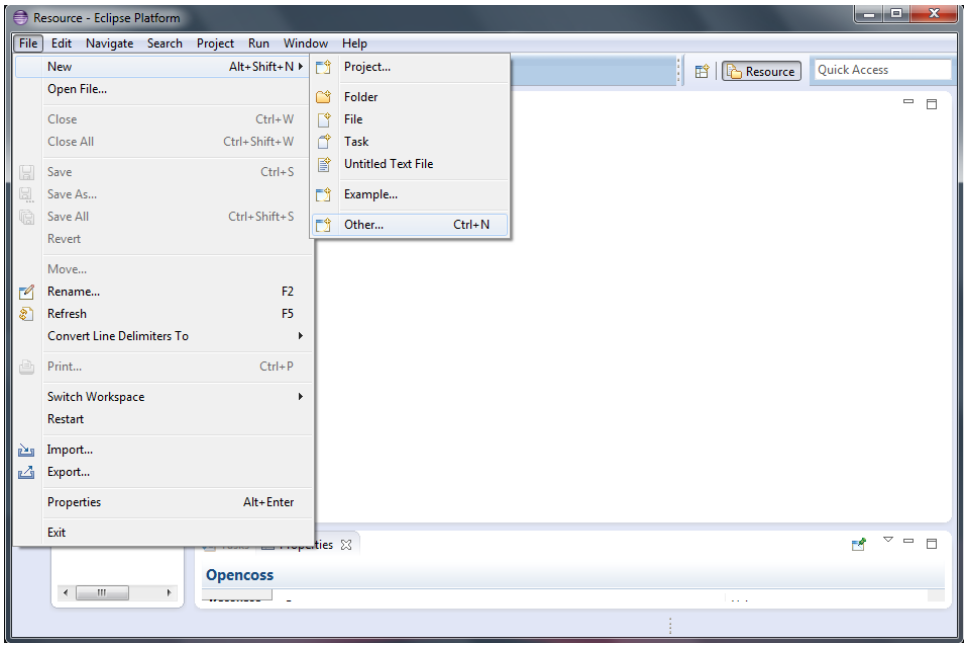


Figure 108 - New Property Model menu File -> New -> Other

- Inside the category of the wizard *AMASS* , select the Mapping *Model* and press the *Next* button:

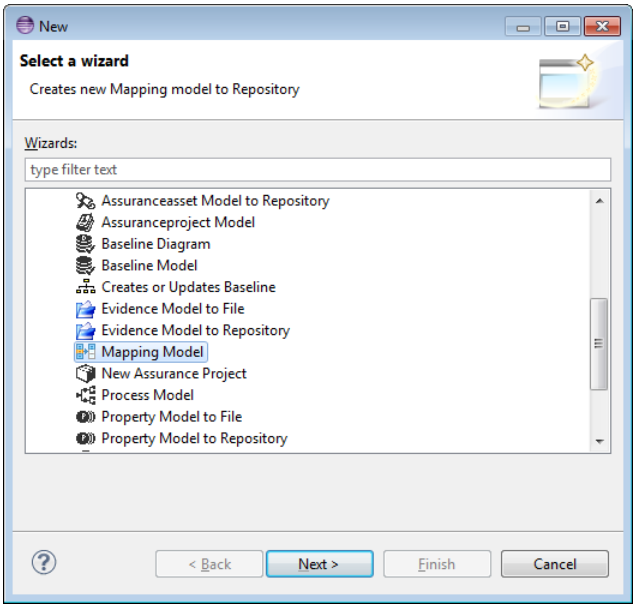


Figure 109 - New Mapping Model I

- Enter o select the parent folder, the resource name and press the Next button:

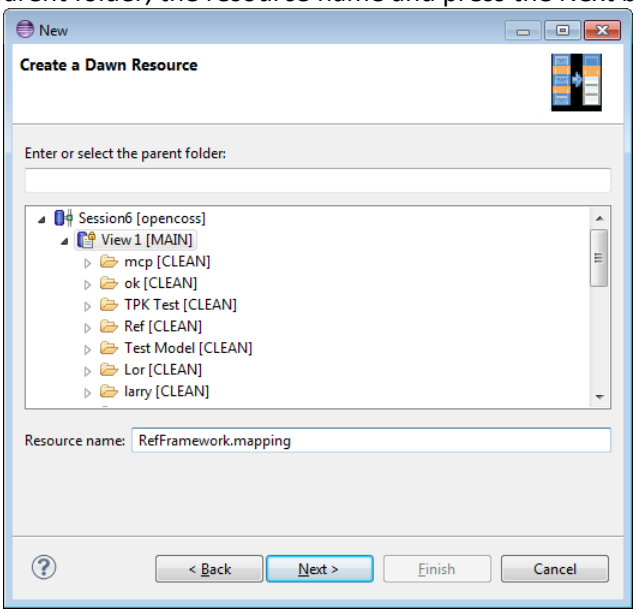


Figure 110 - New Mapping Model II

- And finally, select the “Map Model” object to create.

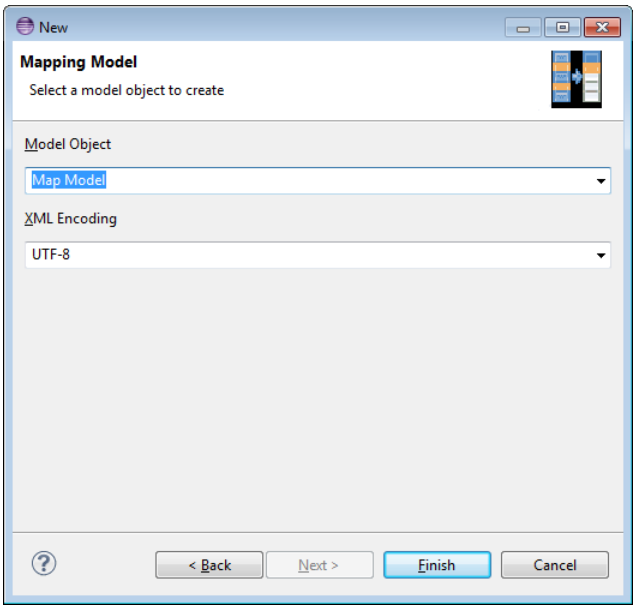


Figure 111 - New Mapping Model III

Once the Mapping Model has been created, the first item is presented to the user.

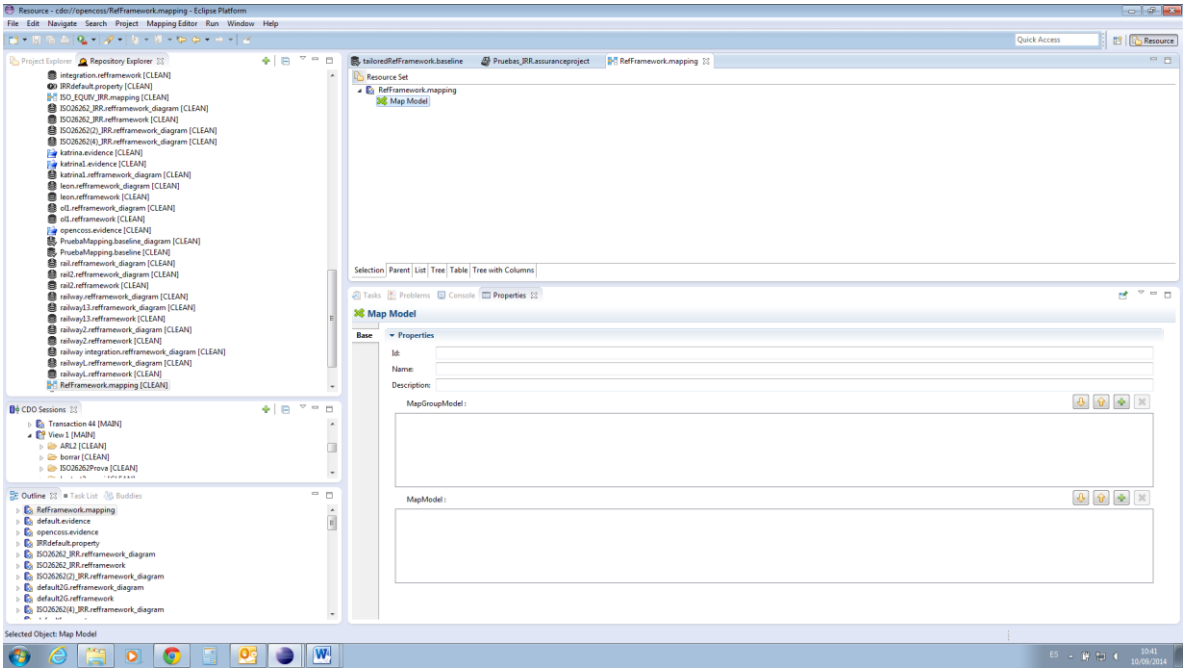


Figure 112 - Mapping Model

6.9 Map Group edition

6.9.1 Add a map group

It is possible to add map groups to a mapping model in two ways:

- Select the model element, press the right button of the mouse and select the contextual menu *New Child –> Map Group*

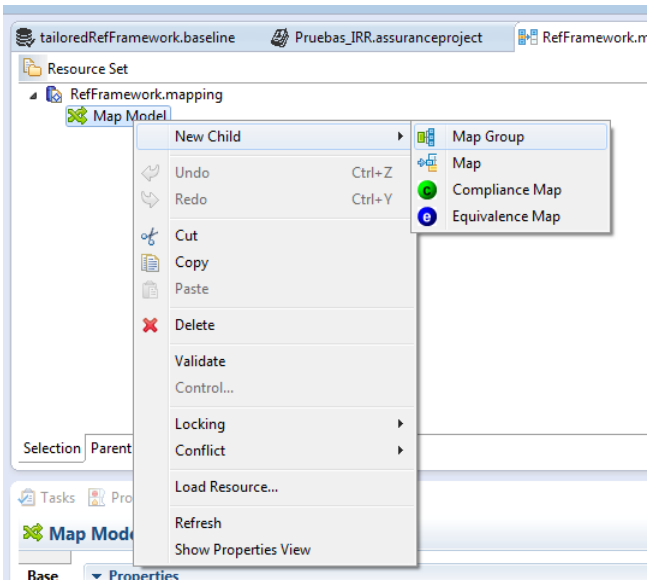


Figure 113 - Add New Map Group (I)

- Or, select the model element, and press the icon button *in the base tab*

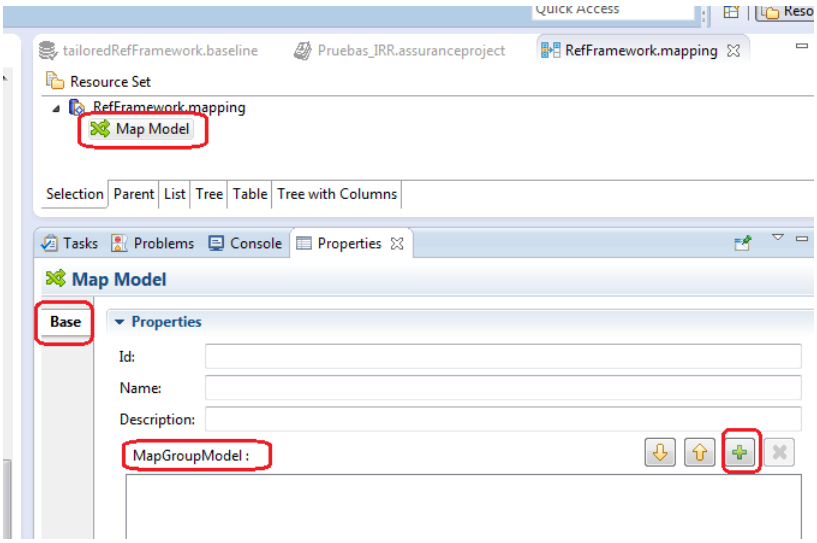


Figure 114 - Add New Map Group (II)

After these actions, in the properties zone, the framework presents several fields to describe the new map group:

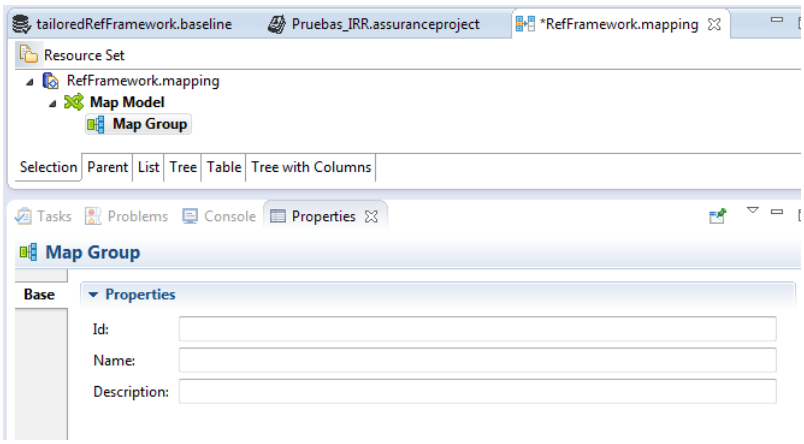


Figure 115 - Map Group properties

- Id: Map group identifier.
- Name: Map group name.
- Description: Map group description

6.9.2 Delete a map group

To delete a map group:

- Select the map group, press the right mouse button and select the contextual menu *Delete*.

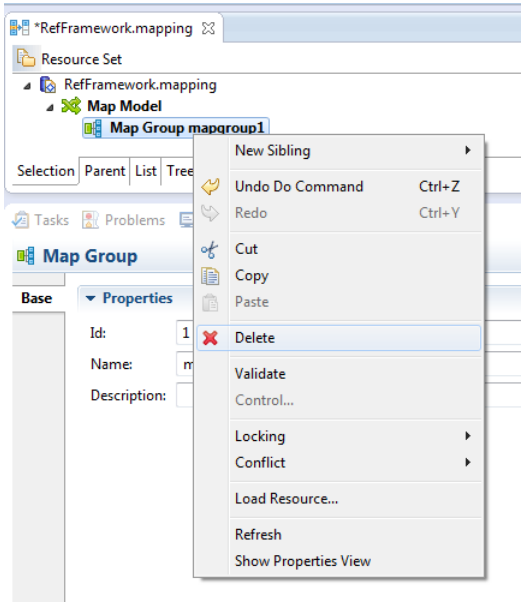


Figure 116 - Delete Map Group I.

- Or select the branch Model that contains the map group to delete, select the map group and press *the icon button*

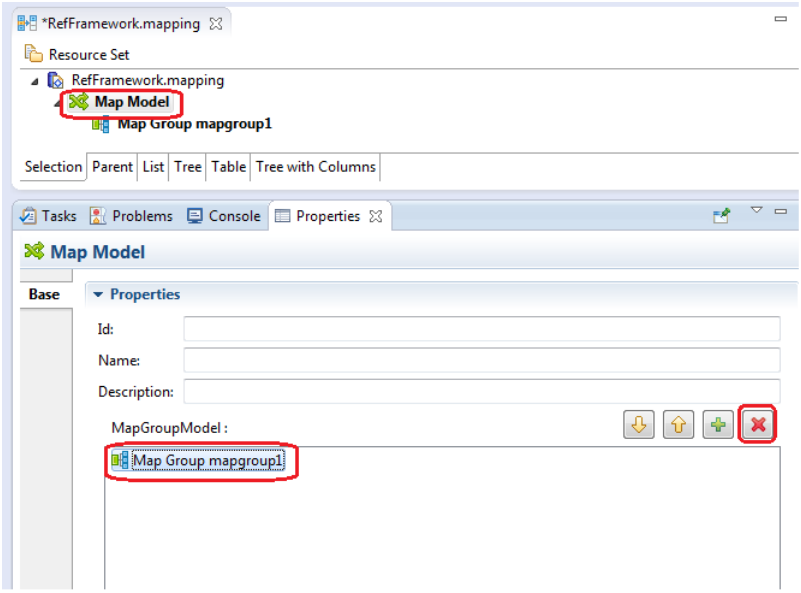


Figure 117 - Delete Map Group II.

6.10 Map edition

6.10.1 Add a map.

It is possible to add maps to a mapping model in two ways:

- Select the model element, press the right button of the mouse and select the contextual menu *New Child –> Map*

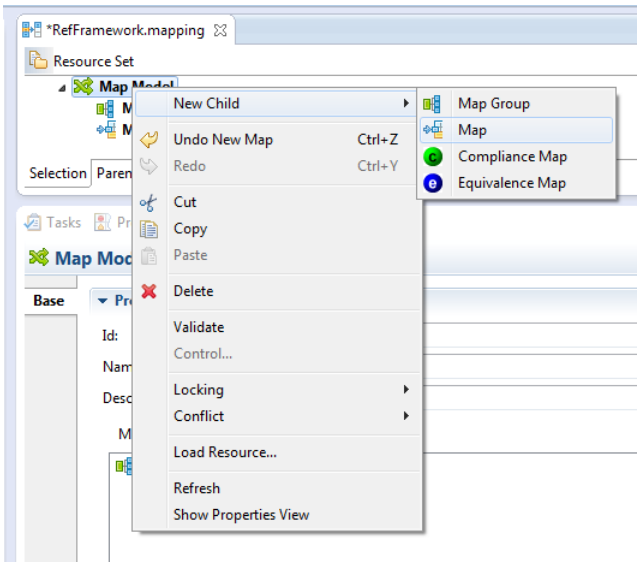


Figure 118 - Add New Map (I)

- Or, select the model element, and press the icon button *in the base tab associated to the label Map Model:*

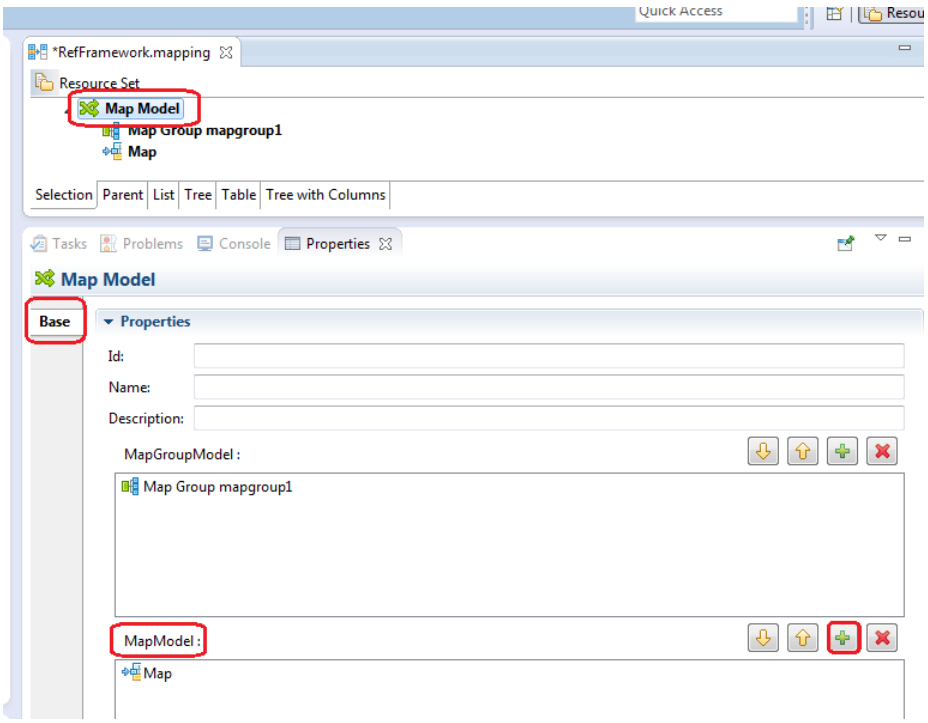


Figure 119 - Add New Map (II)

After these actions, in the properties zone, the framework presents several fields to describe the new map:

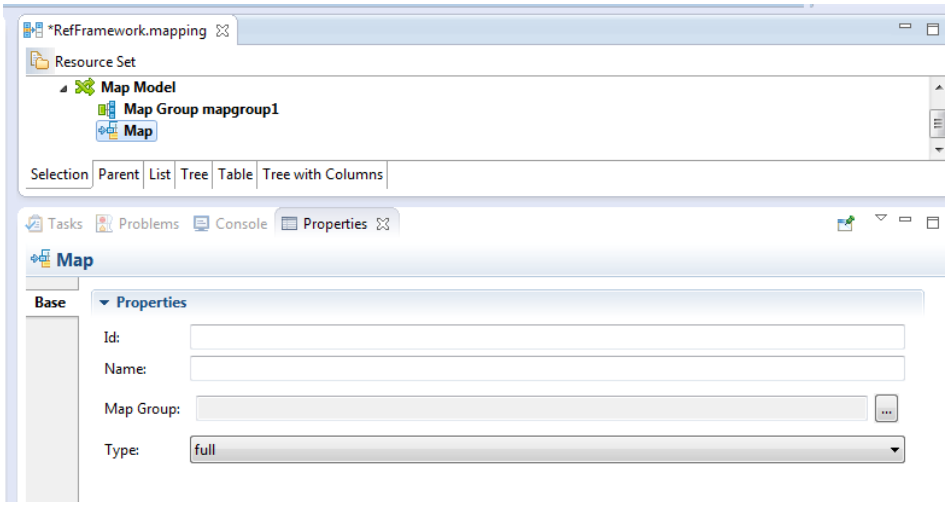


Figure 120 - Map properties

- Id: Map identifier.
- Name: Map name.
- Map Group: Map Groups associated to the map.
- Type: Map type: full, partial o not map.

6.10.2 Delete a map.

To delete a map:

- Select the map, press the right mouse button and select the contextual menu Delete:

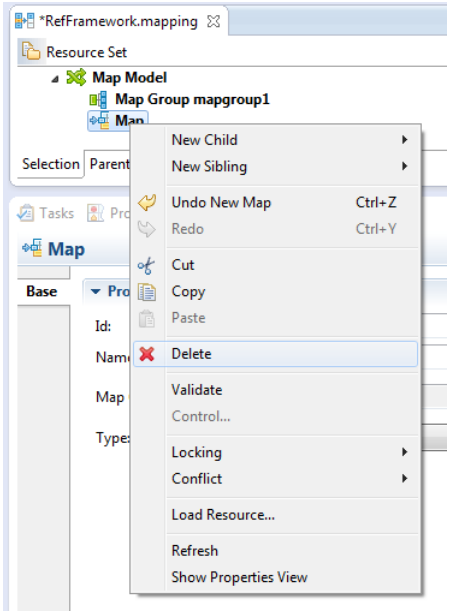


Figure 121 - Delete Map I.

- Or select the branch Model that contains the map to delete, select the map and press *the icon button* associated to the label MapModel:

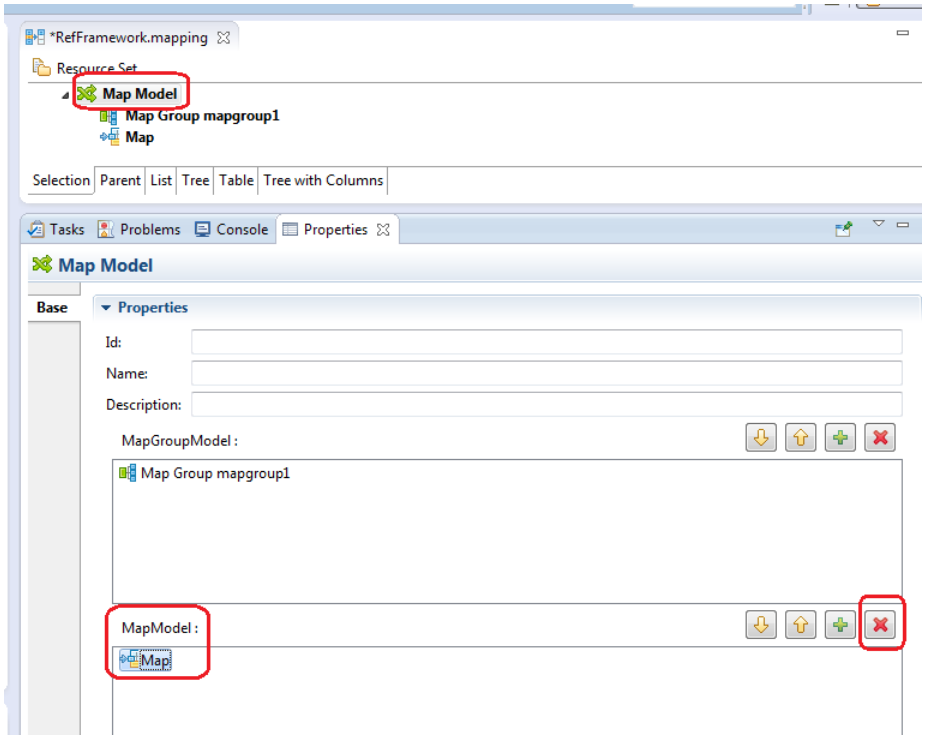


Figure 122 - Delete Map II.

6.11 Compliance via automatic generation of process-based arguments

According to the vision of AMASS for compliance management, the process development plan needs to fulfil the standards and should provide the justification of compliance. In particular, the automatic generation of process-based arguments arguing about compliance of processes. For instance, work products (e.g., design documents and verification report) show/argue/prove that the plans comply with the requirements. In this context, EPF Composer is used for modelling the definition and planning of processes and OpenCert is utilized for visualizing the generated arguments (i.e., argumentation model and diagram). The guidance for the creation of process elements are provided in Section 4, and also available in the EPF Composer manual [1]. The Process-based Argument Generator plugin takes the delivery process in the form of XML file as an input and transforms it into arguments (model and diagram).

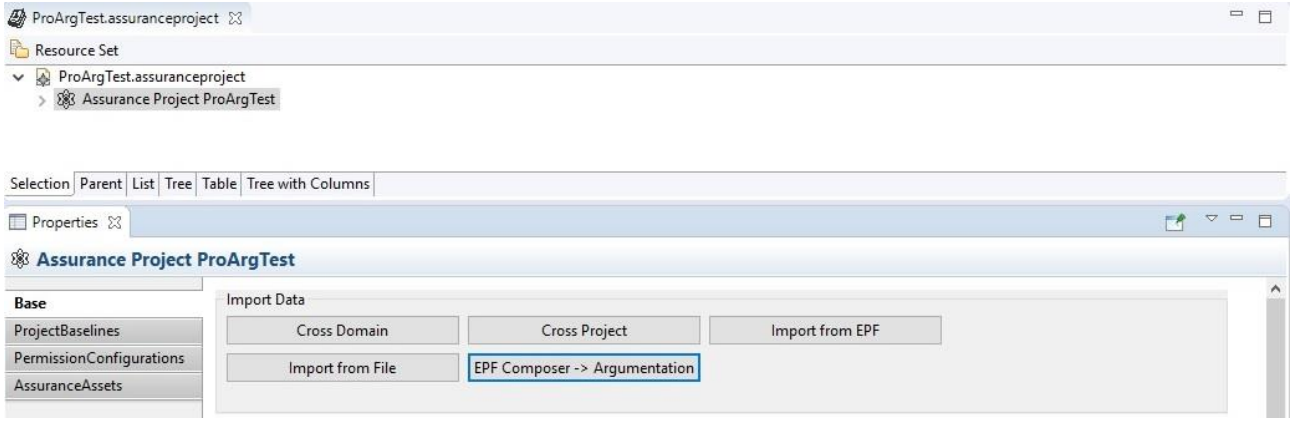


Figure 123 - Generation of process-based arguments button

To generate the process-based arguments, following steps need to be performed:

1. Export a delivery process modelled in EPF Composer using the XML export functionality. To do this, export one or more method plug-ins option would be selected under the Export XML. It might be noted that the plugin to be exported would not contain more than one delivery processes. For the next release, the export option for individual process(es) will be made available.

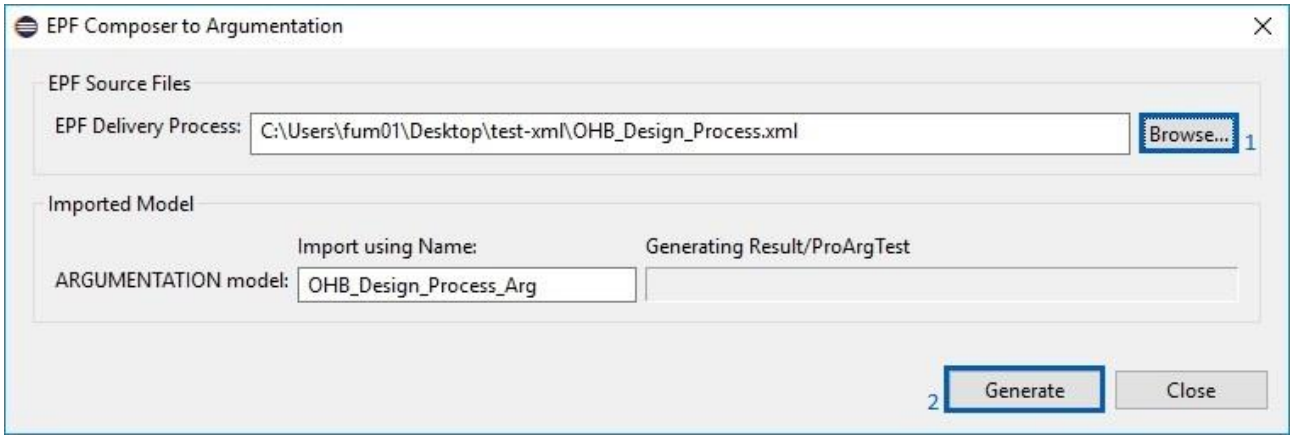


Figure 124 - EPF Process to Argumentation Model: XML file selection and generation buttons

2. Open the target assurance project model in OpenCert and press the button “EPF Composer -> Argumentation” on the properties of the Assurance Project element of the model, as shown in Figure 121. So, the generation command is activated from the OpenCert. However, for the next release, we will also support the generation command from any process in EPF Composer.

3. Next, the EPF XML file needs to be selected using the “Browse” functionality, as depicted in Figure 122. Note that the browsing dialog filters the files with the “.xml” extension only. At the end of the “EPF process to Argumentation Model” dialog, the “Generate” button starts the transformation process.
4. After completing transformation process, the generated argument model and diagram text will be shown in the text box below the label “Generating Result / [Assurance Project folder]”. If the transformation has errors, the Result box shows them. Otherwise, the EPF process has been successfully transformed into arguments. As depicted in Figure 123, the user will need to press the “OK” button, then the “Close” button to finish the generation process.

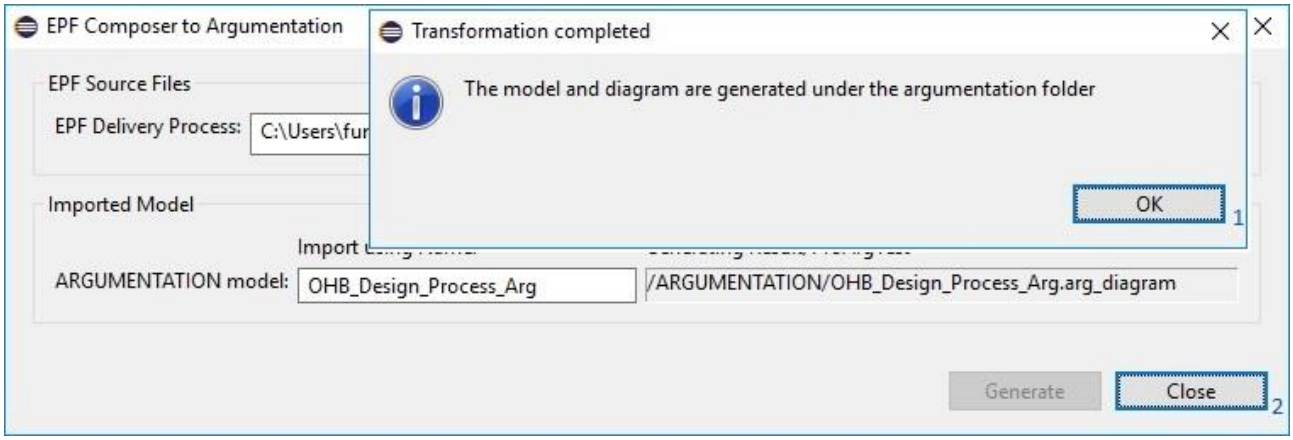


Figure 123 - Transformation successfully completed

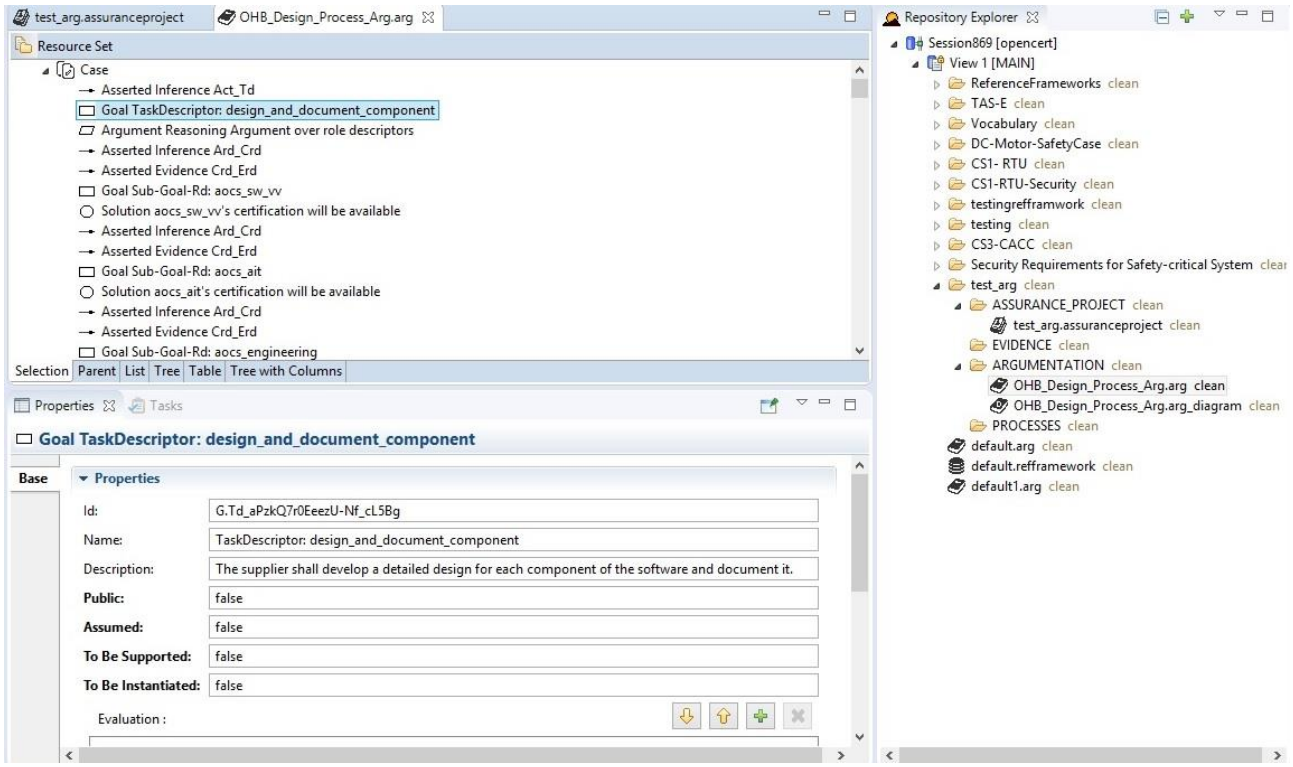


Figure 124 - Generated Argumentation Model

5. Argumentation model and diagram are generated under the ARGUMNETATION folder, as shown in Figure 124. The generated model and diagram will be shown in the editor window when you double click on them. For displaying elements in argumentation diagram editor, user has to select the elements from the Navigation View and drag them to the editor, see Figure 125. In the argumentation diagram, Claim (Goal in GSN) shows Id and Description of the Activity and/or TaskDescriptor. If user wants to see name of

the Goal corresponding to TaskDescriptor he/she has to click on the element; the Id, Name and Description will be shown in the properties tab of the Base.

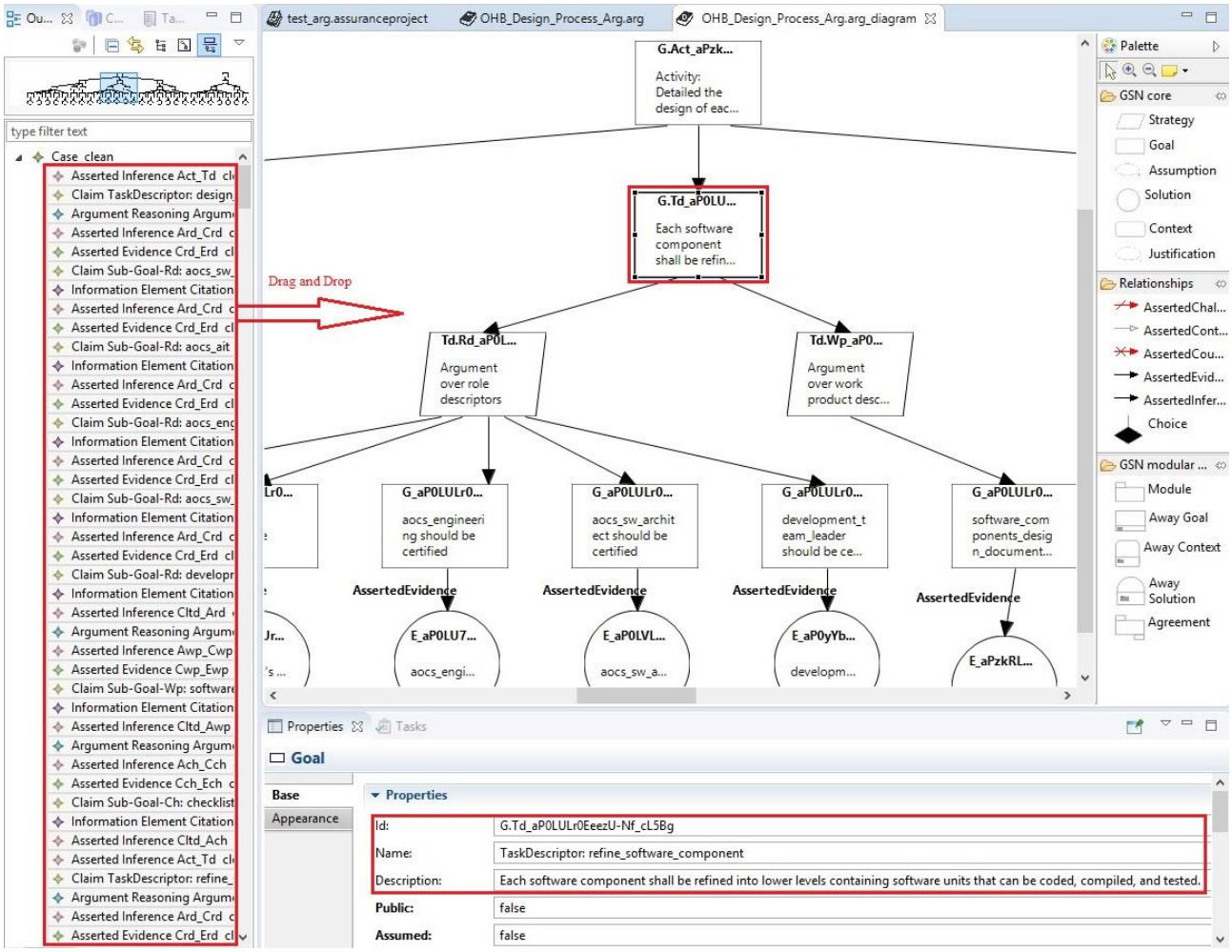


Figure 125 - Argumentation Diagram

In this prototype, we have mapped the basic process elements to arguments. However, in the next iteration we will provide detailed mapping of elements. The ProcessComponent that contains the information of the process has transformed into a Case, whereas Activities and TaskDescriptors have mapped into Claims in argumentation model and Goals in argumentation diagram. The crucial process elements associated to a TaskDescriptor, namely a set of RoleDescriptors, WorkProductDescriptors, Guidelines, ToolMentors and Tools, Checklist, and Example have transformed to ArgumentReasoning elements (aka GSN strategies) in order to divide a goal into sub-goals. Evidences associated to these elements are mapped into InformationElementCitation typed solutions showing that the particular goals have been achieved.

6.12 Import models into Assurance Projects

The user can import models from different assurance assets stored locally into an existing assurance project. Open the .assuranceproject model in the models edition area. In the properties view the user will find a button with the text “Import from file”.

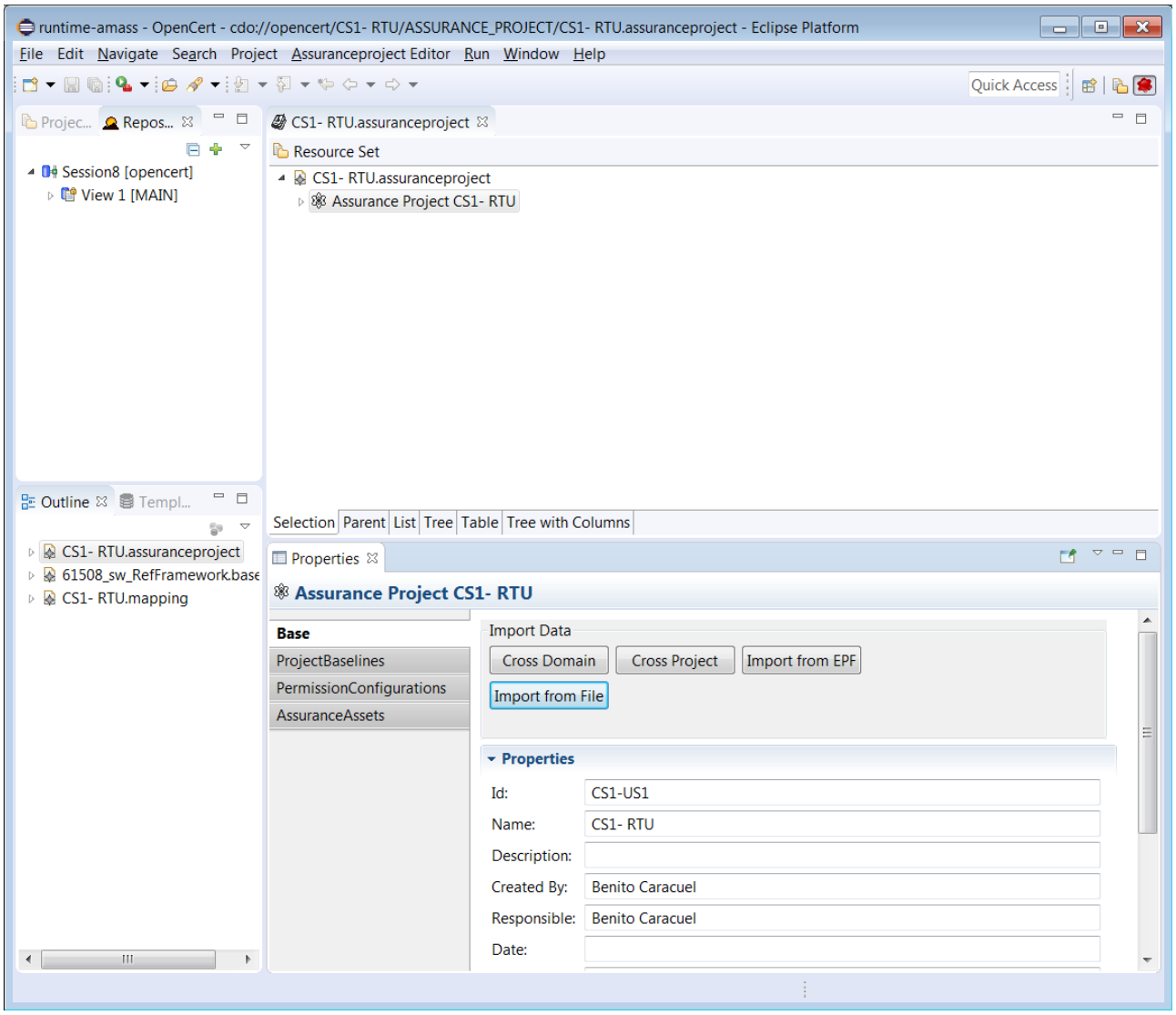


Figure 125 - Assurance project editor with the import from file button

When the user press the “Import from File” button, a new window will appear. Here the user should select the type of model to be imported (Argumentation model, Evidence model or Process model)and the source, where the file is located in the computer. By default the imported model will have the same name as the file unless the user chooses to change it in the Destination model text.

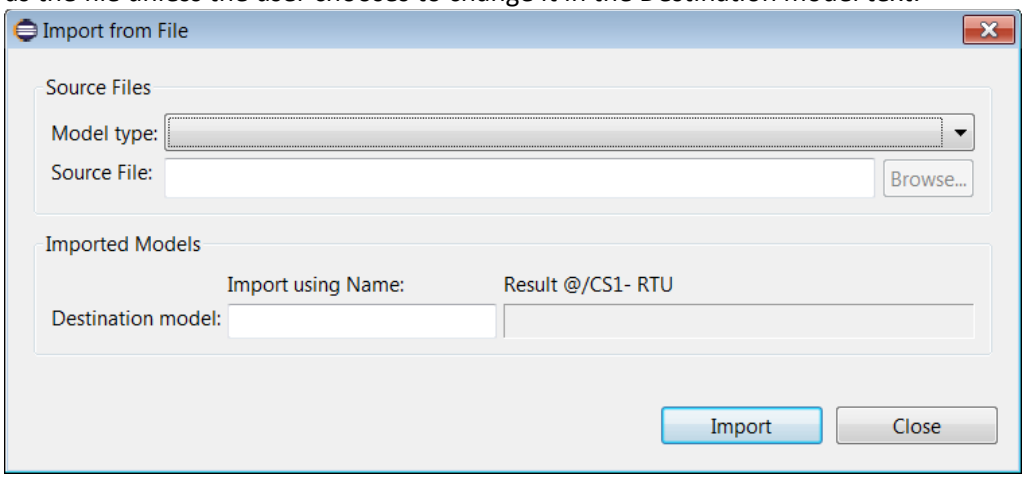


Figure 126 - Import from File window

The models will be imported in the actual assurance project in the associated assurance project in the respective folders, the argumentation model in the ARGUMENTATION folder, the evidence model in the EVIDENCE folder and the process model in the PROCESS folder.
NOTE: the diagram part associated with the model will not be imported, just the model. It can be visualized in the respective tree view editors.

7 System Component Specification

This section documents the usage of the Papyrus and CHESS support to allow the modelling of the system architecture, with focus on contract-based design, and the modelling of the links between the architecture and assurance related information, allowing to put the basis for the AMASS approach for architecture-driven assurance.

7.1 Create a CHESS project, model and diagrams

CHESS project (i.e. the folder where the model and other artefact are stored) and model can be created by using the CHESS dedicate Eclipse wizard (File->New->Others->CHESS…).
Please note that currently CHESS projects/models cannot be created in a CDO checkout but only as folder/file in the current workspace.

CHESS project is basically a Papyrus project customized with the CHESS *nature*.

CHESS model is a Papyrus UML+SysML+MARTE model coming with a predefined set of packages/views and with the CHESS profile automatically applied on it, to allow modelling of contracts, dependability and real-time concerns.

CHESS profile and tool can also be used to exploit different analysis at system level, and to apply the CHESS model driven methodology for the design, analysis and implementation of critical SW systems. The application of the CHESS methodology for SW development is not mandatory in the context of the AMASS solution; more information about the CHESS methodology can be found at https://www.polarsys.org/chess/ .

The CHESS editor is actually a customization of the Papyrus editor, offering a set of features on top of the CHESS model. To have an overall understanding of the Papyrus environment, so for instance regarding the different views, commands, diagrams, please check the Papyrus user manual available online: https://wiki.eclipse.org/Papyrus_User_Guide.

Below are some examples of SysML Block Definition Diagram (BDD) and Internal Block Diagram (IBD) which can be used to model the system hierarchical architecture, i.e. the blocks, ports and the connections.

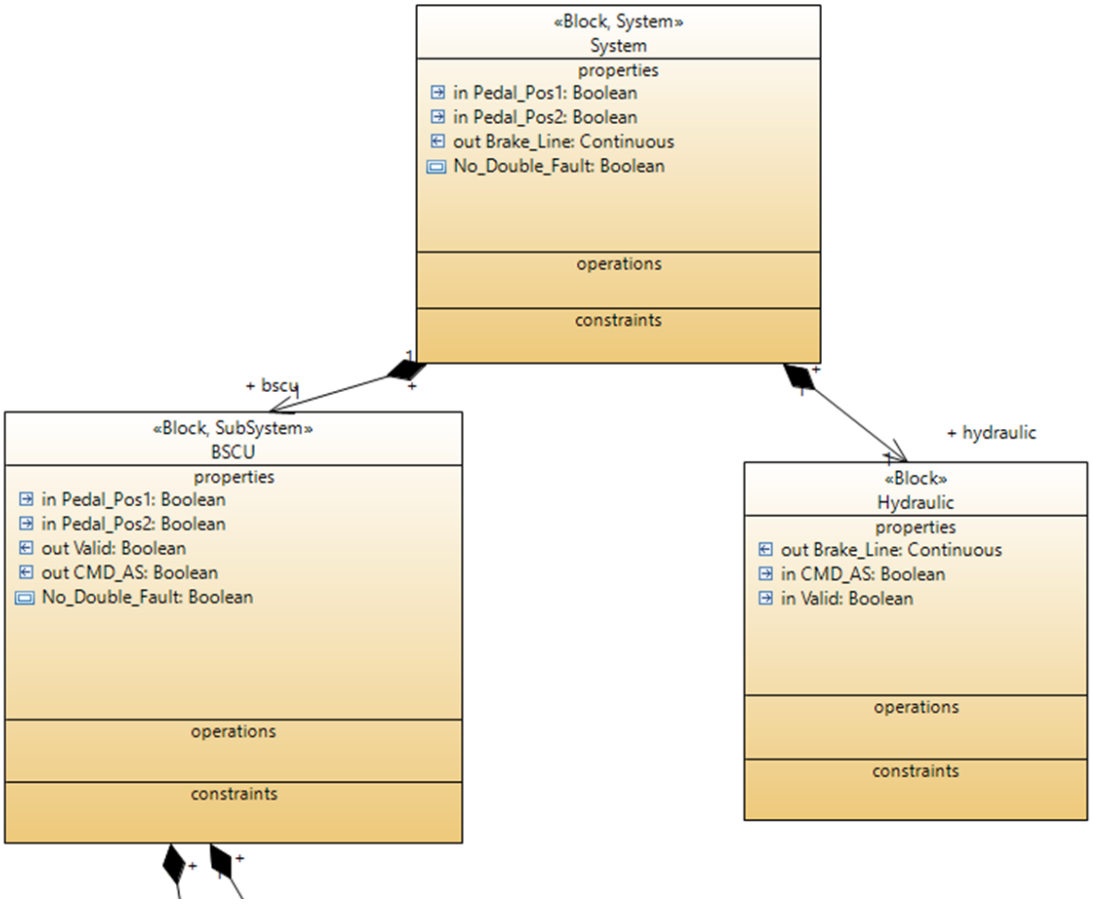


Figure 127 - Block Definition Diagram example

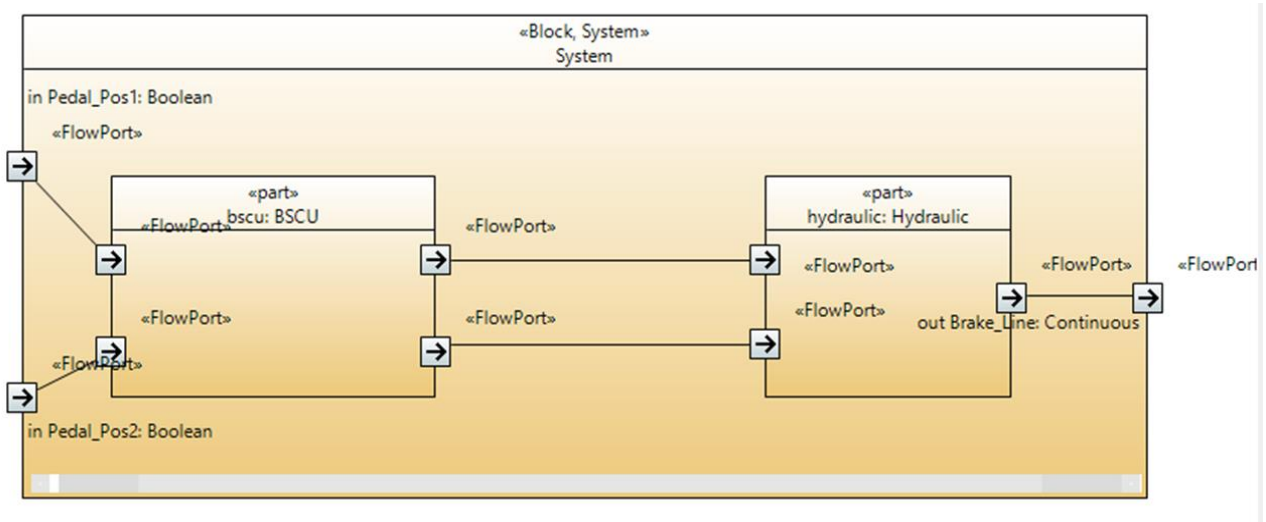


Figure 128 - Internal Block Diagram example

7.2 Creating Requirements

System requirements can be created by using the SysML Requirement diagram. On the model explorer, right click on CHESS RequirementView package and create a SysML Requirement diagram. Use the palette to create Requirement entities.

Papyrus also supports the import of requirements from different sources, e.g Excel files or ReqIF model; more about this support can be found at [11].

7.3 Create FormalProperties

FormalProperties (i.e. the entities which play the role of Assumption and Guarantee in a Contract) can be creates manually by using the dedicated tool in the *Contracts* palette.
It is also possible to package a FormalProperty in a Contract by creating the former directly in the latter.
To create a FormalProperty in a Block Definition Diagram (BDD)/Component diagram:

- Select FormalProperty from the Contract palette and click on the BDD
  - Give a proper name to the FormalProperty

To create a FormalProperty associated to a specific component (i.e. the owner of the FormalProperty), in a Block Definition Diagram (BDD)/Component diagram:

- Select FormalProperty from the Contract palette and click on the specific component in the BDD
  - Give a proper name to the FormalProperty

To create a FormalProperty packaged in a Contract:

- Select FormalProperty from the Contract palette and click on the Contract
  - Give a proper name to the FormalProperty

7.4 Edit Formal Property

To edit a FormalProperty:

- Select the FormalProperty from the Project Explorer View, or select its graphical element from the BDD.
- Open the tab “PropertyEditor+” in the Property View and write the formal property in OCRA Language (Appendix B. OCRA Language to define Formal Properties and Contracts). Pressing CRTL + SPACE it is possible to have a list of supported keywords of the OCRA Language to define the formal property. If the owner component is set correctly, the content assist can suggest also the input/output ports and the attributes of the owner component to type. Moreover, it notifies whether a typed word does not belong to a keyword/port/attribute, see Figure 129.

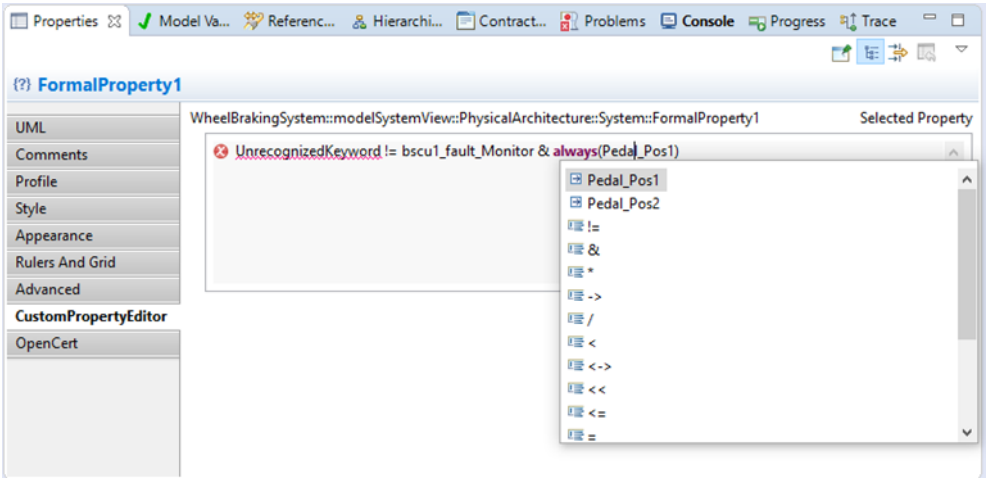


Figure 129 - Property Editor with content assist

To associate an expression to the FormalProperty:

- Select the FormalProperty

- In the UML tab of the Properties view edit the current value of the Specification field and provide the expression in the Value field

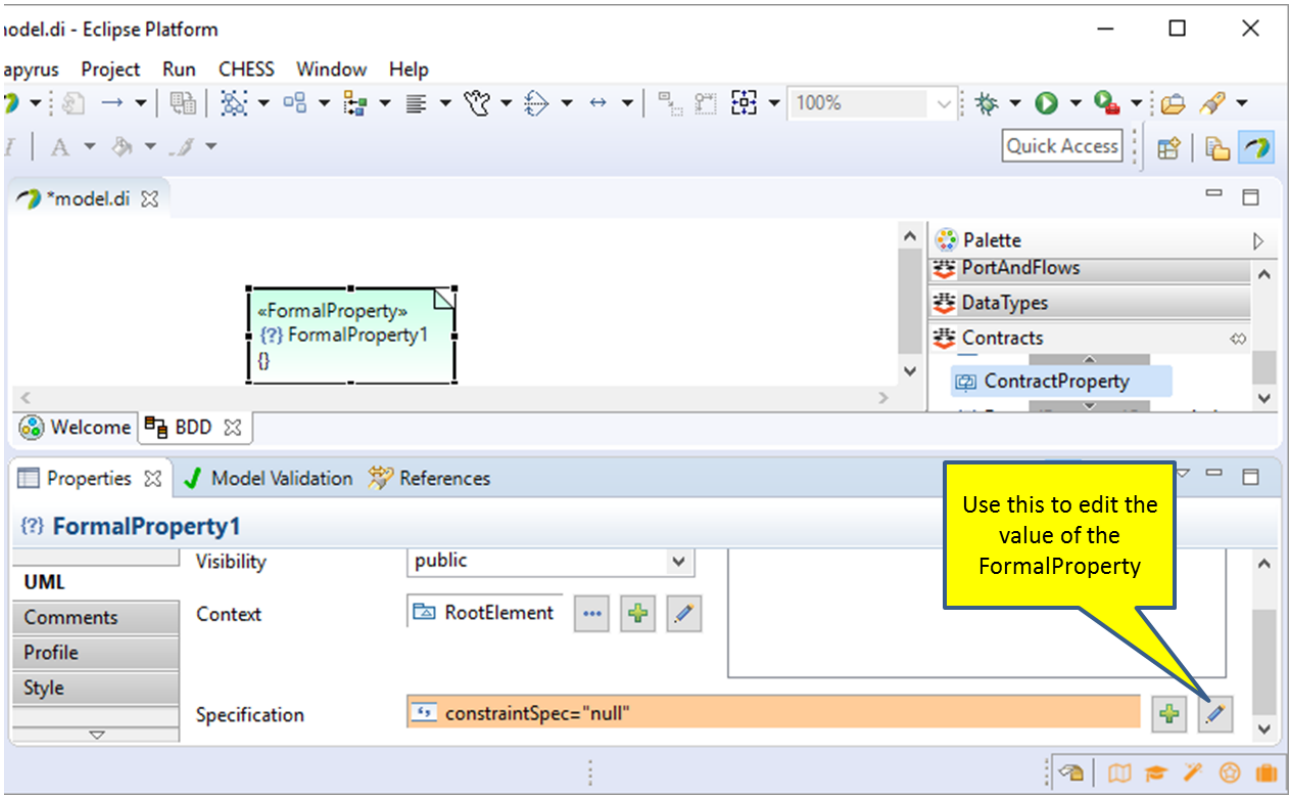


Figure 130 - Creating a Formal Property

Use the formalize property of the FormalProperty stereotype, available in the Profile tab of the Property View, to link the requirements formalized by the current FormalProperty (see *Figure 131*).

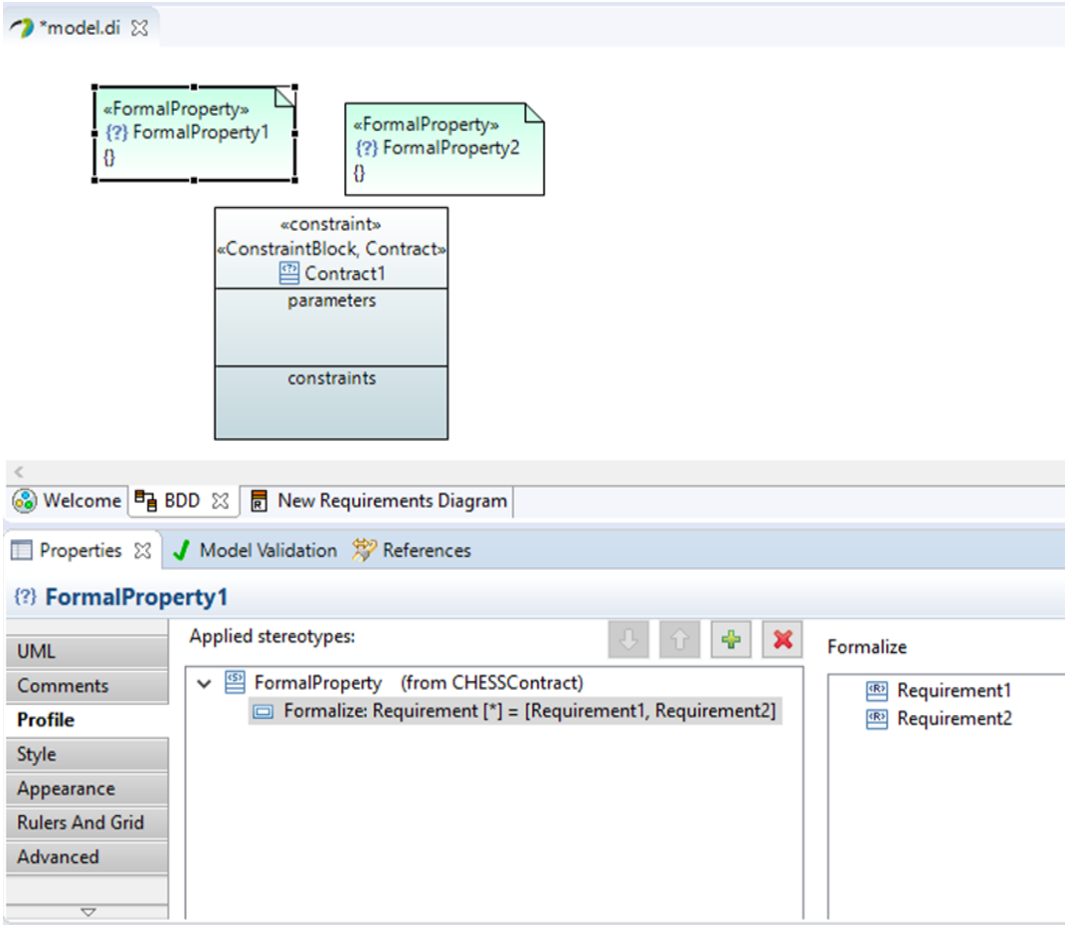


Figure 131 - Formalizing Requirements

7.5 Creating a Contract

Contracts can be created in a BDD and in a Class/Component Diagram.
To create a Contract:

- Open a BDD/Class diagram
- Select Contract from the Contracts palette and click on the diagram
  - A popup appears to choose if a new contract must be created or if an existing one must be instantiated (see Figure 132).
  - Give a proper name to the Contract

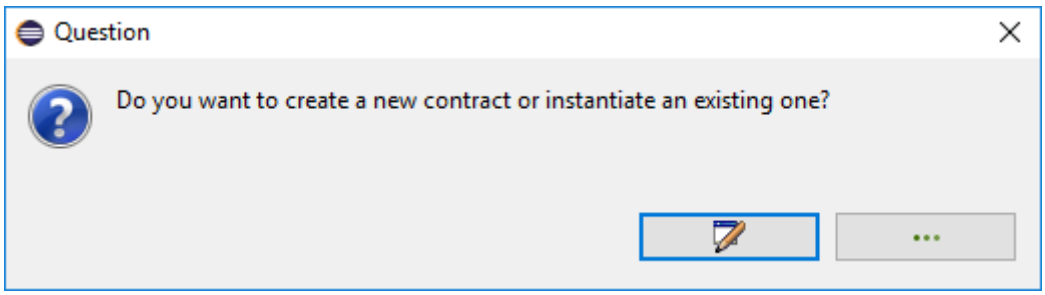


Figure 132 - Popup to create a new contract or instantiate an existing one.

Alternatively, it is possible to create a Contract without the need to create its graphical representation from a BDD:

- Select the component to assign the contract. The element can be selected in the Model Explorer View or in the graphical editor.
- Open the tab “ContractEditor+” in the Property View, type the name of the contract in the “new contract's name” area and click the “Add Contract” button (see Figure 133). A popup appears to choose if a new contract must be created or if an existing one must be instantiated.

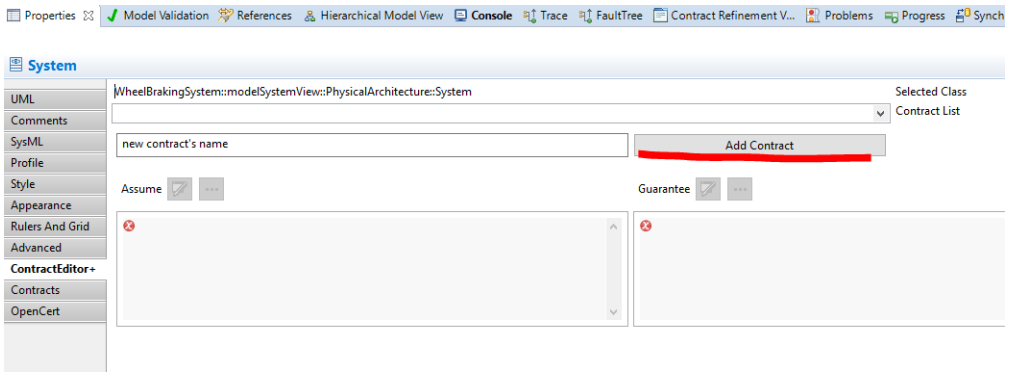


Figure 133 - The Add Contract button in the ContractEditor+

7.6 Specify Assumption and Guarantee for a Contract

To specify an Assume (or Guarantee) FormalProperty for a given Contract:

- Select the Contract, open the Profile tab in the Property view and set the Assume (or Guarantee) attribute of the Contract stereotype to the (previously created) FormalProperty.

The Assumption and the Guarantee properties for a given Contract can also be specified through the dedicated ContractEditor+ tab in the property editor (see Figure 134). To be able to use the ContractEditor+ tab:

- Select the Contract to edit, or
- Select the Block/Component and then select the ContractProperty (see below) from the ContractList in the ContractEditor+ tab.

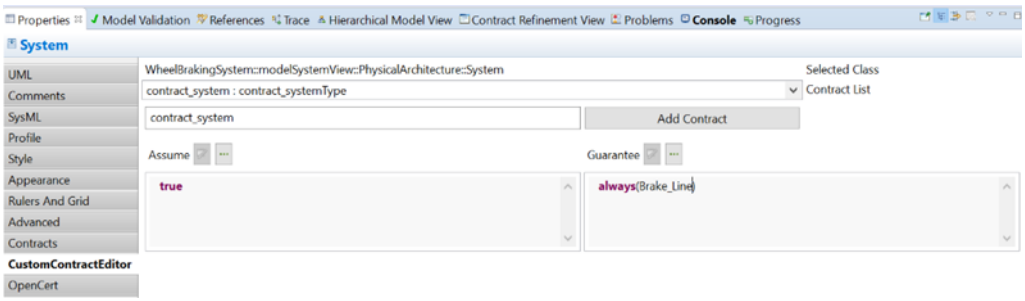


Figure 134 - Editing the Contract’s Assume and Guarantee

7.7 Associate a Contract to a Block/Component

To associate a Contract to a Block/Component the following actions need to be performed:

- Create a ContractProperty inside the Block/Component (see Figure 135). The ContractProperty acts as a special attribute of the Block/Property.
- Type the just created ContractProperty with the Contract by using the UML tab in the Properties view.

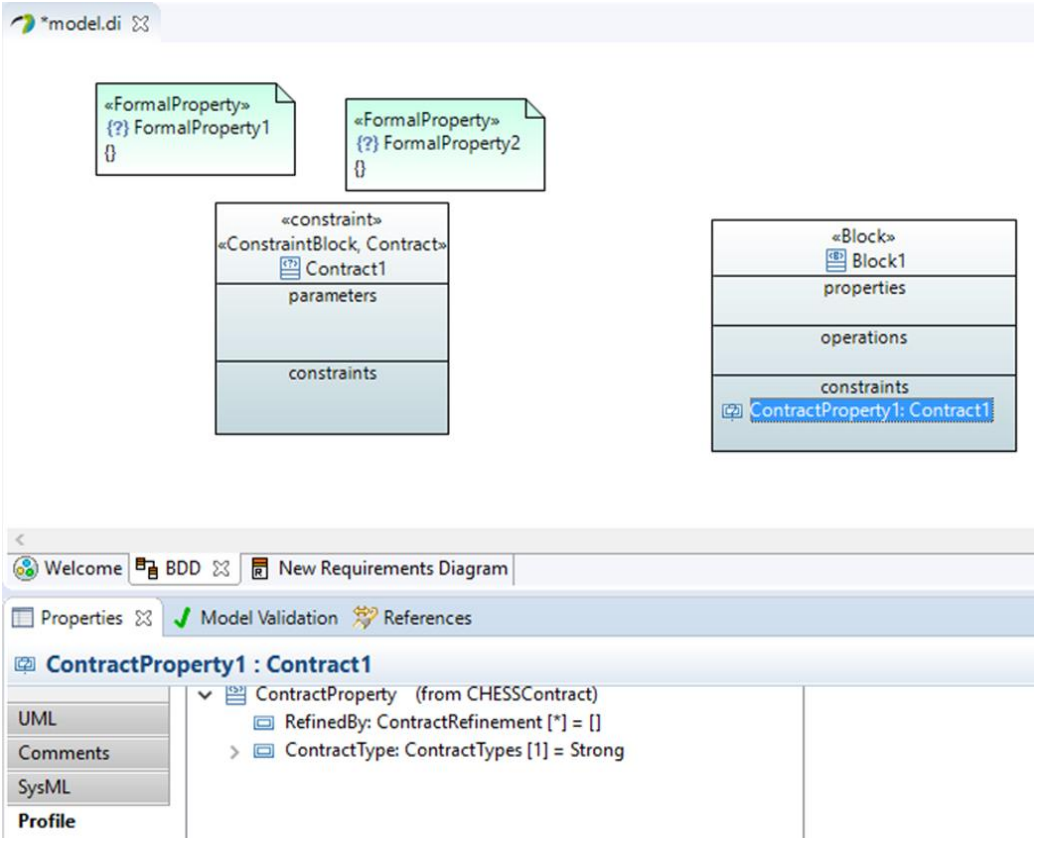


Figure 135 - ContractProperty

The ContractProperty stereotype comes with the attribute ***ContractType*** which can be used to set the ***Strong*** or ***Weak*** property of the Contract associated to the architectural entity (Sljivo, Gallina, Carlson, & Hansson, November 4-7, 2013)[5]

7.7.1 Selection of weak contract for Block/Component instances

While a strong contract associated to a Block/Component (must) hold for all the instances of the Block/Component, a weak contract associated to a Block/Component can hold for a given instance of the Block/Component if the environment where the instance is placed met the assumption. So for a given instance it has to be possible to specify the set of weak contracts specified for the typing Block/Component (if any) which hold for the instance. To do so, the following steps have to be performed:

- Select the Block/Component instance in the SysML Internal Block Diagram / UML Composite Structure Diagram.
- Apply the ComponentInstance stereotype to the instance.
- Select the Contract tab in the properties editor (see Figure 136): the Contract tab shows the strong and weak contract inherited by the classifier typing the instance. In particular the Weak Contract area can be used to check the weak contracts that hold for the current instance.

[5] While the strong assumptions and guarantees must be satisfied always in order for component to be used, the weak pairs offer additional information in some specific contexts where besides the strong assumptions, the weak assumptions are to be met as well.

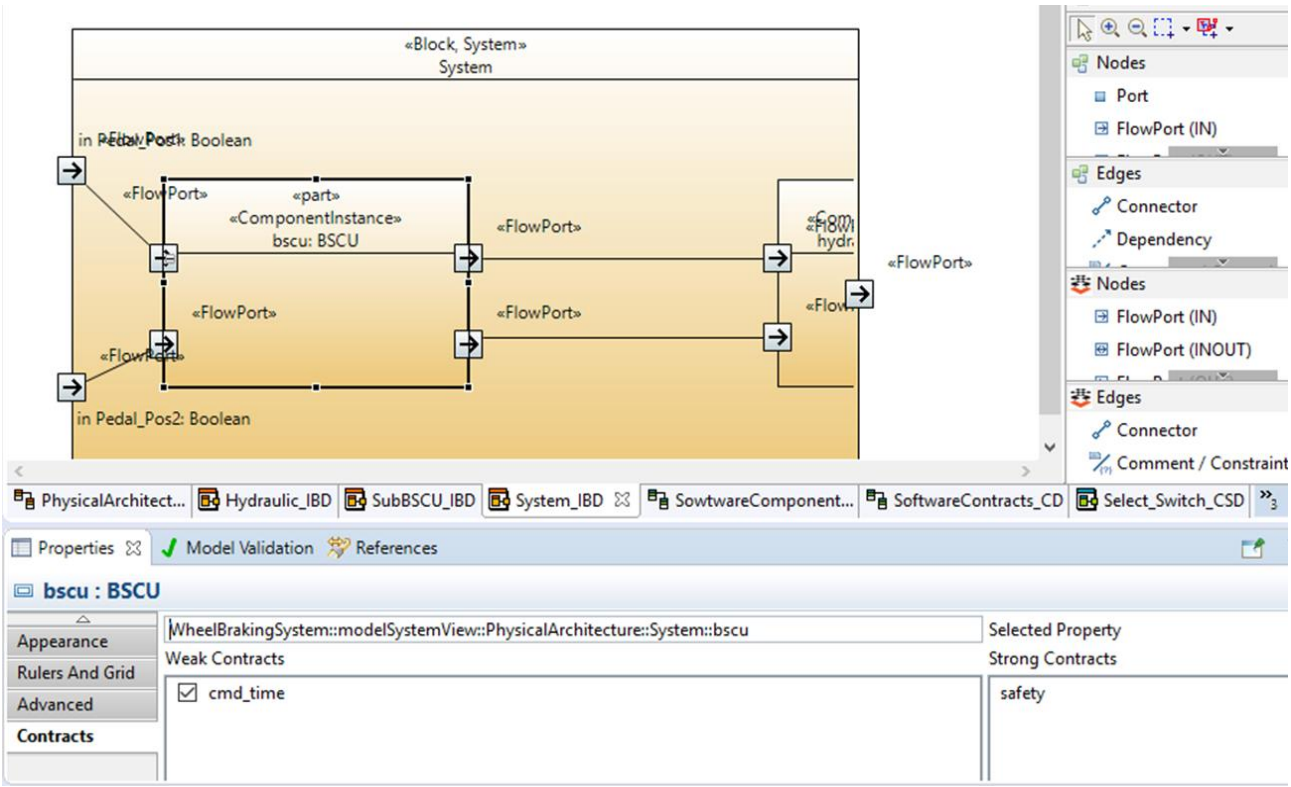


Figure 136 - Contract tab for instances

The information about the weak contracts which hold for the given instance is automatically set in the *WeakGuarantees* attribute of the *ComponentInstance* stereotype.

7.7.2 Contract Refinement

Once a model component that has a contract has been decomposed, it is possible to define the contract's refinement. The refinement of a contract can be specified following these steps:

- Select a ContractProperty of a Block
- Right-Click and select Formal Verification -> Set Contract Refinement

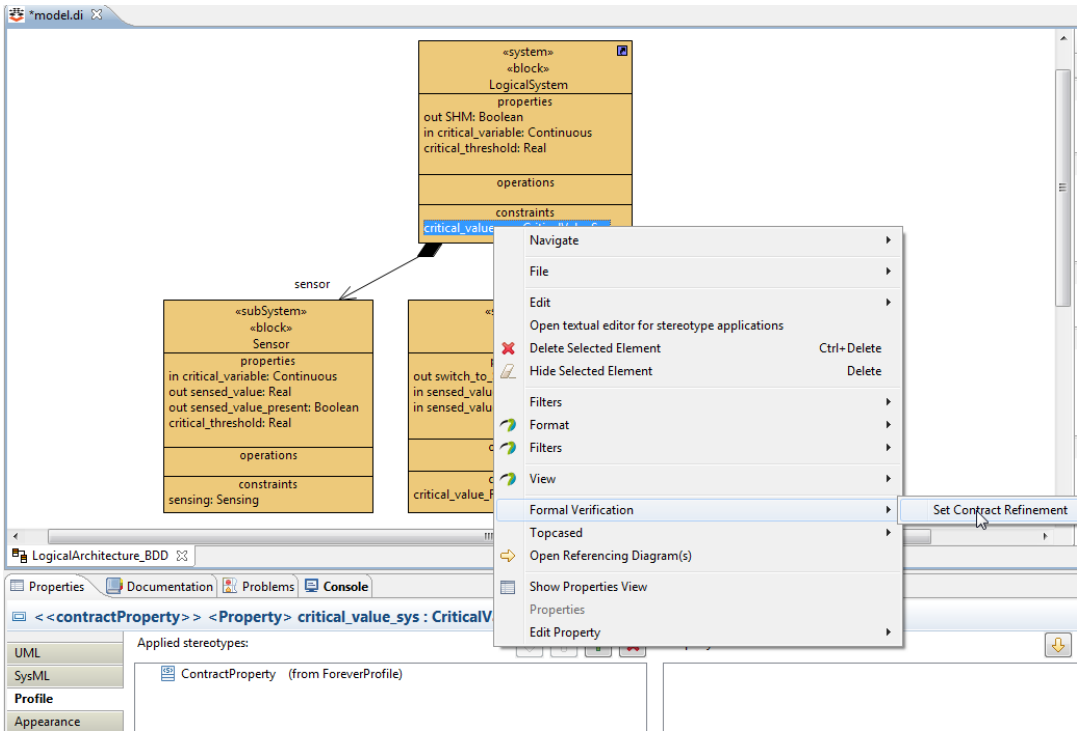


Figure 137 - Set Contract Refinement Command

- In the new dialog window select the Contract Properties from the list (CTRL-A selects all) and click OK. This dialog window shows the Contract Properties in the format: *InstanceName.ContractProperty*. This allows the selection of instance-based Contract Properties (instead of type-based).

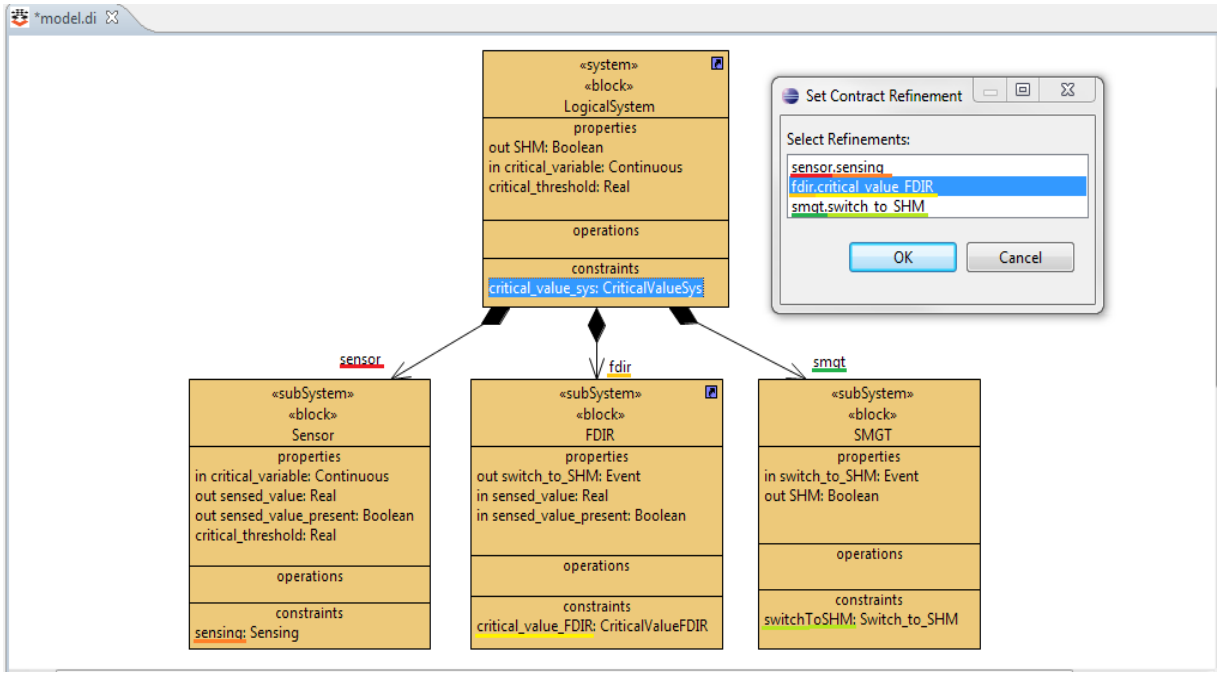


Figure 138 - Refinement Selection

Be sure that the *Aggregation* kind of the instances is set to *composite* as shown in the bottom-right part in the figure below. This is required to let Set Contract Refinement command to work properly.

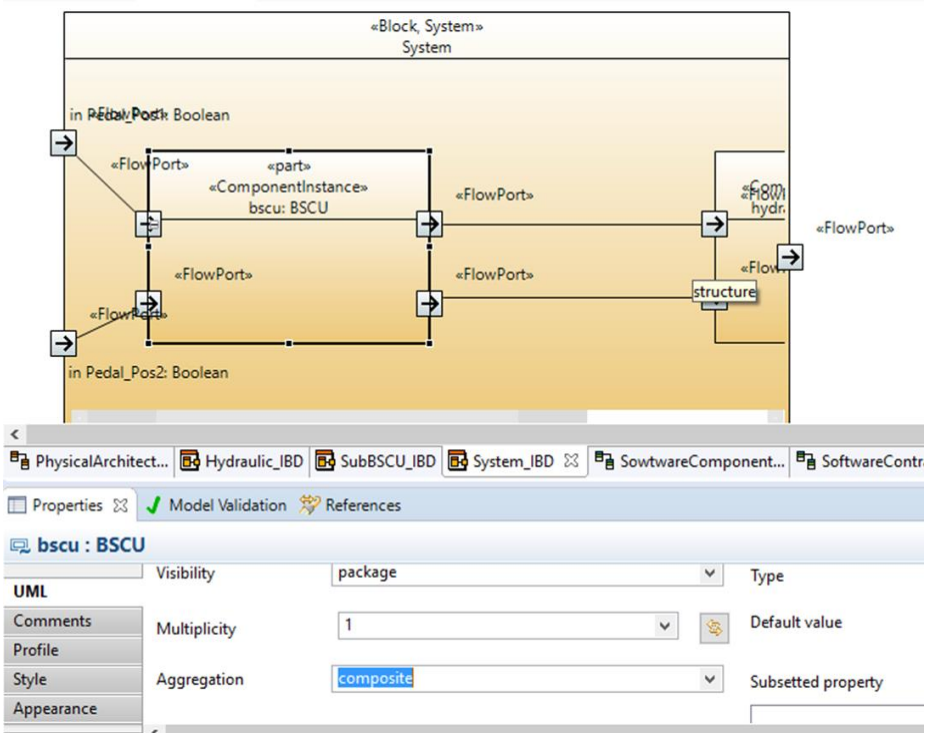


Figure 139 - Composite Aggregation

- The information about the refinement is set in the RefinedBy attribute of the ContractProperty stereotype of a Block and available in the Profile tab of the Property view:

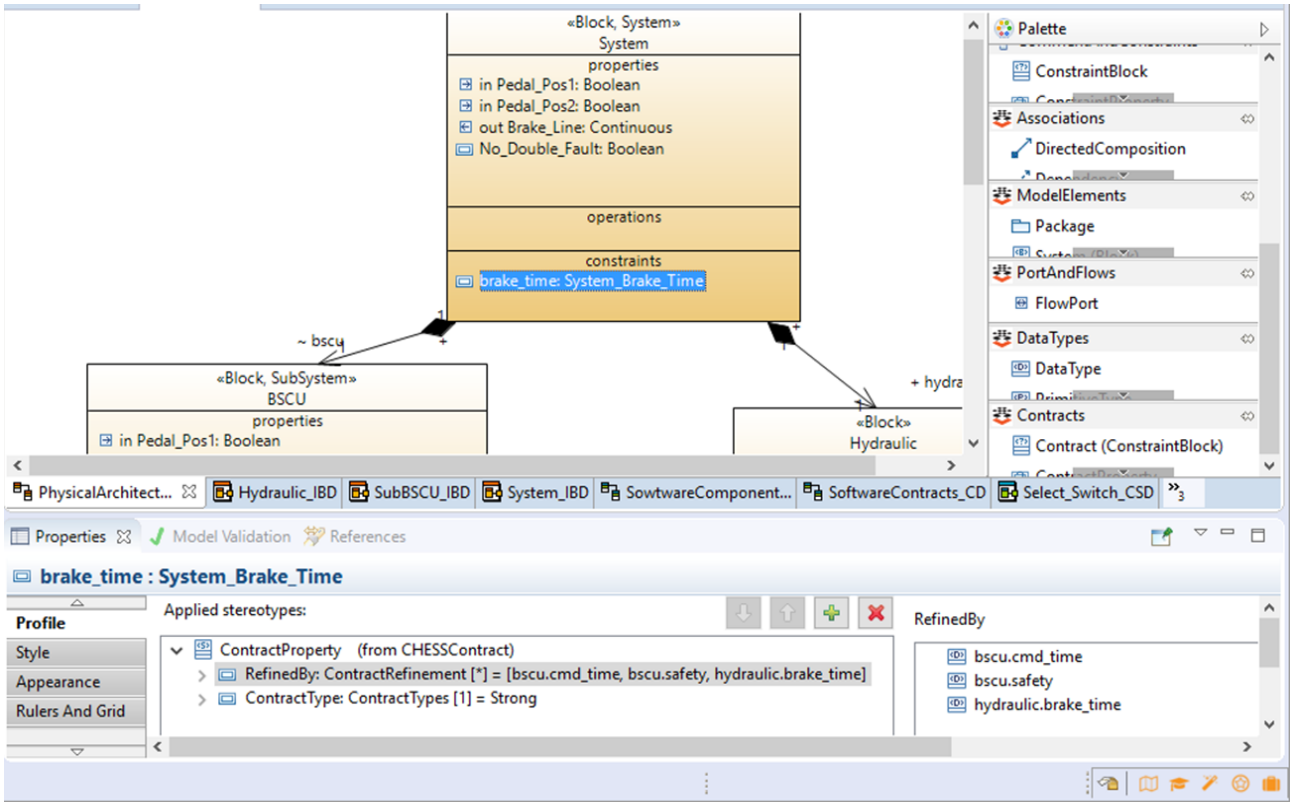


Figure 140 - Contract Refinement

7.8 Specify Component Behaviour (nominal and faulty)

Nominal behaviour of component can be provided by using UML State Machine diagrams. Please contact Intecs if you want to know about the kind of information that can be provided for the nominal behaviour to enable integration with xSAP verification tool for automatic fault tree generation.

Faulty behaviour for a component can be provided by using the CHESS dependability profile.
Several possibilities exist in CHESS to model the faulty behaviour of a component.
For instance, faulty behaviour can be modelled in a dedicated state machine stereotyped with ErrorModel; the ErrorModel represents a particular kind of state machine containing information about faults, errors and failure modes and their propagation internal to a given component.
Once defined, the ErrorModel state machine can be attached to a given component by stereotyping the component with ErrorModelBehaviour.

The ErrorModelBehavior stereotype comes with the errorModel attribute, which has to be used to reference the actual ErrorModel state machine to be attached to the component (Figure 141).

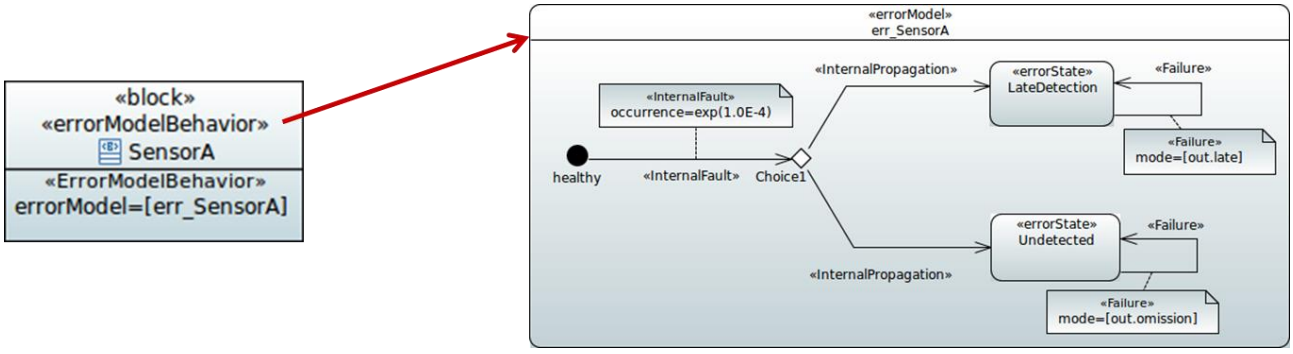


Figure 141 - ErrorModel behavior

The following information can be provided through an ErrorModel state machine:

- Initial state
  - It represents the “healthy” state of the component
- Errors
  - UML State, with the «ErrorState» stereotype
- Internal faults
  - UML Transitions, with the «InternalFault» stereotype
    - connecting the initial state and an error state
    - occurrence – time to fault occurrence (time distribution)
- Internal propagations
  - UML Transitions, with the «InternalPropagation» stereotype
    - delay – time after which propagation occur
    - weight - relative probability of occurrence
    - externalFaults – Boolean expression on the occurrence of external faults (i.e., failures incoming on input ports of the component)
- Failures

- UML Transition, with the «Failure» stereotype
  - mode – the failure mode(s) under which the failure manifest itself on the port(s) of the component

The possible propagation paths between components may be enriched with dependability information.

Propagation may occur when functional relations exists between two components (so the components are connected through their ports via UML Connectors)

By default it assumed that immediate and deterministic propagation will occur through these paths. Diifferent behaviour may be specified using the Propagation stereotype, the latter extending the UML Connector with the following attributes (*Figure 142*):

- prob – probability that propagation occurs
- propDelay – delay after which propagation occurs (time distribution)

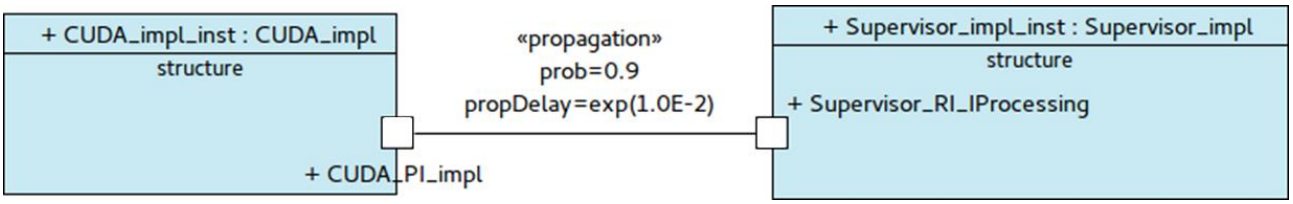


Figure 142 - Propagation path between two components

7.9 Managing Analysis Contexts

Analysis contexts are used in CHESS to collect information about a given analysis to be run; an analysis context is used at least to refer to the set of entities that have to be considered for the analysis. If needed, the analysis context can be used to provide analysis specific configuration parameters.
Analysis context can be created in the CHESS Dependability View by using class diagrams.

7.10 View Status of System Architecture

CHESS provides a hierarchical view that shows the decomposition of the system component into sub-components. It shows also the contracts assigned for each component. The system is graphically represented as the top element of the view, see Figure 143.
To show the view, go to Window – Show View – Hierarchical Model View

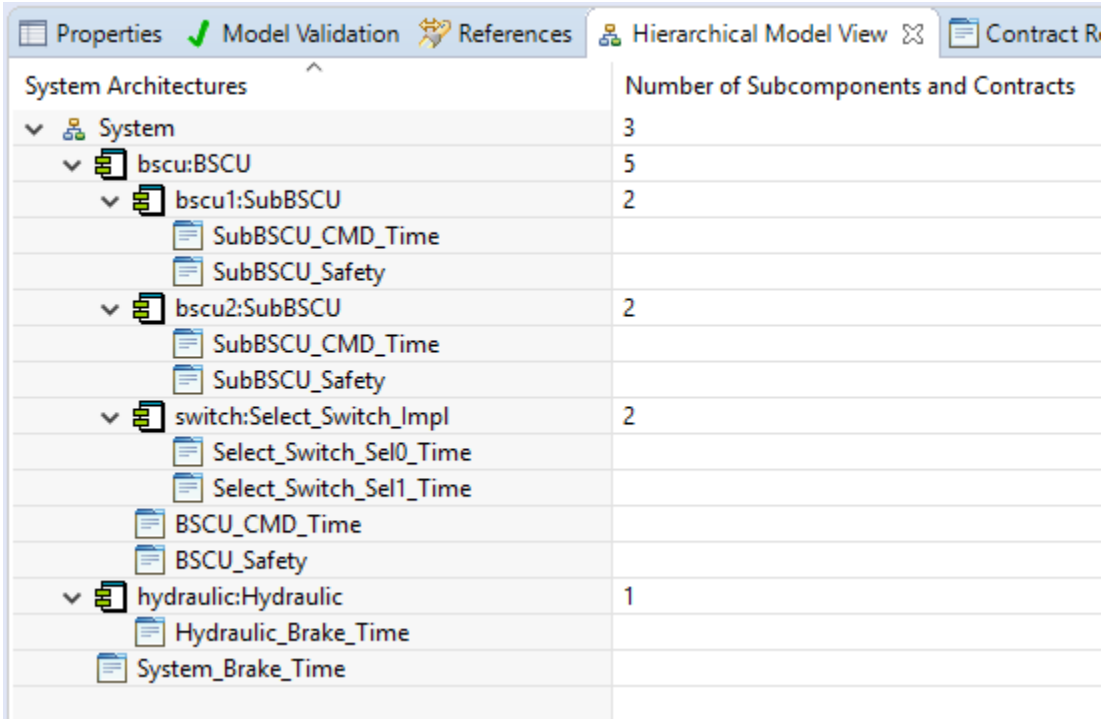


Figure 143 - Hierarchical view of the system decomposed into sub-components and contracts

CHESS also provides a hierarchical view that shows the contracts with their refining contracts. The weak contracts are graphically represented as a document with a "W" on top, see Figure 144.
To show the view, go to Window – Show View – Contract Refinement View

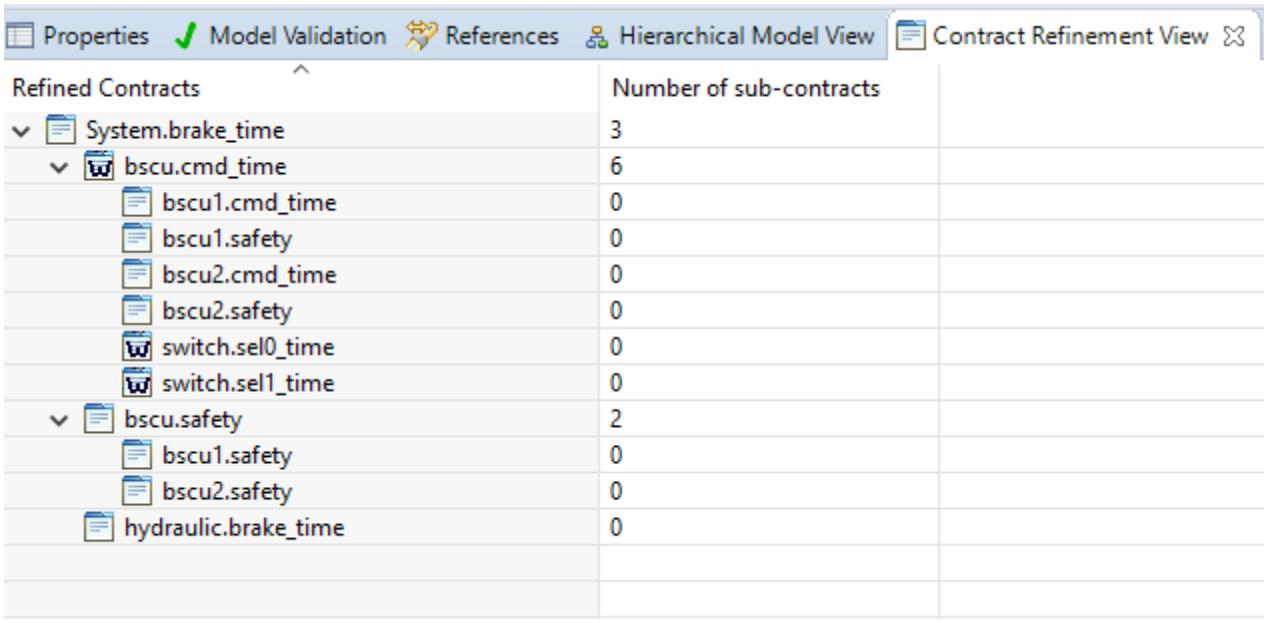


Figure 144 - Contract Refinement View

A report about contracts status with respect to the assurance can be also generated through CHESS; from the MoelExplorer view coming with Papyrus/CHESS, right click and select

Validation → CHESS → Validate Contracts for Assurance

then issues with the defined contracts are reported in the Model Validation view.

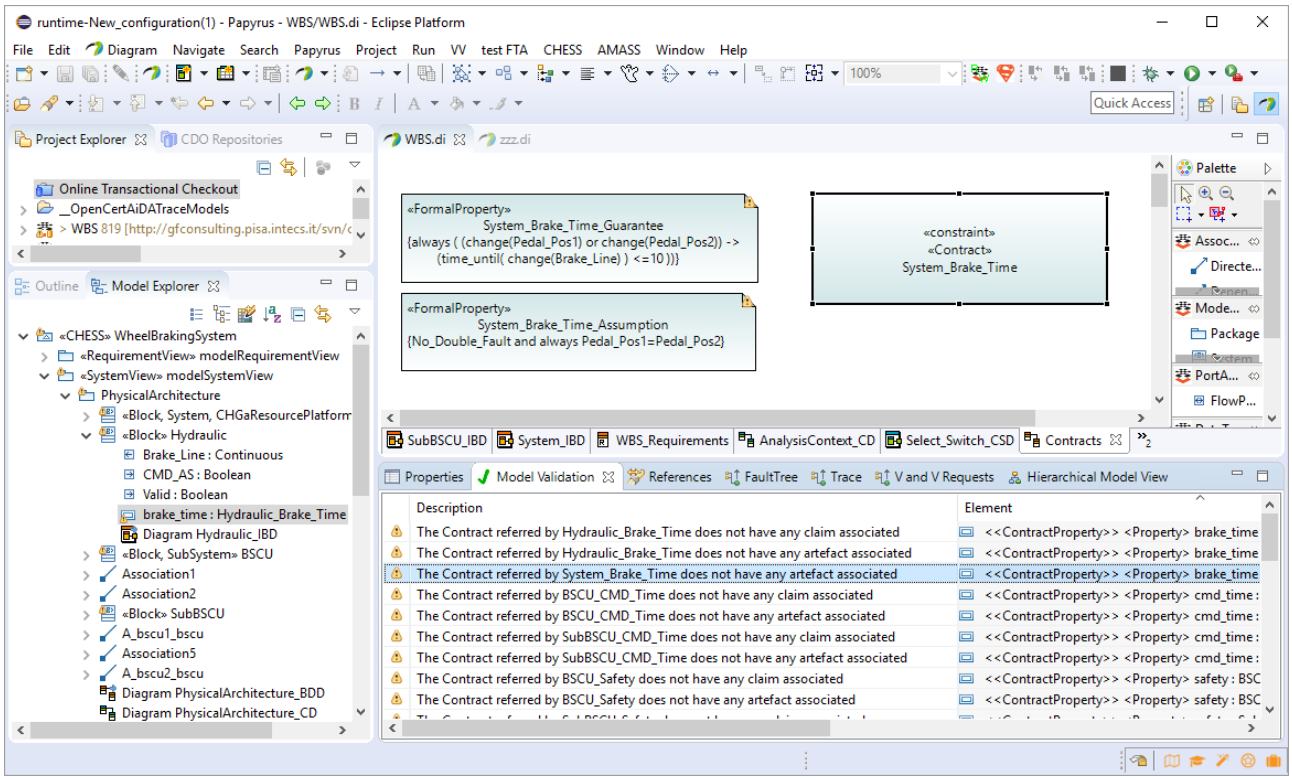


Figure 145 – Validating contracts status

7.11 Setup of External V&V Tools

In the subfolder of the bundle \V&VTools, there are builds of the V&V tools.
To enable the tools that are available locally:

- go to Window - Preferences - Model Checking – Tools, see Figure 146.
- For each V&V tools (OCRA, nuXmv, XSap)
  - Click on Browse and set the path of the executable(e.g. CHESS_directory\V&VTools\OCRA\Ocra-win64.exe).
- Click on Test, to verify that the tool is compatible with CHESS.

To get access to V&V Tools that are exposed as web services via OSLC:

- Click on the check box OSLC Enabled
- Set the URL of the service catalog
- Set the path of the service catalog
- Click on Show services catalog to verify that the web service expose the V&V tools.

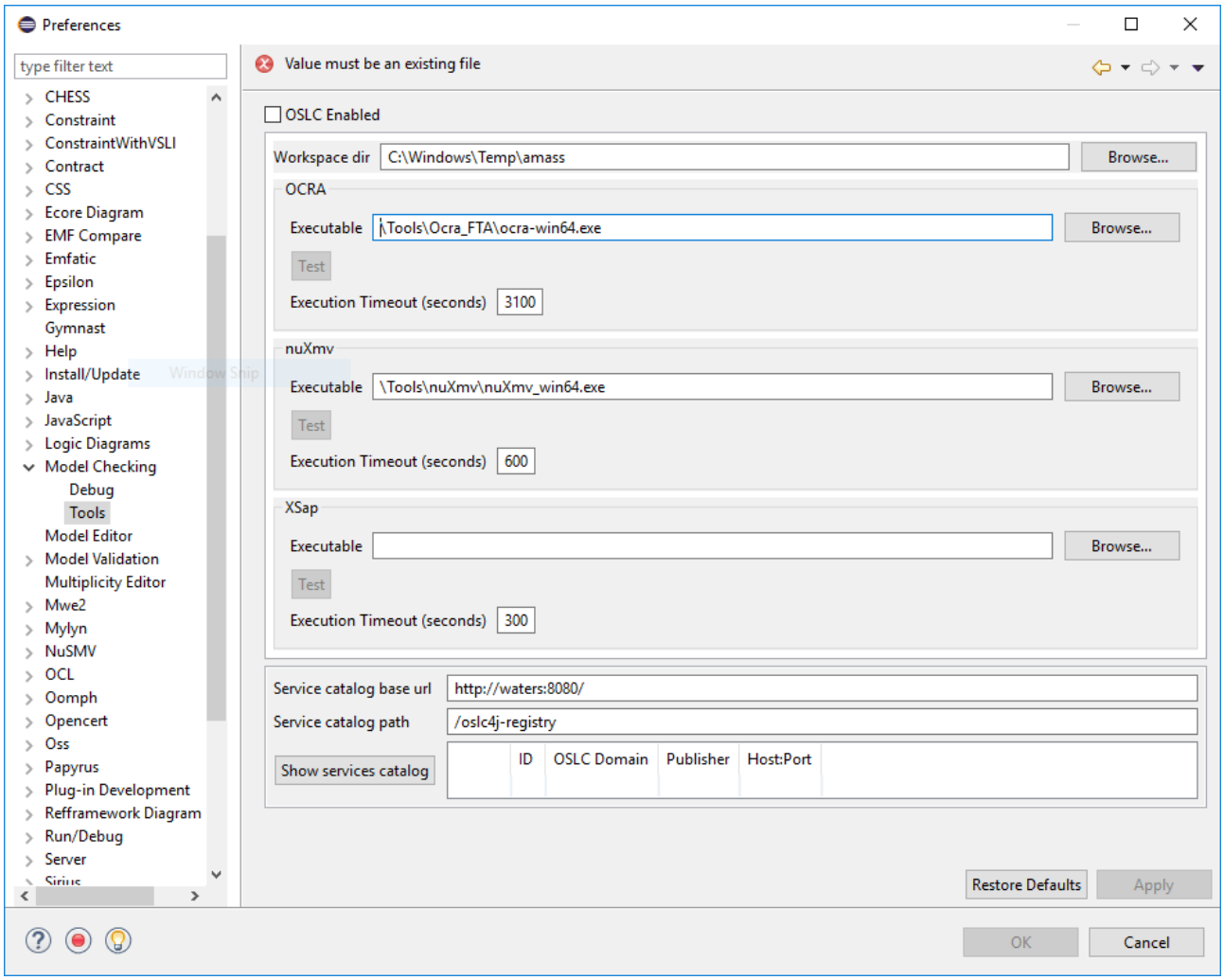


Figure 146 – The preferences window to configure the V&V tools.

7.12 Perform Check of Contract Refinement

To verify that the contract refinements are done correctly, perform the following steps:

- Select a component (in the Model Explorer View) or the corresponding graphical representation (in the diagram editor). The contract refinements considered will be the ones associated to the selected component and the ones associated to its sub components. This operation includes recursively all the contracts along the subcomponents, from the root to the leaves of the system.
- Perform the check contract refinement: right click on the selected component, then go to AMASS-Functional Verification – Check Contract Refinement on Selected Component (see Figure 147).

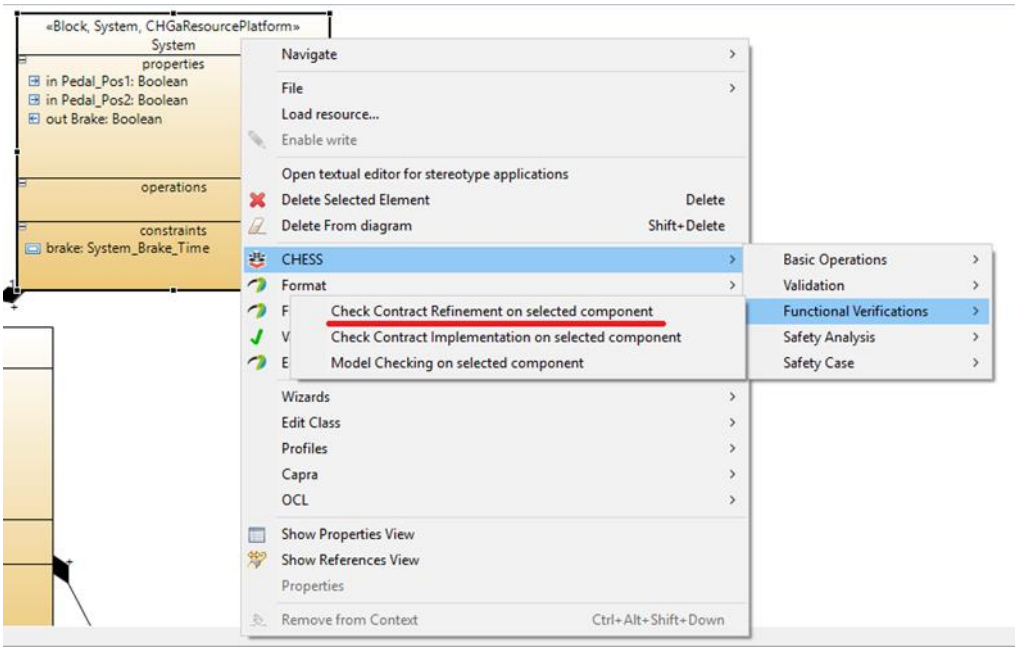


Figure 147 - Perform the check of the contract refinements

7.13 Perform Check of Component Implementation on Contracts

To verify that the state machines defined in the model satisfy the contracts, perform the following steps:

- Select a component (in the Model Explorer View) or the corresponding graphical representation (in the diagram editor). The contracts and state machines considered will be the ones associated to the selected component and the ones associated to its sub components. This operation includes recursively all the contracts and state machines along the subcomponents, from the root to the leaves of the system.
- Right click on the selected component, then go to AMASS-Functional Verification – Check contract implementation on Selected Component (see Figure 148).

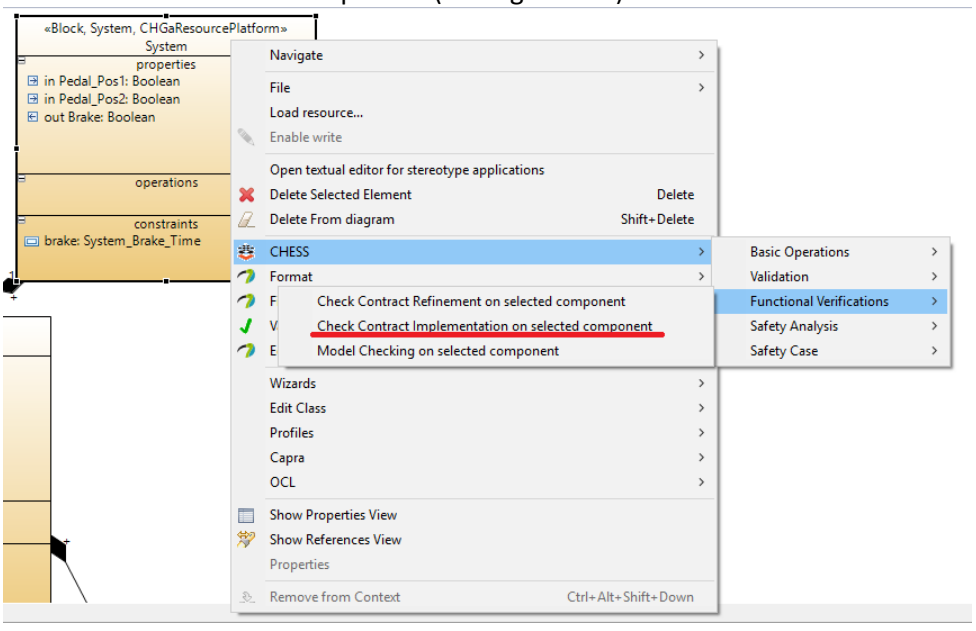


Figure 148 - Perform the check implementation based on contracts

7.14 Perform Consistency Check of Formal Properties

This validation is done by checking if a specific guarantee of the contract satisfies the assumption of another contract. To verify the formal property, perform the following steps:

- Select a component (in the Model Explorer View) or the corresponding graphical representation (in the diagram editor). The properties available to check will be the assumptions and guarantees of contracts owning to the selected component and to its sub components. This operation includes recursively all the properties from the root to the leaves of the selected component.
- Right click on the selected component, then go to AMASS- Validation – Check Validation Property on Selected Component (see Figure 149).
- A popup appears to set the parameters of the command (see Figure 150, see [7] for more details).

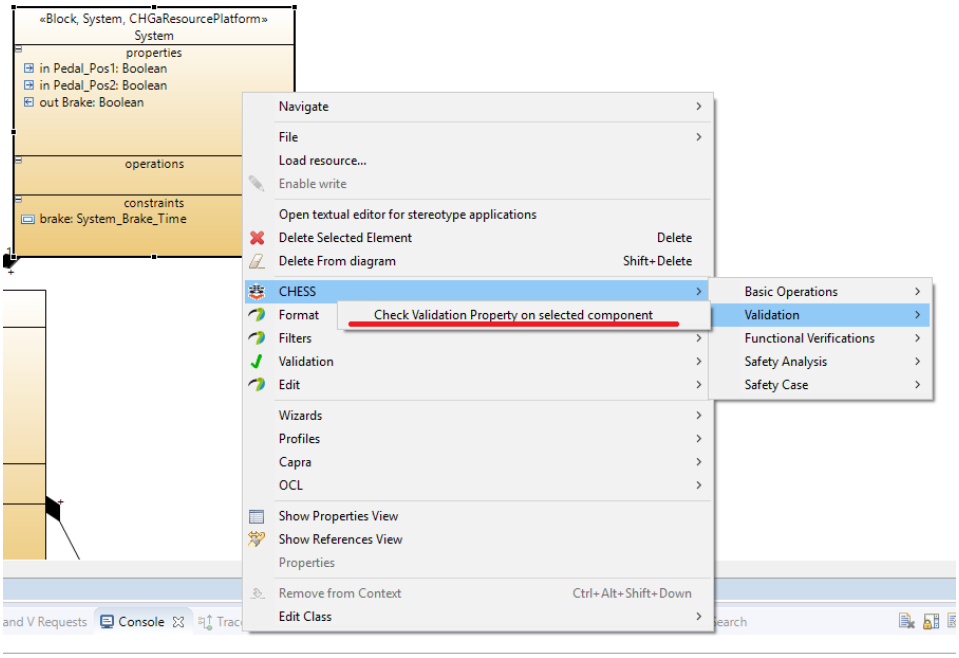


Figure 149 - Perform the consistency check of formal properties

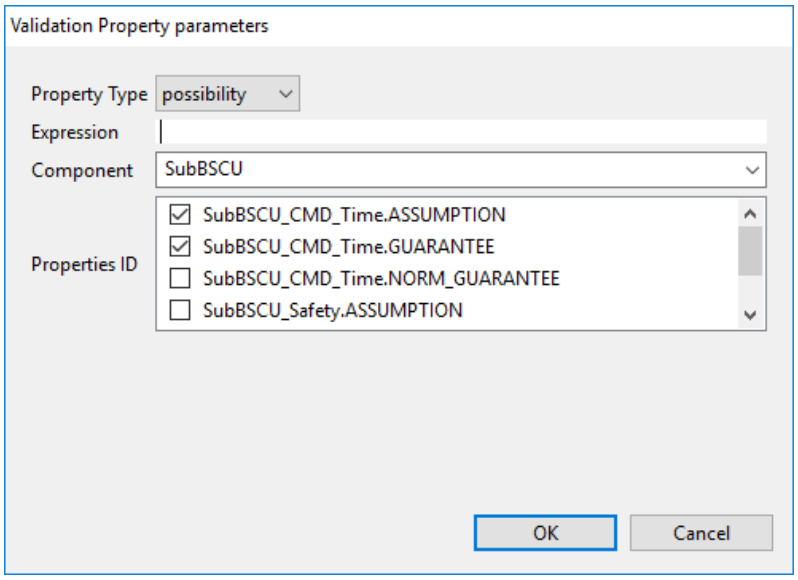


Figure 150 - Validation Property parameters

7.15 Perform Model Checking on Component Behaviour

To execute the model checking, perform the following steps:

- Select a component (in the Model Explorer View) or the corresponding graphical representation (in the diagram editor). The components behaviour to check will be the behaviour of the selected component and the behaviour of its sub components. This operation includes recursively all the behaviours from the root to the leaves of the selected component.
- Right click on the selected component, then go to AMASS-Functional Verification – Model Checking on Selected Component (see Figure 151).
- A popup appears to set the parameters of the command (see Figure 152, see [9] for more details).

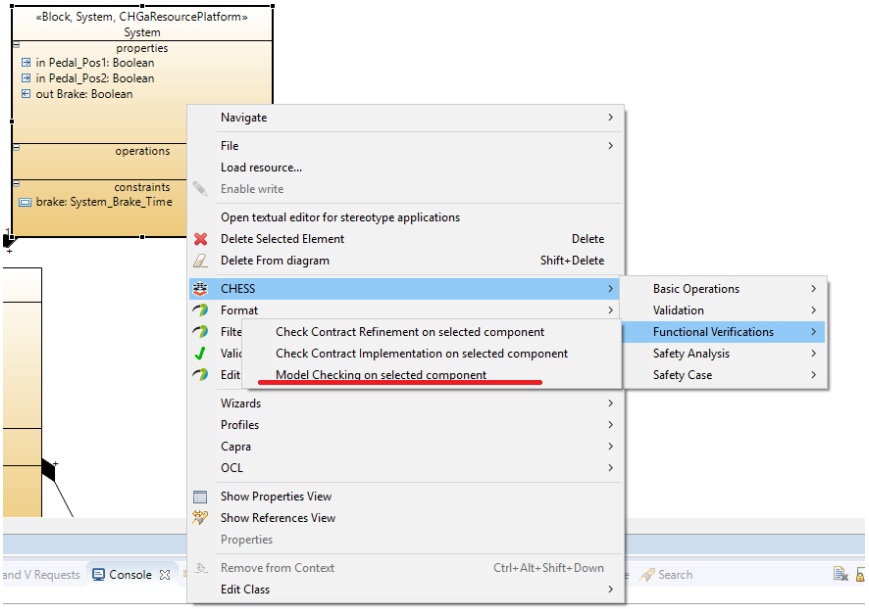


Figure 151 - Perform the model checking of system behavior

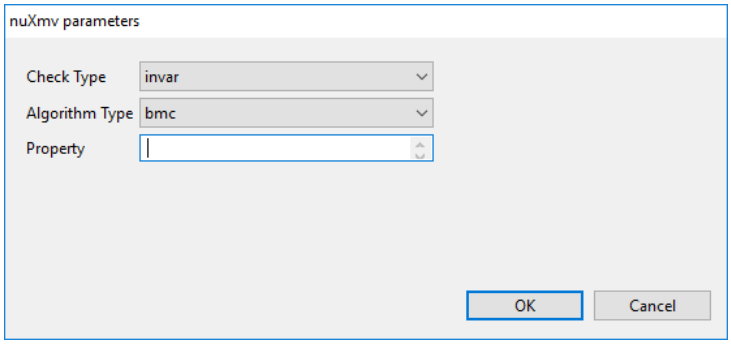


Figure 152 - Model Checking parameters

7.16 V&V Manager

An alternative approach to the verification activities described in the sections **¡Error! No se encuentra el origen de la referencia.**, **¡Error! No se encuentra el origen de la referencia.**, **¡Error! No se encuentra el origen de la referencia.**, **¡Error! No se encuentra el origen de la referencia.** is the usage of the V&V Manager. Depending on the selected model elements and on their context (e.g. the availability of

associated models) and on the available Verification Servers and verification tools the V&V Manager supports the decision making about which V&V activities shall be performed.
A plugin that is under development and provides the access to the verified component model could not be incorporated successfully into the V&V Manager. The dependent functionality was not implemented yet. Therefore, currently user have to select formal properties and interface requirements from the file system and then V&V Manager send them to Verification Servers, where are V&V tools that returns the V&V Results.
As for the planned GUI the invocation of the V&V Manager is to be performed by the following steps:

- Select a component (in the Model Explorer View) or the corresponding graphical representation (in the diagram editor). The contracts considered will be the ones associated to the selected component and the ones associated to its sub-components. This operation includes recursively all the contracts along the subcomponents, from the selected element to the leaves of the system.
- Right click on the selected component, then go to Validation -> V&V Manager.

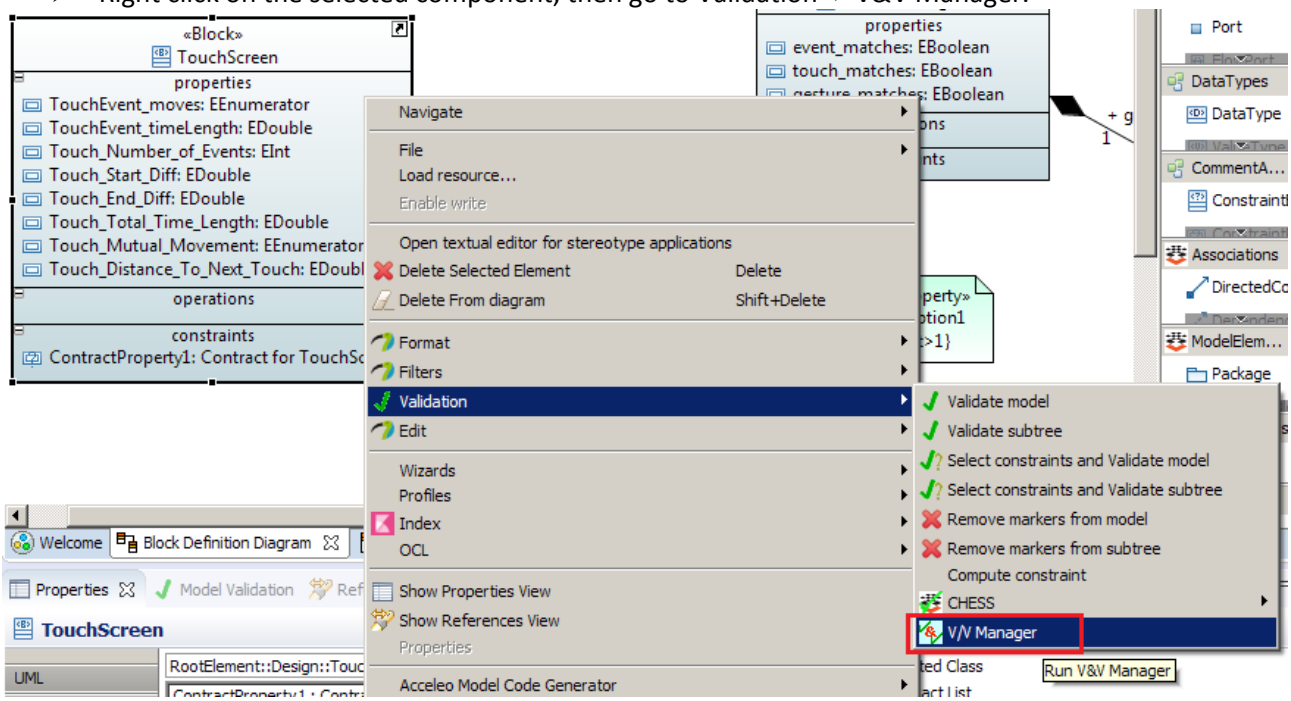


Figure 153 - Invoke the V&V Manager

7.17 Generate Contract-based Fault Tree

To execute the contract-based safety analysis, perform the following steps:

- Select a component (in the Model Explorer View) or the corresponding graphical representation (in the diagram editor). The contracts considered will be the ones associated to the selected component and the ones associated to its sub components. This operation includes recursively all the contracts along the subcomponents, from the root to the leaves of the system.
- Right click on the selected component, then go to AMASS-Safety Analysis – Contract-based Safety Analysis on selected component.

7.18 Generate Documentation

To automatically generate the documentation, perform the following steps:

- Select a component (in the Model Explorer View) or the corresponding graphical representation (in the diagram editor). The information used to generate the documentation will be related to the

selected component and to its sub components. This operation includes recursively the information from the root to the leaves of the selected component.

- Right click on the selected component, then go to AMASS-Safety Case – Document Generation-Generate documentation on selected component.
- A popup appears to set the options related to the format of the document and to the style of the diagrams (see Figure 154). For this prototype the supported format is HTML format.
- A popup appears to select the folder that will contain the document and the diagrams.

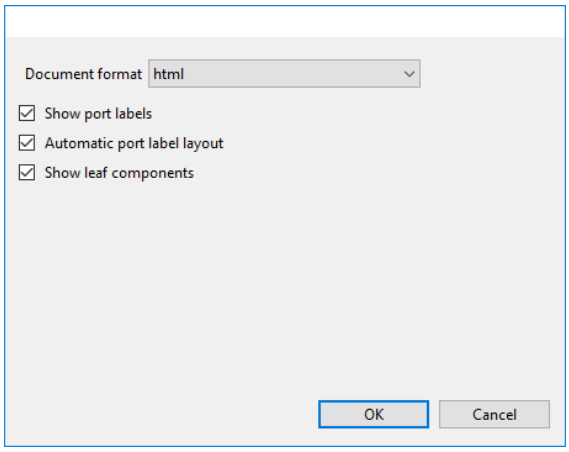


Figure 154 - Popup to set the preferences of the generated document and diagrams

7.19 Managing links between architecture and assurance

According to the AMASS component meta-model specification (see AMASS D3.2), Contracts and FormalProperty can be linked to assurance case entities, in particular to Claim and Artefact, the latter created through the AMASS platform (see sections 8 and 9).

In particular:

Contract has the following relationships:

- supportedBy: Artefact [0..*]
  Allows to model that a Contract statement, in particular its guarantees, can be supported by artefacts (e.g. the latter referring some verification results)
- claim: Claim [0..*]
  Allows to further clarify a contract statement; e.g. that the contract is derived from some analysis or is based on some specification
- agreement: Agreement [0..*]
  the agreement owns the arguments about how the assumption of a contract are fulfilled in the context of the given system

Moreover the Contract entity is extended with the following attributes which allow to attach information to the contract that can be of help for the assurance:

- contextStatement: String
  store the informal description of what the contract means (which would be the context statement in the corresponding argumentation)
- artefactStatement: String
  explain how a particular artefact relates to the contract

FormalProperty entity has following relationships:

- claim: Claim [0..*]
  Allows to map the guarantees of the contract to claims
  Allows to associate a claim (e.g. GSN away goal) to each of the contract's assumptions

Component and Block is extended with the following relationship:

- referenceArgumentation: ArgumentationElement
  the arguments associated to the block instance

AnalysisContext is extended with the following relationships:

- artefact: Artefact [0..*]
  allows to associate the artefact produced by the analysis that have been invoked through the analysis context

Given the availability of Claim and Artefact entities in the AMASS database the following steps can be performed to create the association:

1. In the CDO Repositories view, connect to the existing remote AMASS repository owning the assurance case entities. In the New Repository wizard (see Figure 155) set the proper value for the Host and Port attributes, according to the AMASS server configuration, and set "opencert" as repository name, then click Finish.

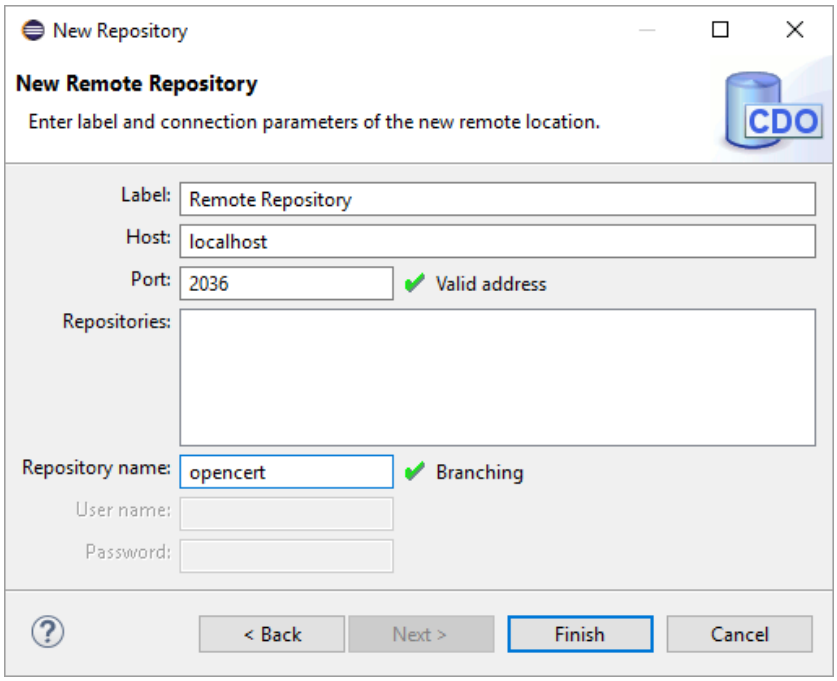


Figure 155 - Connecting to the AMASS repository

2. Right click on the just created Remote Repository and select Checkout. Navigate the checkout in the Project Explorer view to retrieve the assurance-related entity to trace.

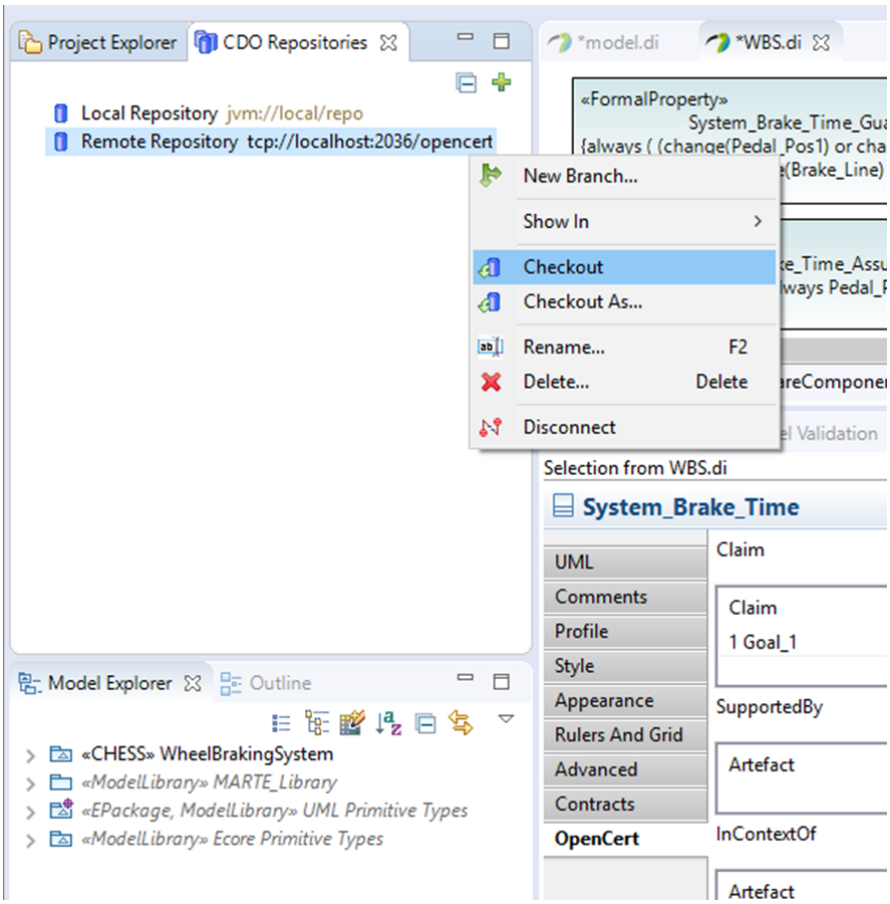


Figure 156 - CDO Repositories view

3. Select the entity in the CHESS model to be linked
4. Open the OpenCert tab in the Properties view. The OpenCert tab shows the relationships that can be created.
5. Pin the Properties view

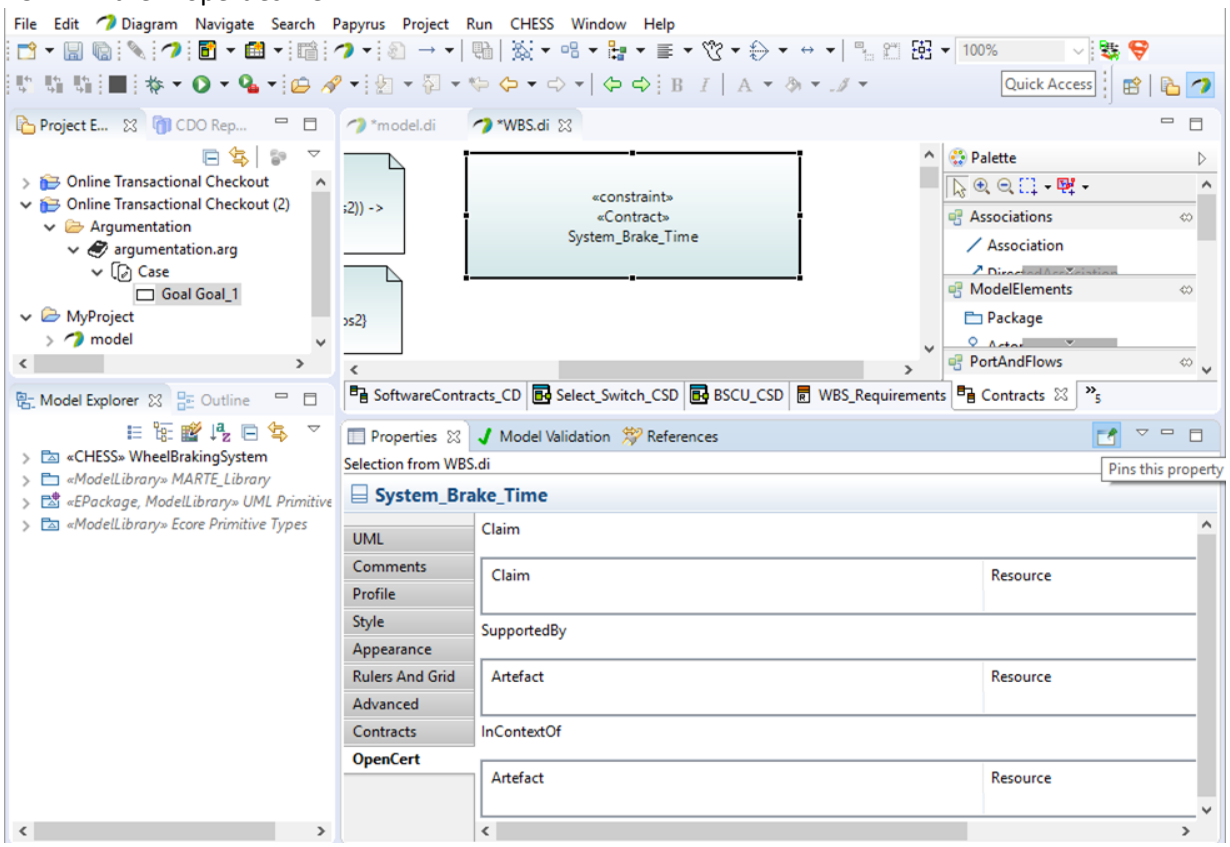


Figure 157 - OpenCert tab for the Contract

6. Drag the assurance related entity (Claim, Artefact, Agreement, ArgumentationElement) from the Project Explorer view to property area of the OpenCert tab.

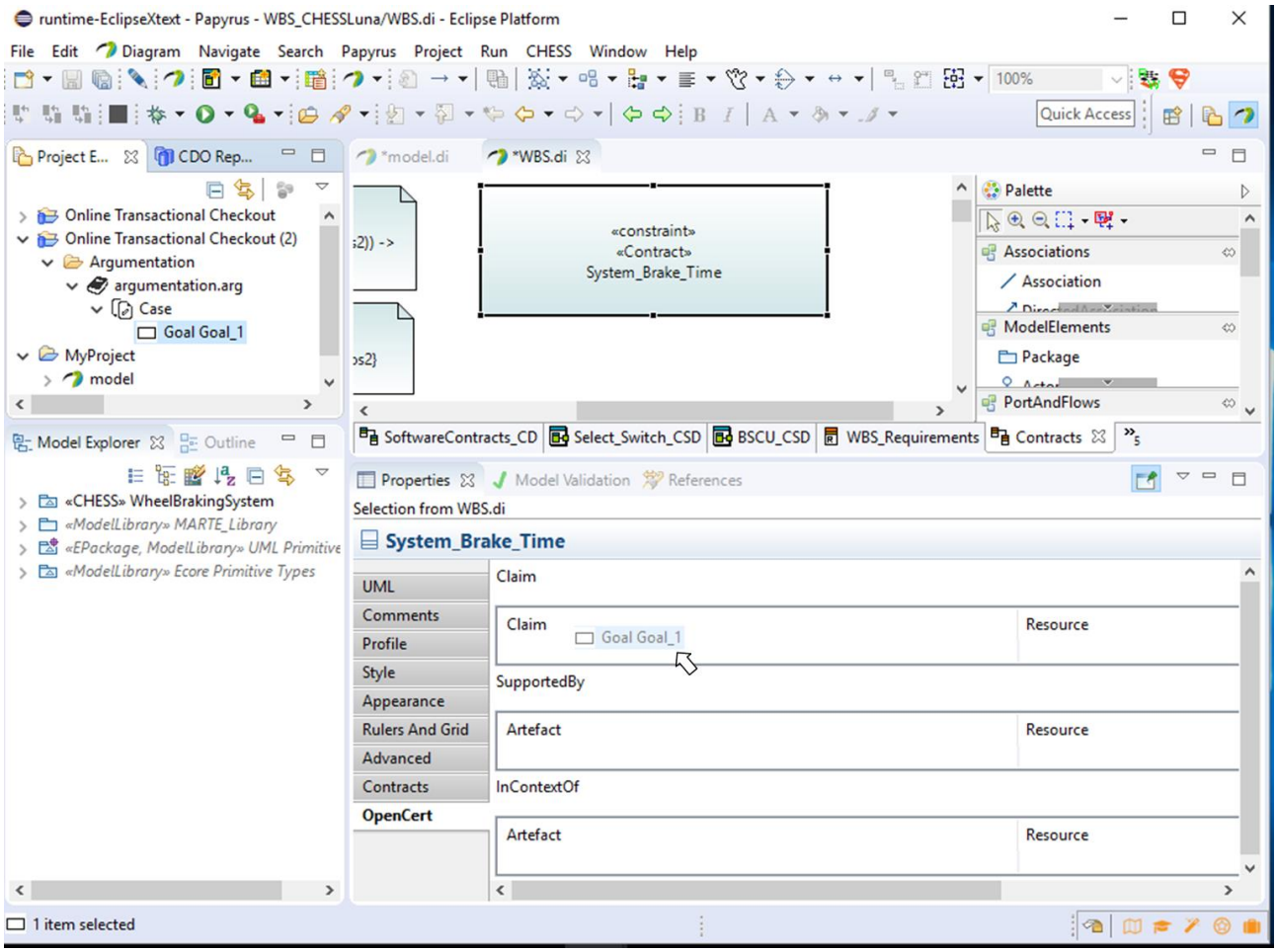


Figure 158 - associating a Claim to a Contract

7. A link is created. To delete the link, select the entity in the OpenCert tab and click on the delete button available on the right of the OpenCert tab.
8. Double click the assurance related entity in the OpenCert tab to select the corresponding entity in the ProjectExplorer.

NOTE: the aforementioned support about the creation/navigation/delete of links between Contract/FormalProperty and Claim/Artefact (the latter available through CDO) works with CDO or file based Papyrus model.
In case of usage of file-based Papyrus models, if the checkout of the AMASS repository is deleted from the workspace and created once again later, it is possible that the links already created between contracts and assurance entities cannot be retrieved in the OpenCert CHESS tab. In this case the following procedure has to be performed on the Papyrus .uml model file:

1. Make sure you have the Papyrus “Di view” deselected:
    a. From the ProjectExplorer View, click on the top-right arrow, choose “Customize View…” command and from the AvailableCustomizations view make sure that the “Di view” is not selected.
2. Select the checkout from the ProjectExplorer view, in the Properties view check the value for the ID property
3. Make sure the CHESS model is not currently open with the Papyrus graphical editor. From the ProjectExplorer view, right click on the .uml model file and select CHESS→Synchronize with CDO checkout

4. In the SynchronizeWithCDOCheckout window write the ID of the checkout you want to work with and press ok

NOTE: The links manged through the OpenCert tab are stored in a local traceability model which is automatically created and maintained in the current workspace.

8 Assurance Argumentation Management

8.1 Preferences

Set some configuration Parameters in Window→Preferences →OpenCert→Argumentation.
In the section, you can define parameters required by the Argumentation diagram editor.
The parameters which can be defined are below:

- Modules directory preference. This folder contains all argumentation modules stored from previous argumentation.
- Patterns directory preference. This folder contains all argumentation patterns templates.
- Patterns CDO directory preference. This folder contains all argumentation patterns templates stored at the repository.
- Modules CDOdirectory preference. This folder contains all argumentation modules stored from previous argumentation at the repository.

These folders are Eclipse Projects that should be created previously.

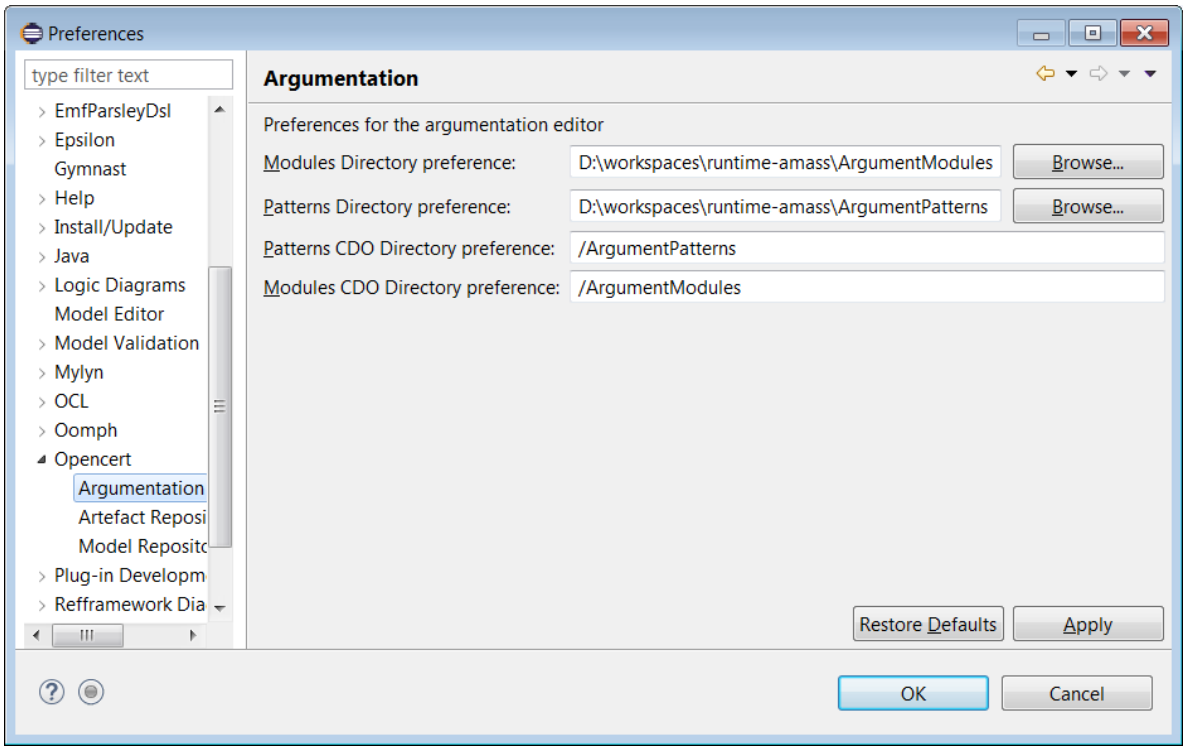


Figure 159 - Argumentation Preferences

8.2 Creating and Saving a Diagram

We are able to work in two ways. Either by creating assurance cases locally in the workspace. This is done specifically when creating argument patterns which are described in Section 8.4. In the other way the assurance cases are stored in a database which let the users work in a cooperative and distributed way.

8.2.1 Creating a New Diagram

In File Format

To create a new file-based argumentation diagram, follow the procedure below and generate a new diagram in the project folder.

Select "New" → "Other…" from the File menu of eclipse → Select "Arg Diagram to File" in the OpenCert category and click the "Next" button. → Select the folder to store the new diagram → Enter the name of the diagram to be created and click the "Finish" button.

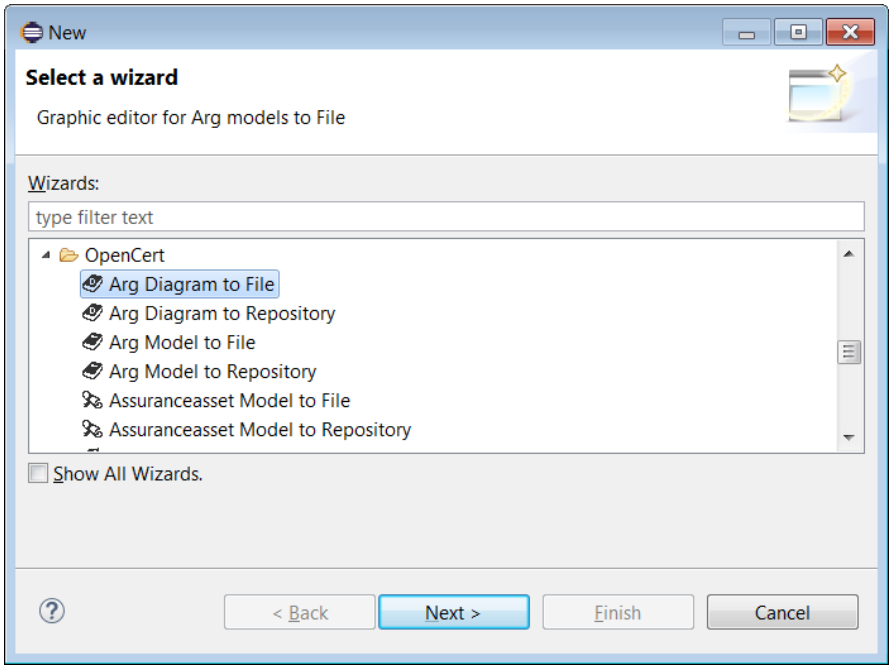


Figure 160 - File-based Argumentation Diagram wizard I

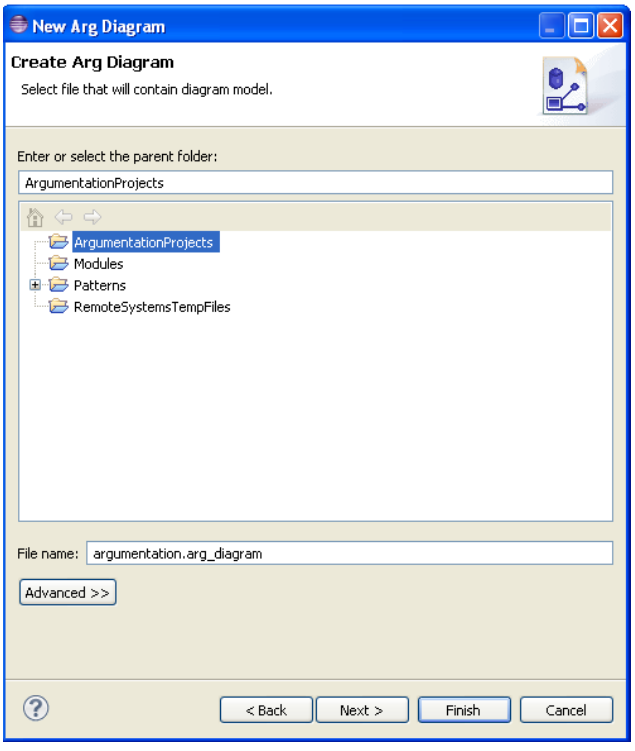


Figure 161 - File-based Argumentation Diagram wizard II

In Database Format

To create a new database-based argumentation diagram, follow the procedure below and generate a new diagram in the project folder. This allow the user to work in a collaborative manner.

Select "New" → "Other…" from the File menu of eclipse → Select "Arg Diagram to Repository" in the OpenCert category and click → Select the folder to store the new diagram → Enter the name of the diagram to be created and click the "Finish" button.

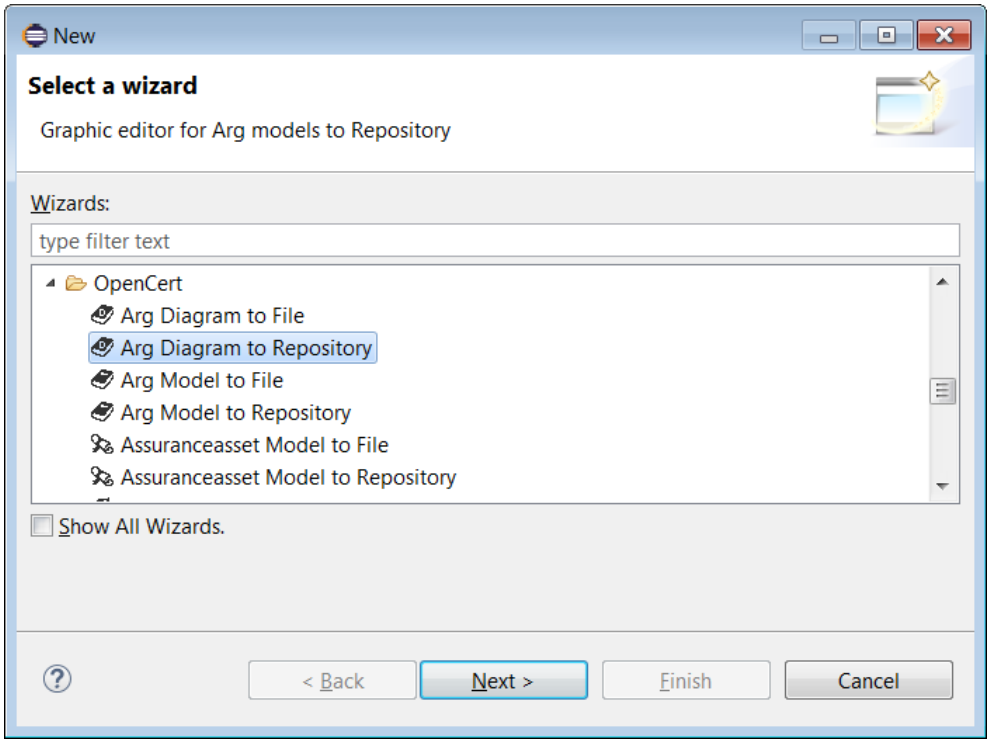


Figure 162 - Database-based Argumentation Diagram wizard I

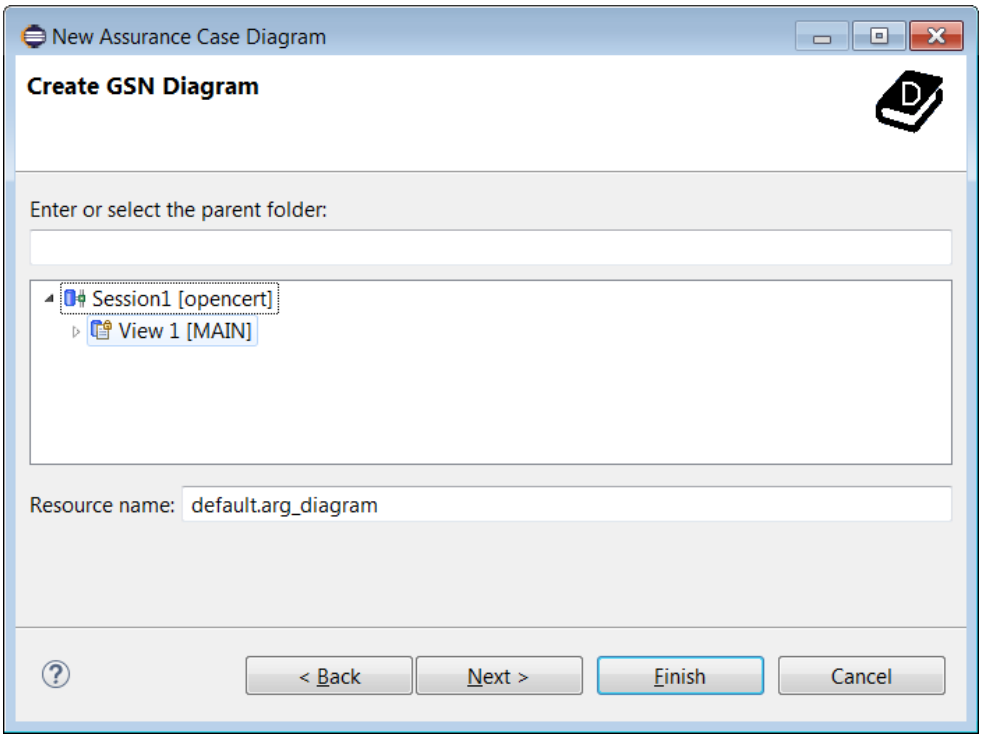


Figure 163 - Database-based Argumentation Diagram wizard II

8.2.2 Creating a Diagram at the project creation time

When we create a new assurance project (See section 5.1 Create an assurance project and a Baseline) a new argumentation is created. This argumentation is created applying a transformation to the baseline. The rules that apply to the transformation are the following:

- Every reference activity is transform into a claim. The id, name and description of the reference activity become also the id, name and description of the claim.
- Every reference requirement is transform into a claim. The id, name and description of the reference requirement become also the id, name and description of the claim.
- Every reference artefact is transform into an information element with the property type marked as solution. The id, name and description of the reference artefact become also the id, name and description of the information element.
- When a reference activity has sub activities, then an asserted inference relationship is created, the source is the claim transformed from the top activity and the target the claim transformed from the sub activity
- When a reference activity has reference requirements, then an asserted inference relationship is created, the source is the claim transformed from the reference activity and the target the claim transformed from the requirement

8.2.3 Opening a Diagram

In File Format

Double click on the project folder on the Package Explorer tab in order to expand the folder. The stored diagrams will be shown. Double click on Argumentation Diagram information file (.arg_diagram) to open a diagram in the editing window. The diagram can then be edited.

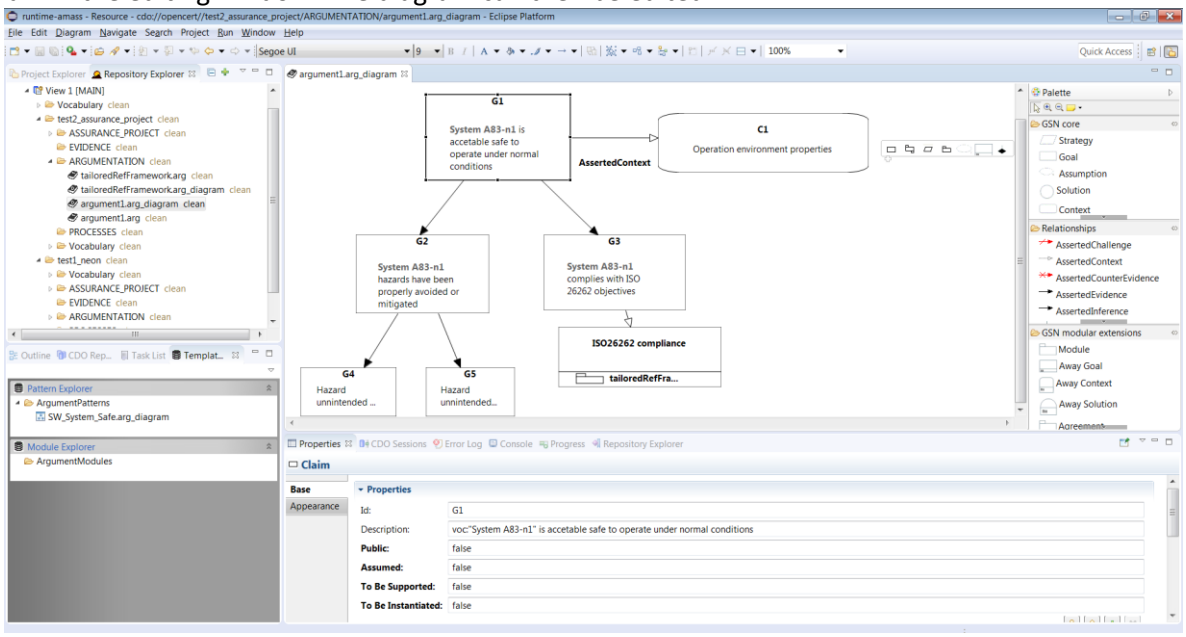


Figure 164 - Open File-based Argumentation Diagram

In Database Format

Double click on the project folder on the Repository Explorer tab in order to expand the folder. The stored diagrams will be shown. Double click on Argumentation Diagram information file (.arg_diagram) to open a diagram in the editing window. The diagram can then be edited.

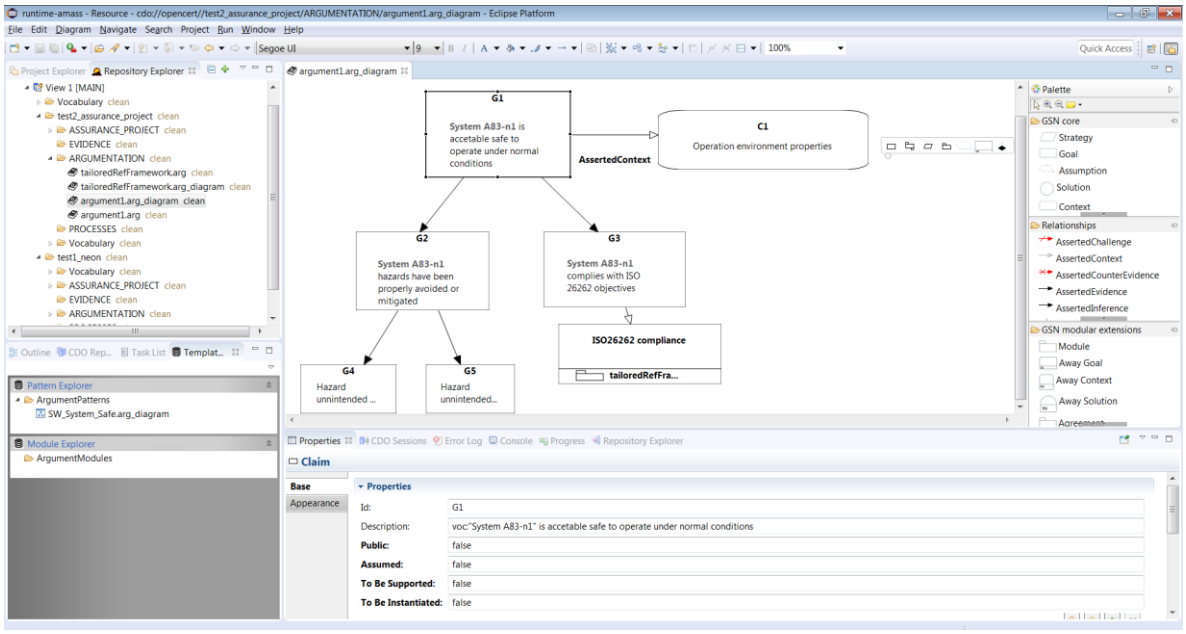


Figure 165 - Open Database-based Argumentation Diagram

8.2.4 Saving a Diagram

To save a diagram, select one of the following items in the “File” menu.

1. “Save” item
   This utility is only available for file-based Argumentation Diagram.
   The contents of the selected editing window will be saved in the model information file and the diagram information file.
2. “Save As...” item
   This utility is only available for file-based Argumentation Diagram.
   The contents of the selected editing window will be saved in the model information file and the diagram information file with a different name.
3. “Save All” item
   The contents of all editing windows will be saved in the corresponding model and the diagram information files/database.

8.3 Editing Functions

8.3.1 Editing a Diagram

Nodes and relationships (or links) selected from Palette can be added to the canvas. Just select the node from the Palette, go to the editing window and select the place and size of the element.

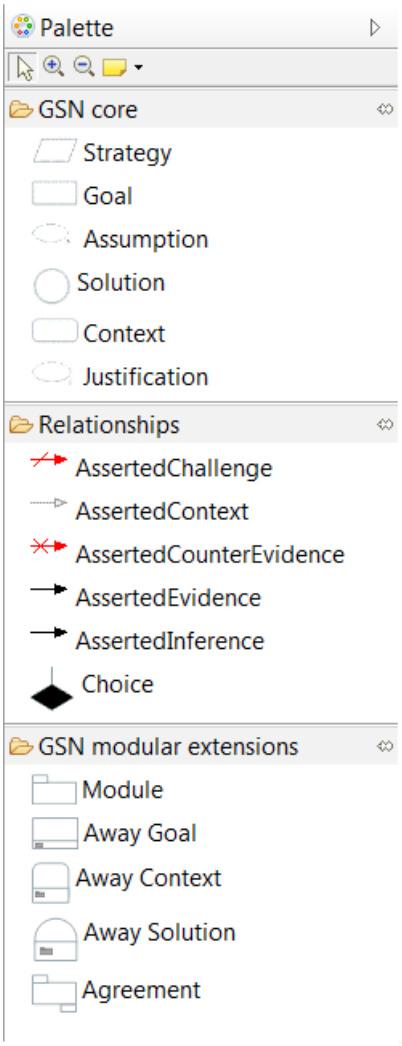


Figure 166 - Argumentation Palette

The palette is structure into three different sections. Section "GSN core" includes the main nodes for argumentation. These nodes implement the GSN graphical notation, however internally it uses the SACM metamodel. The "relationships" includes all the different links between the different nodes. "GSN modular extensions" includes those nodes specific for the modular argumentation.

Graphical notation	GSN concept	Extended SACM metamodel used in CACM
	Goal	Claim
	Context	InformationElementCitation Property type="context"
	Strategy	ArgumentReasoning
	Solution	InformationElementCitation Property type="solution"
	SolvedBy	AssertedInference, but only if the target is not a solution
	SolvedBy	AssertedEvidence, but only if the target is a solution
	InTheContextOf	Asserted

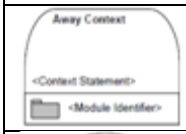
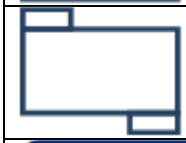
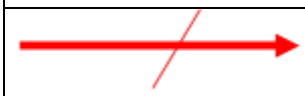
	Underdeveloped	Property toBeSupported=”true”
	To be instantiated	Property toBeInstantiated=”true”
	AwayGoal	ArgumentElementCitation Property type=”claim”
	AwayContext	ArgumentElementCitation Property type=”context”
	AwaySolution	ArgumentElementCitation Property type=”solution”
	Module	Argumentation
	Contract	Agreement
	Assumption	Claim Property assumed=true
	Justification	InformationElementCitation Property type=”justification”
	N/A	AssertedCounterEvidence An AssertedCounterEvidence by Claim (A – the source evidence cited – and B – the target claim) denotes that the evidence cited by A is counter-evidence to the truth of Claim B (i.e., Evidence A suggests the conclusion that Claim B is false).
	N/A	AssertedChallenge An AssertedChallenge by Claim A (source) to Claim B (target) denotes that the truth of Claim A challenges the truth of Claim B (i.e., Claim A leads towards the conclusion that Claim B is false)
	Public Goal	Claim Property Public=”true”
	Optionality	AssertedInference Property multiplicity=optional
	multiplicity	AssertedInference Property multiplicity=multi
	Choice	Choice

Table 2 - Argumentation graphical notation

In case of the links, just select the link from the palette, then on the editing window click on the source of the link and then release the click on the target of the link.
Popup menu opens when the mouse cursor is placed in the graphic-object editing area and is kept still for a moment. A node can be created by selecting the corresponding icon in the menu.
On the other hand, Properties View manages the properties of the current element under edition.

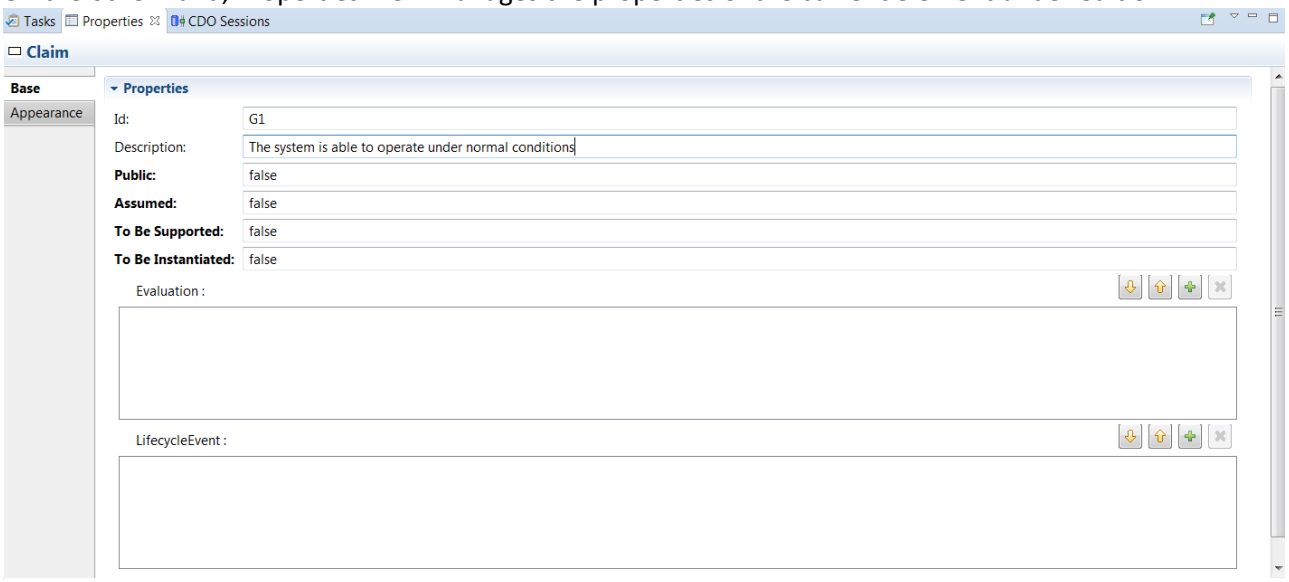


Figure 167 - Claim properties

Goal
These elements have the graphical notation as Goals on GSN and are stored as Claim in CACM metamodel. The identifier property indicates uniquely the goal on the argumentation. Do not use the same identification on different elements on the same argumentation.
To be supported property indicates that the claim will be further developed on a future.
Assumption
In the palette is considered as a core element; however in CACM it is stored as a claim with assumed property as true. It references the assumption concept and has the same graphical notation assumptions in GSN. It indicates an assumption in relation of a goal.
Strategy
This element has the same graphical notation as strategies on GSN but it is stored as Argument Reasoning in CACM. The identifier property indicates uniquely the element on the argumentation. Do not use the same identification on different elements on the same argumentation.
To be supported property indicated that the argument reasoning will be further developed on a future.
Justification, Context and Solution
The Information Element Citation concept stored in CACM, has different graphical notations depending to the concept which is referencing. The property “Type” could have the values

- Justification: it references the justification concept and has the same graphical notation as justifications in GSN. It justifies the validity of a claim.
- Context: it references the context concept and has the same graphical notation as contexts in GSN. It indicates the context of a claim.
- Solution: it references the evidence concept and has the same graphical notation as solutions in GSN. It supports the validity of a claim.

The identifier property indicates uniquely the element on the argumentation. Do not use the same identification on different elements on the same argumentation.
Information elements references to a specific artefact. The url property indicates the location of the artefact associated. See section “Connecting a Diagram to Artefacts” for further details for information on these artefacts.

Argumentation Modules

This element has the same graphical notation as argument modules on GSN. They are used when the assurance argumentation is done in a composable way. The identifier property indicates uniquely the element on the argumentation. Do not use the same identification on different elements on the same argumentation.

The Location attribute indicates where is stored the argumentation diagram file with the content of the module. By default is should be stored on the places indicated on the preferences.

8.3.1.1 Copying and Pasting an Element

Elements in a diagram can be copied and pasted.

To copy an element, select the element and click “Copy” in the “Edit” menu. The copied element can be pasted by clicking “Paste” in the “Edit” menu without selecting any element.

8.3.1.2 Deleting a Node or a Link

Please, do not press the Del key. The element will not be completed deleted.

To delete a node or a link, click to select the node/link and perform either of the following steps.

1. Press the “BS” key.
2. Right click the item and select “Delete from Model” from the context menu.

PS: An argument can be deleted. But child elements are also deleted and the argument cannot be edited ever.

8.3.2 Create multi-diagrams from an Argumentation model

The tool allows managing different views of a model through a set of diagrams. Once a model is available, a new diagram view can be created and special edition functionalities are available as follows:

1. Thanks to the Outline view, it is possible to drag and drop concepts from the model to the diagram.
2. Once a concept has been selected, it can be hidden through the “Delete from diagram” option available in the contextual menu. This option does not delete the concept from the model.
3. Once a concept has been selected, it can be deleted through the “Delete from model” option available in the contextual menu. This option delete the concept from the model permanently. If this deleted concept is visible in another diagram files, this concepts will be shown with a cross icon in the upper right corner to show that it does not exist anymore.

Create multi-diagrams in File Format

Once a model is available, a new diagram view can be created by using the “Initialize arg_diagram diagram file” option in the contextual menu.

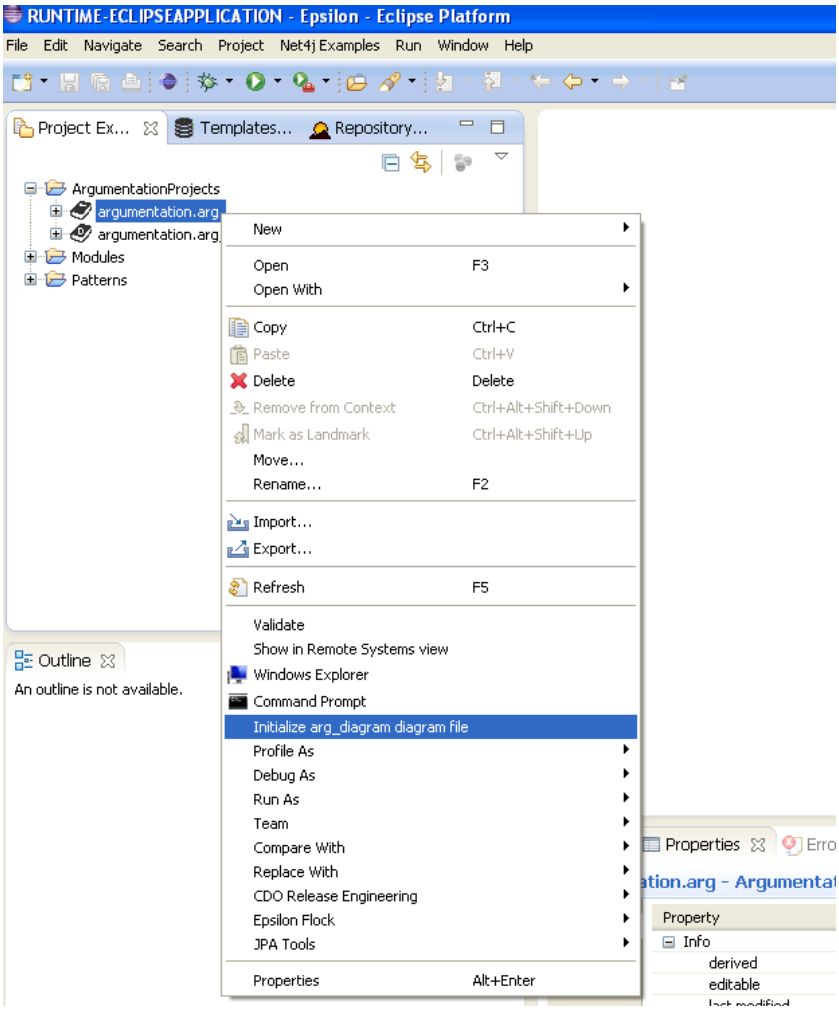


Figure 168 - Initialize a diagram file

This option launches a wizard that at the first step requires the folder and the name of the new diagram file. Next, the root element of the model (Case type) must be selected as the root element of the new diagram. After that, the diagram is ready for edition.

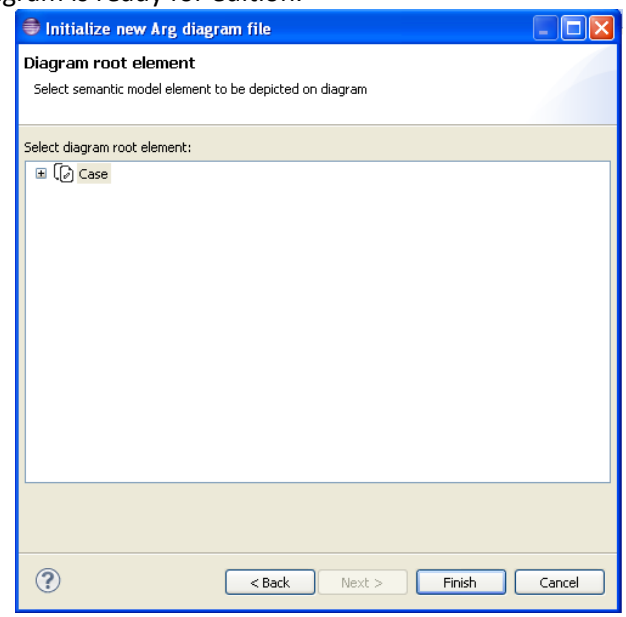


Figure 169 - Selection of the Case root element

Create multi-diagrams in Database Format

Once a model is available, a new diagram view can be created following the procedure below.

Select "New" → "Other…" from the File menu of eclipse → Select "Arg Diagram to Repository" in the Opencert category and click the "Next" button. → Select the folder to store the new diagram. Then type the name of the diagram to be created and click the "Next" button. → Type the name of the model created previously and click the "Finish" button.

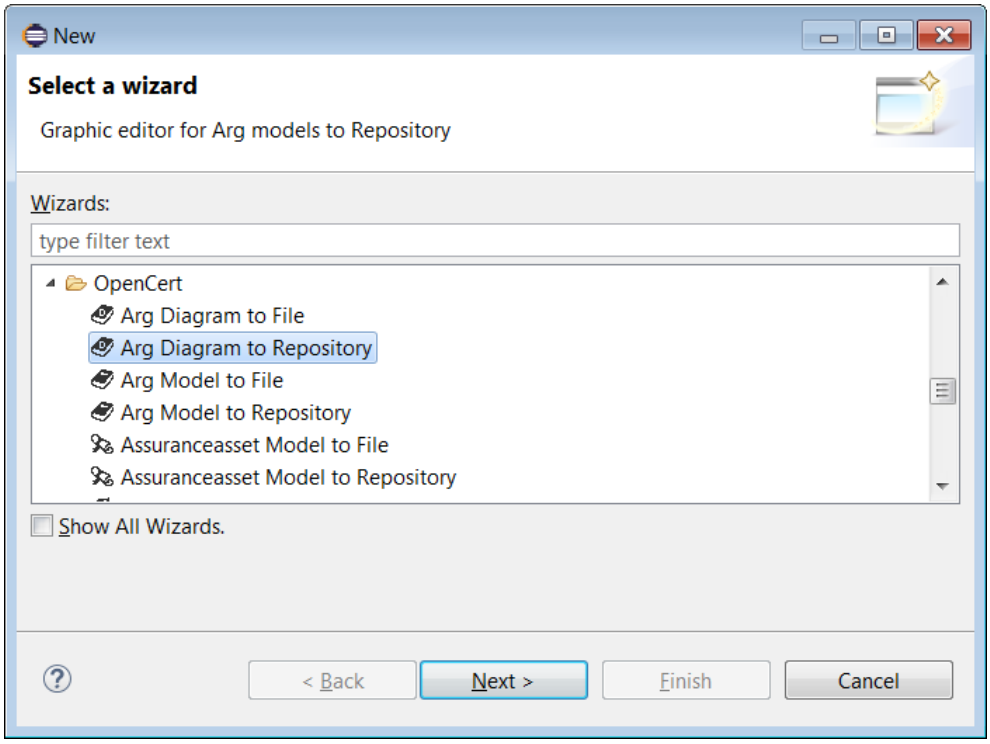


Figure 170 - Database-based Argumentation Diagram wizard I

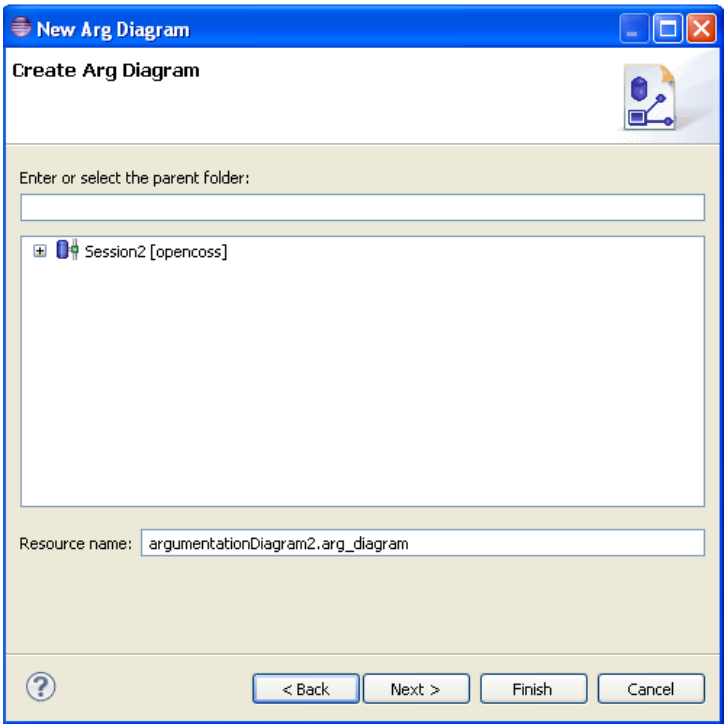


Figure 171 - Database-based Argumentation Diagram wizard II

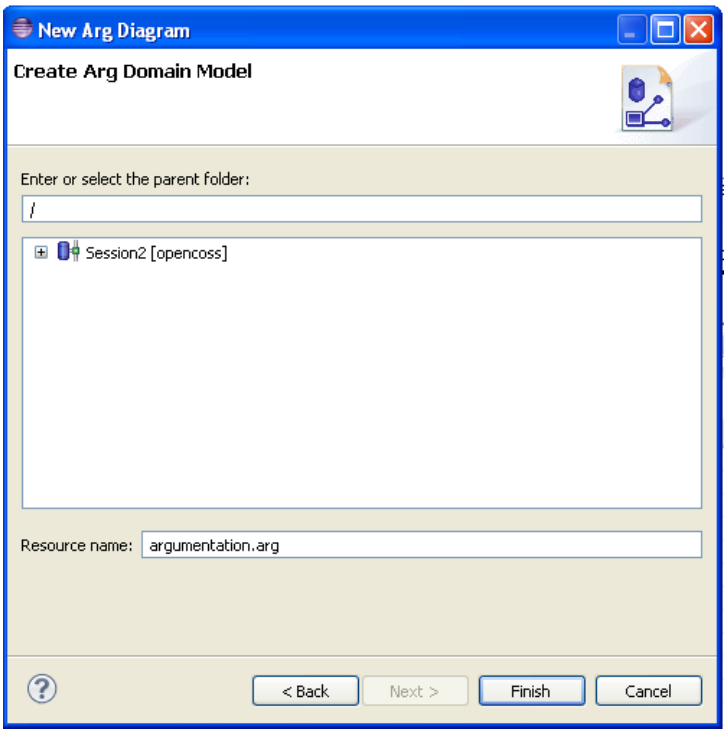


Figure 172 - Database-based Argumentation Diagram wizard III

After that, the diagram is ready for edition.

8.3.3 Connecting an Argument Diagram to Artefacts

First we need to load the evidence model to be linked with. This step is done differently depeding if the argument diagram is file based, or databased stored. See the next subsections for more information.
Once the evidence model is loaded; context, solution or justification elements can be related to its artefacts. To carry out, select one of these elements in the diagram and in the properties views press "+" operator in Artefact section.

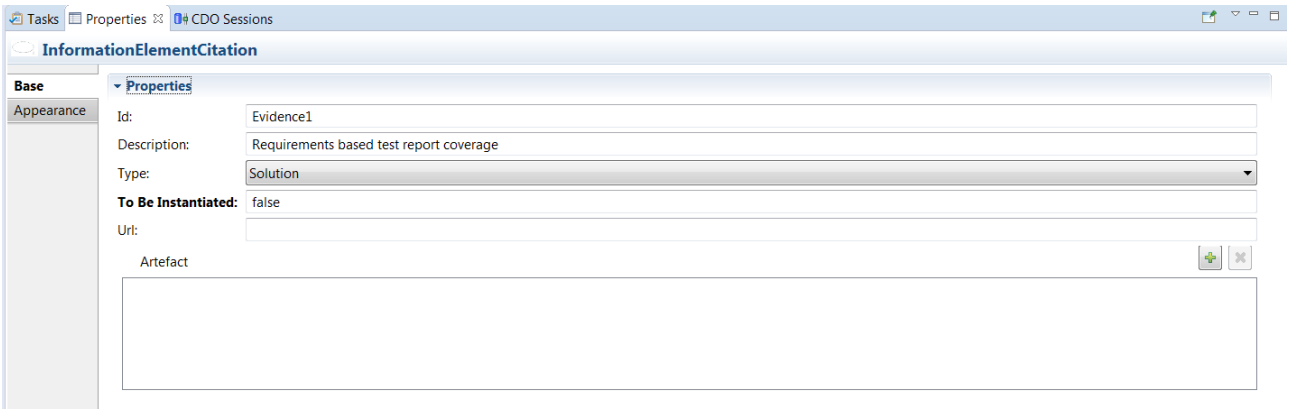


Figure 173 - Artefact selection as solution

A pop-up window will show the Artefacts Model at "All Resources" tab. Finally, choose the required artefact.

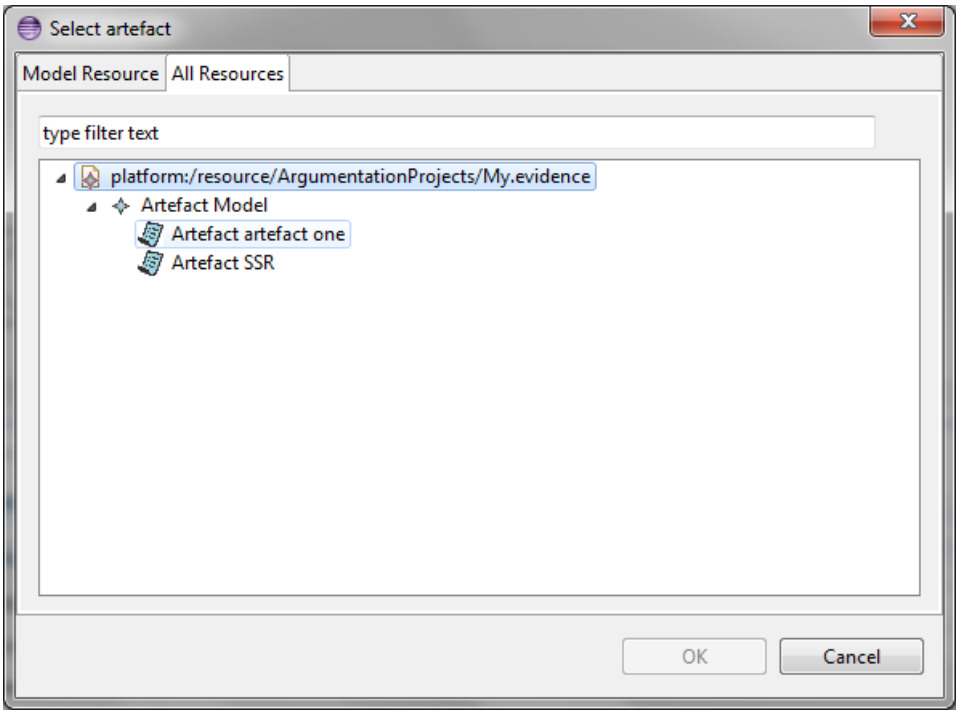


Figure 174 - Artefact selection from resources

In addition, it is possible to launch the artefact editor by double click on one artefact instance as shown below.

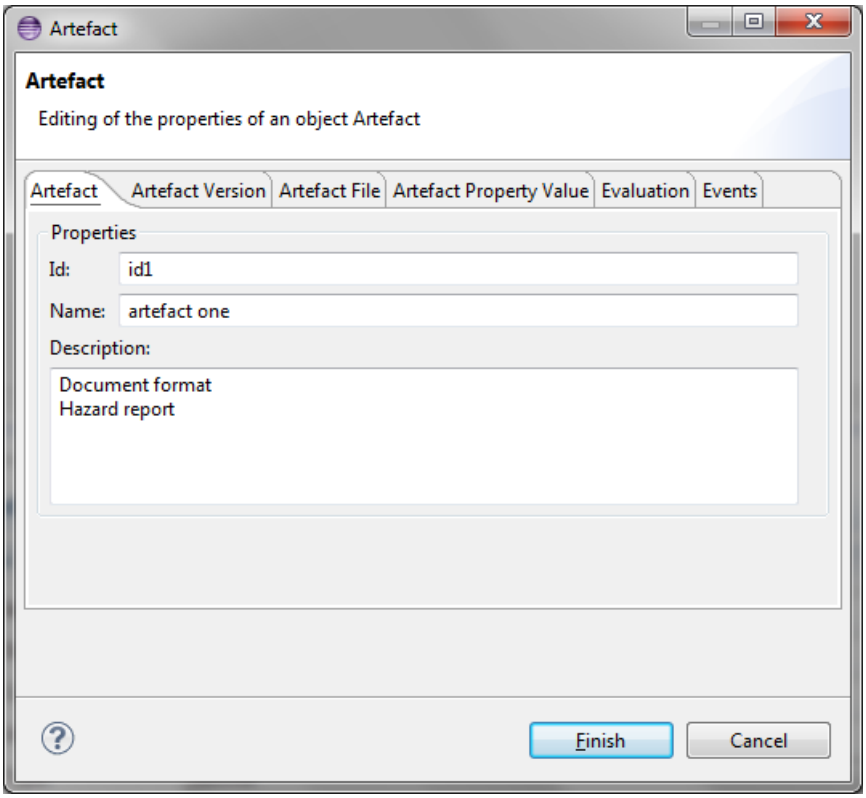


Figure 175 - Artefact edition form

8.3.3.1 Load evidence models in a File based Diagram

Firstly, proceed to load the evidences model (.evidence) from the repository. So, press the Outline and select “Load Resource” in the context menu.

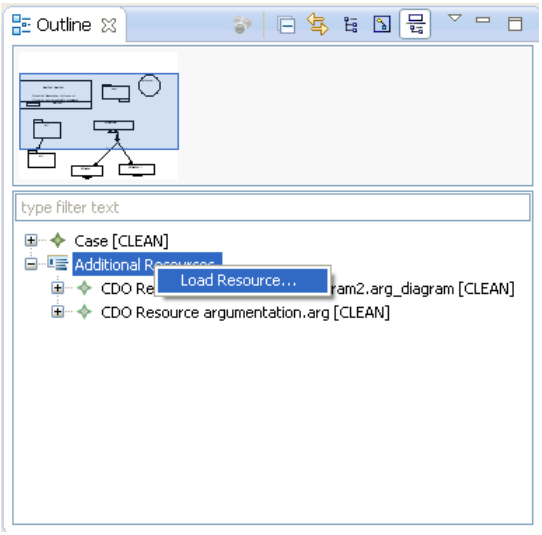


Figure 176 - Load Resource to Argumentation Diagram

Then browse into the workspace to select the evidences model resource.

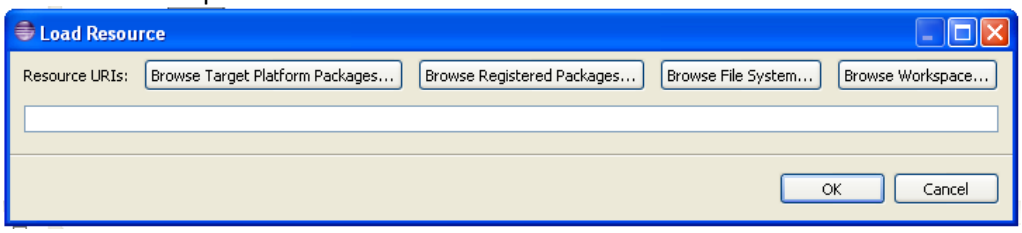


Figure 177 - Select Evidence model as resource

8.3.3.2 Connecting a Data based Diagram to Artefacts

We must load CDO resources for the artefact model (.evidence). We shall do it by including them ub rge Assets AssetsPackage of the Assurance Project. In the repository explorer, we select the .assuranceproject model. In the tree view we select the AssetPackagage and in the properties view we can add the models, such as the evidence models or the argument models to the project.

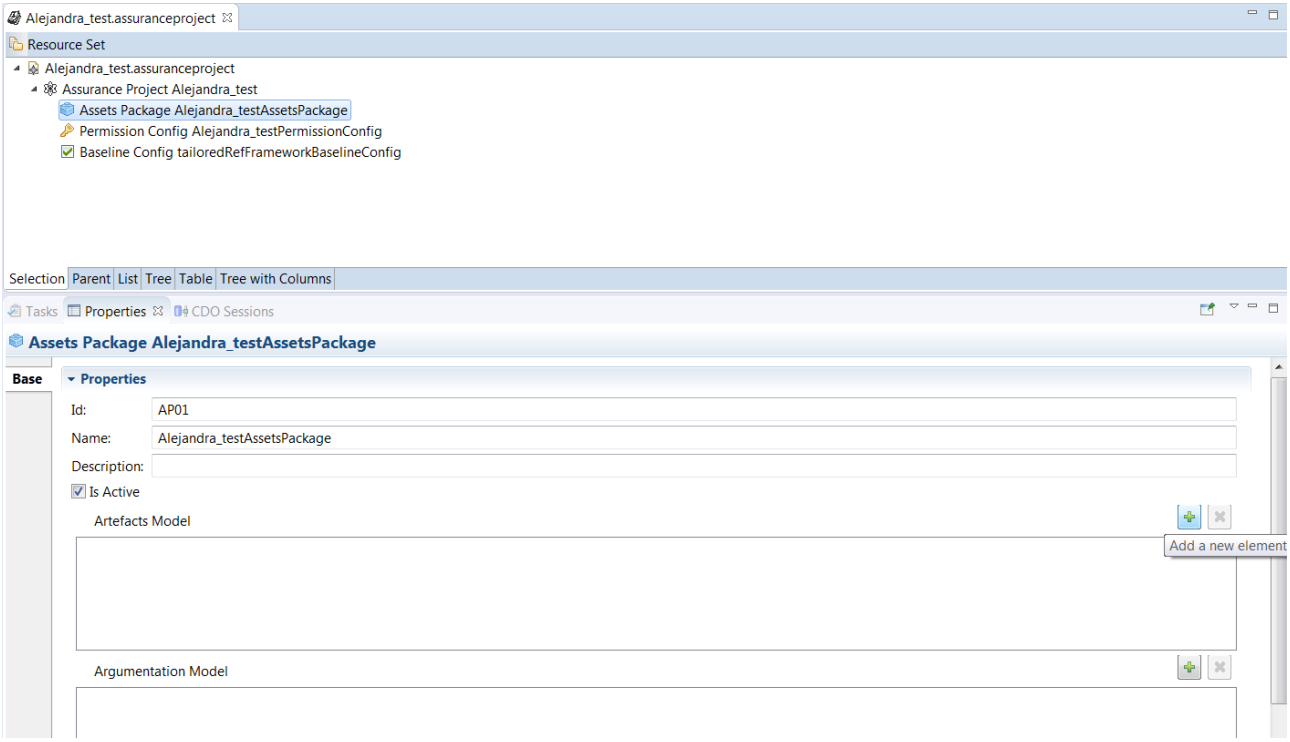


Figure 178 - Linking models to the Assurance Project's Assets Package

Press the plus symbol (+) in the Artefacts model section and a pop up window will appear, letting the user select in the All Resources Tab the evidences model previously created.

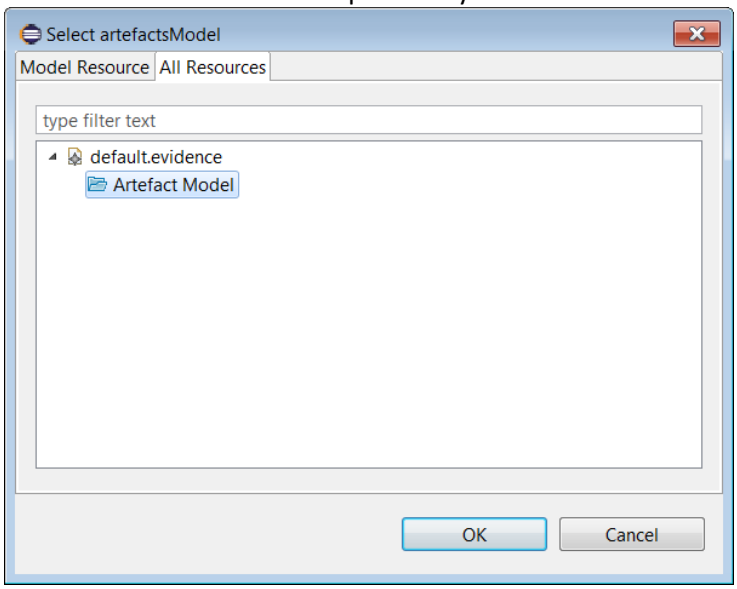


Figure 179- Selecting the evidence model to be included in the assets package

8.4 Patterns

8.4.1 Creating a new Pattern Diagram

According to the GSN standard [4] argument patterns are generic arguments which can be useful for reusable reasoning, akin to software development patterns. GSN create an extension for these abstractions, and so the tooling also support this extension.

To create a new Argumentation Pattern diagram, follow the procedure of "Creating a New Diagram". The only difference with other argumentation diagrams is that Patterns need to be stored on the places designed by the preferences.

8.4.2 Editing a Diagram Using a Pattern or a Module

In order to display the Template View, follow the next steps: Go to window→ Show View →Other
On the Menu expand the AMASS category and select the Templates view.
Double click on a folder will expand the folder. Double click on an Argumentation Diagram file (.arg_diagram) will open the diagram in the editing window.

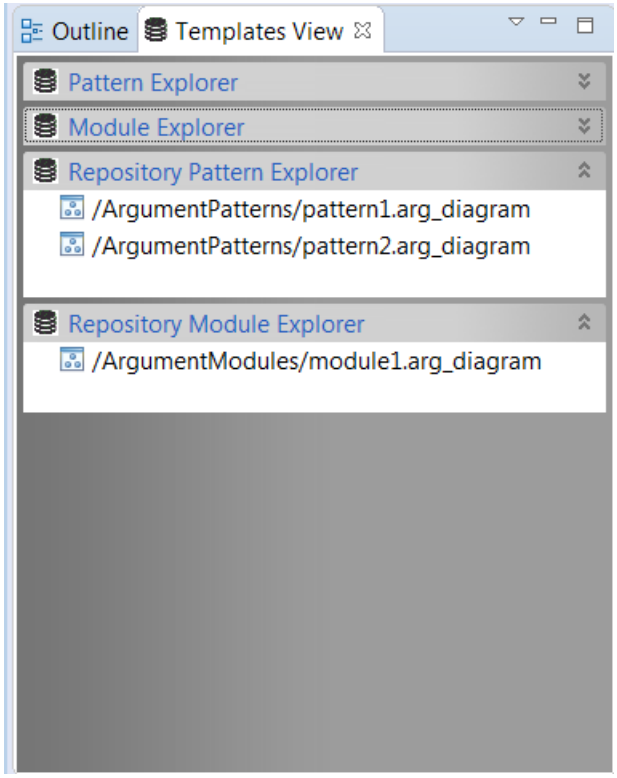


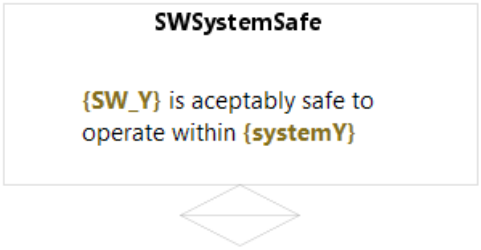
Figure 180 - Argumentation Templates View

Modules and Patterns are stored on places determines by the preferences (see previous section)
The Patterns and Modules should be stored as ready to be reused. From the Template view, the drag and drop operation is activated in order to reuse the pattern or module on the actual argumentation.

8.4.2.1 Editing a Pattern Diagram

Proceed as explained in the "Editing a Diagram" section.
Remember only Pattern Diagrams support structural and entity abstraction. Structural abstractions like multiplicity or optionality are available through "Multiextension" and "Cardinality" properties of relationships. In addition, structural options are addressed by a "Choice"[6] relationship.
Entity abstraction can be accomplished by "To Be Instantiated" and "To Be Developed" properties.
While editing a claim one of the properties than can be change is "To be instantiated", when its value it "true" then the node will change its graphics

[6] At creation time the "Choice" link must be placed inside its source node



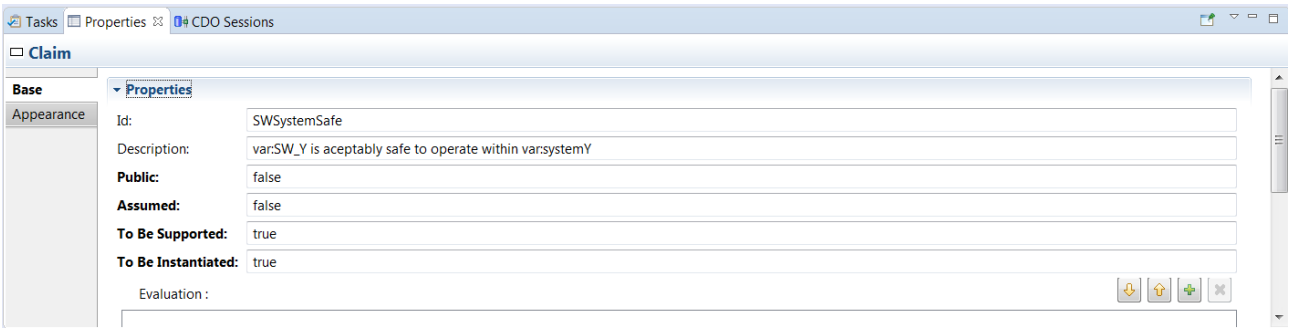


Figure 181 - Claim properties (To Be Instantiated)

The "AssertedInference" has a property called "multiextension" with three different values:

- Normal: is behaves as a regular connection. It has the "supported by" graphical notation
- Optional: it indicates that this connection is optional or alternative connections between the nodes
- Multi: it indicates the generalised n-ary relationships between the nodes. When this option is selected, the attribute "cardinality" should also be modify indicating the "n" value

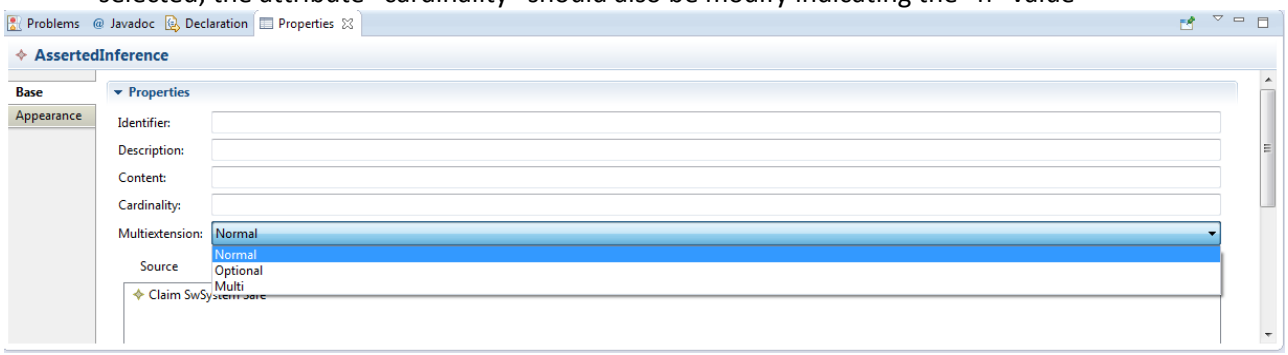


Figure 182 - Claim properties (Multiextension)

The "choice" relationship is specific for patterns edition. For editing, select the Choice at the Palette and then placed it inside its associated top Claim. The graphic for the choice relationship will appear affixed to it. Then add as many asserted inferences as number of possible choices.

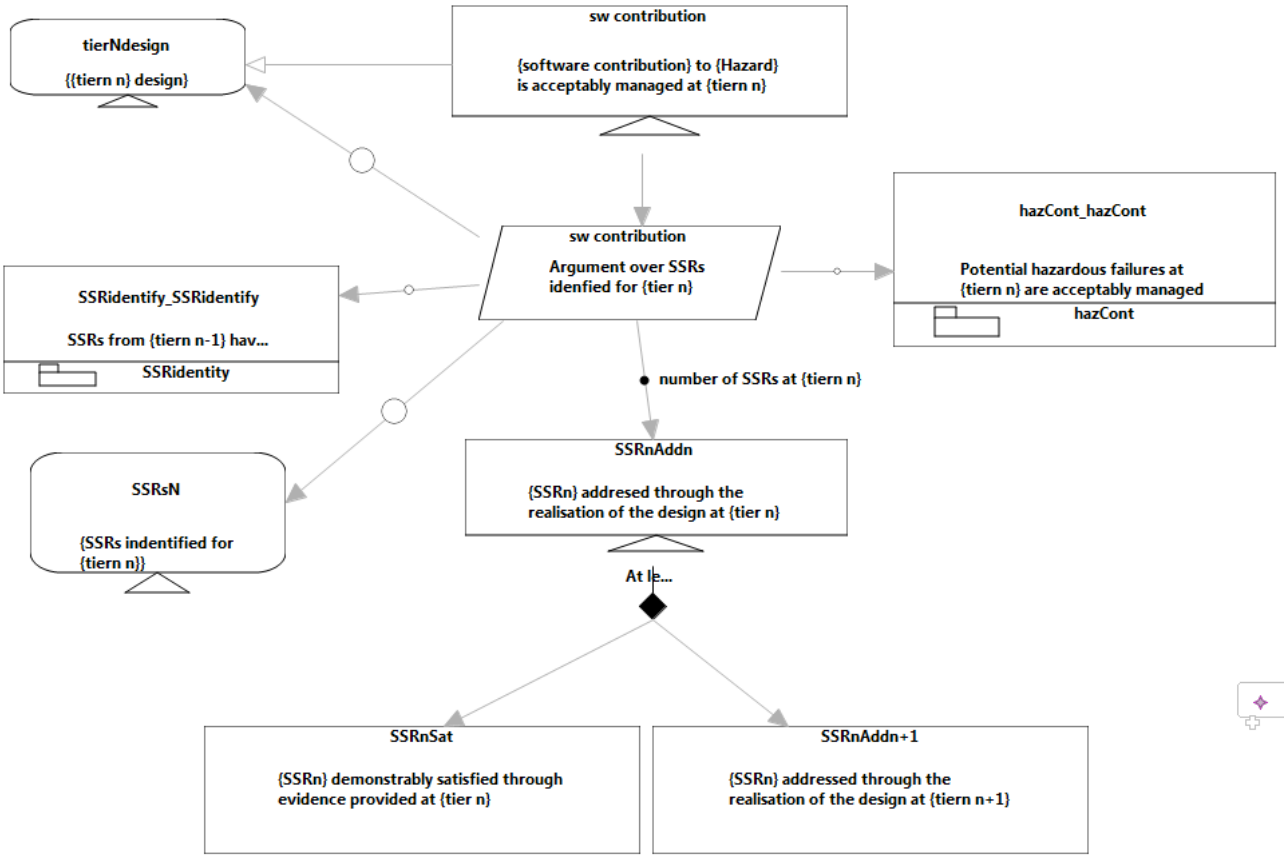


Figure 183 - Example of the software contribution safety argument pattern [5]

8.4.2.2 Adding Elements from Patterns to a Diagram (instantiating a Pattern)

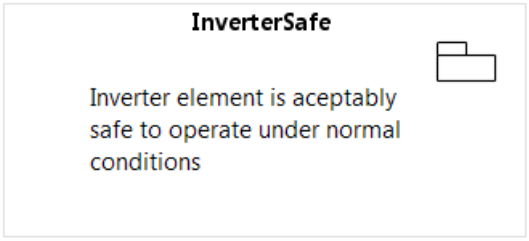
An Argument Pattern can be instantiated (thus all its content copied) into the diagram under edition.
To proceed, drag and drop a Pattern Diagram file into the diagram under edition. You need to open the argumentation diagram uncompleted that needs to include the pattern. Go to the templates view, select the pattern you are interested in, once selected you can drag and drop it into the editing part of the diagram. Once you drop you will see the new elements that have been copied into your argumentation diagram.
Once the drop is done, the “Arrange Selection” feature can be used to move to all the nodes and links. This feature can be found on the top menus as a button.

8.4.2.3 Creating a New Module Diagram

To create a new argumentation Pattern diagram, follow the procedure of “Creating a New Diagram” but generating the new diagram into the Modules directories. The only difference with other argumentation diagrams is that Modules need to be stored on the places designed by the preferences. By default preferences point to a project called “Modules” on the workspace.

8.4.2.4 Editing a Module Diagram

Proceed as explained in the “Editing a Diagram” section.
Remember Modules Diagrams (“Argumentation”) allows representing interrelated modules of argumentations. An “Argument Element Citation” repeats an element presented in another argumentation module which is used to support the argument in the local module. “Public” property indicates that an element is visible to other modules where it can be referenced. While an “Agreement” element represents the agreed relationship between modules.
To indicate that a claim is Public, just indicate on the Claim properties view, the attribute “Public” be “true”. The Public activation is also noticed on the graphic notation.



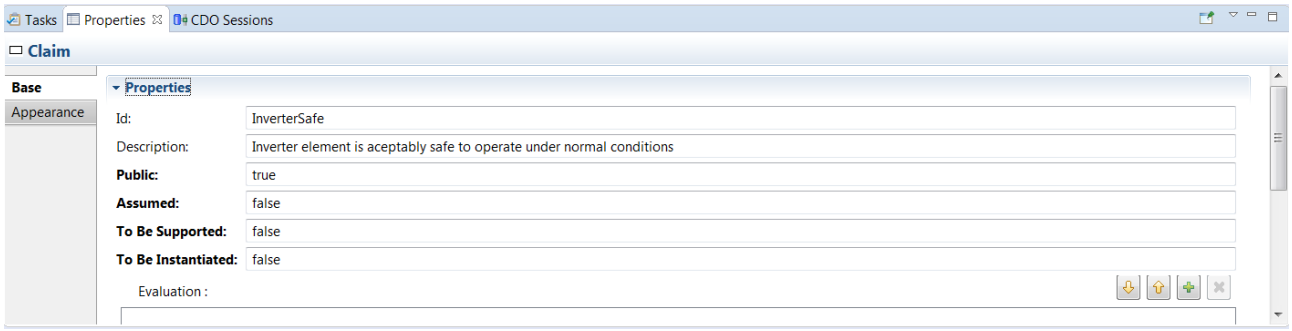
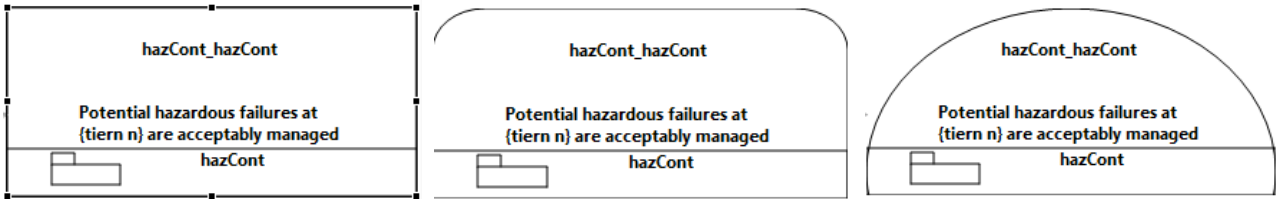


Figure 184 - Claim properties (declared as Public)

Argument Element Citations can represent different concepts; they can reference a claim, a context or a solution. On the properties view, an attribute called “cited type” should be informed and consequently the graphic notation might change.



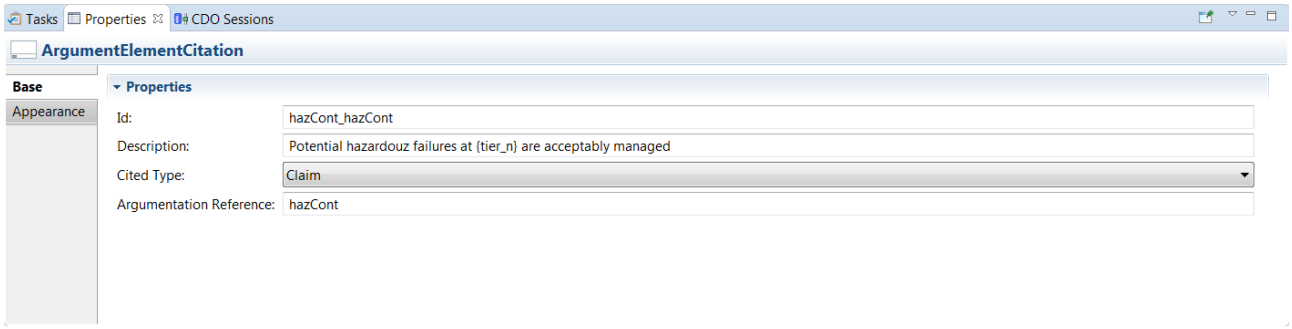


Figure 185 - ArgumentElementCitation properties (reference to a claim)

The “Argumentation Reference” property indicates the reference to the module in which this citation element is described.

8.4.2.5 Adding Elements from Modules to a Diagram (instantiating a Module)

An Argumentation Module can be instantiated (thus all its content copied) into the diagram under edition.
To proceed, drag and drop a Module Diagram file into the diagram under edition.
Once is its instantiated an “argumentation” graphic notation will appear on your diagram.

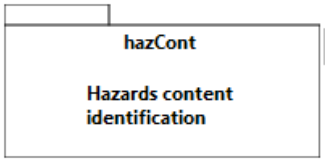


Figure 186 - Argumentation Module

By double clicking on this Argumentation another diagram will appear with the argumentation context of this module. The url property indicates the location of this diagram.

8.4.3 Vocabulary

The safety argumentation supports the usage of terms and term categories which have been defined in a vocabulary. The meaning of some terms and categories is specific to legislative regulations, standards and or the project they are used in. Vocabularies capture the meaning of such terms and categories by providing a definition.

8.4.3.1 Defining Vocabularies

The simplest way to define a vocabulary is to create one in the project where it going to be used. They can be stored either locally in a file or inside a remote repository.

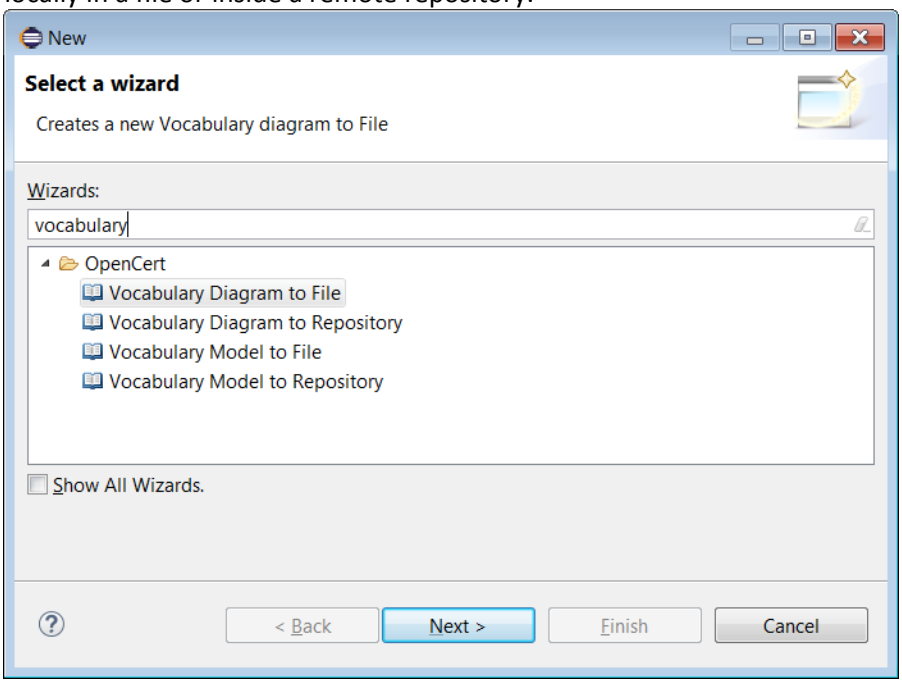


Figure 187 - New Vocabulary

The vocabulary can be visualized in a diagram to which shows how terms are related to each other. The diagram is a visualization of the vocabulary model but can also be used to edit the model.

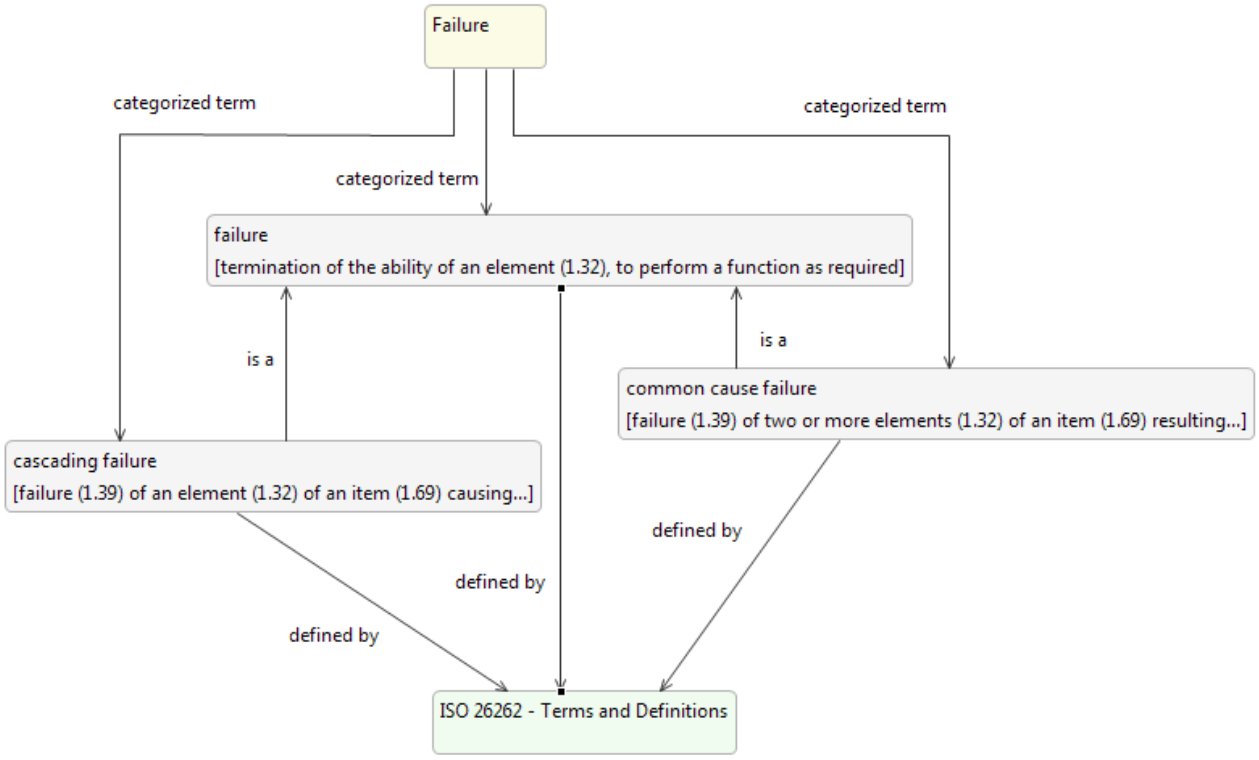


Figure 188 - Example Vocabulary Diagram

Writing vocabularies is a time-consuming process. To save some or all of that work, vocabulary data can be imported from files in a custom XML format. In order to talk about elements of the CCL model in the argumentation, it is not necessary to duplicate the model elements as vocabulary terms. Instead, they can be imported into a vocabulary.

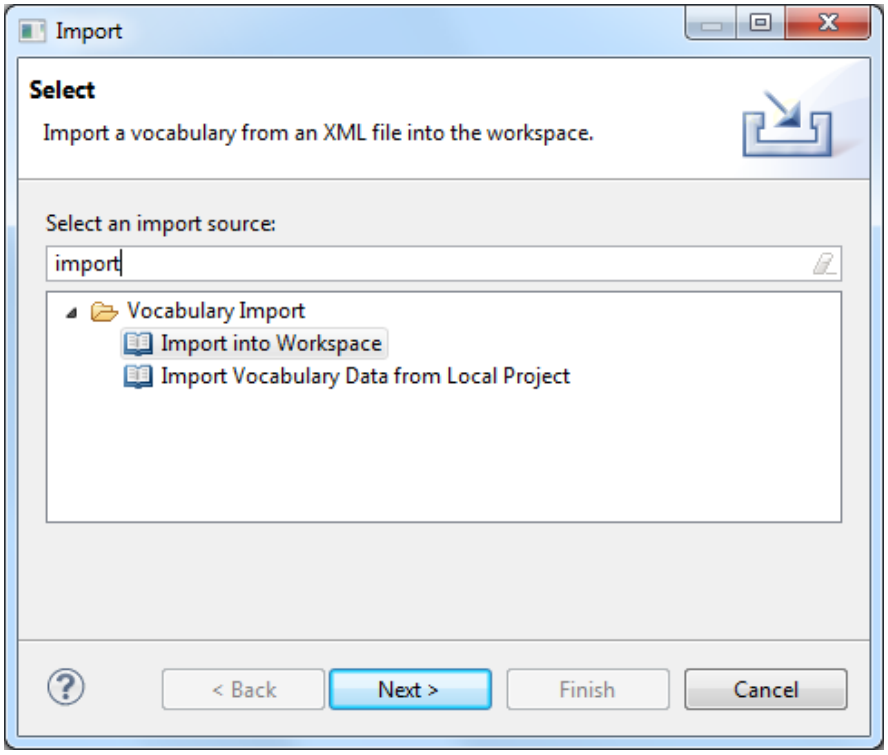


Figure 189 - Vocabulary Import

8.4.3.2 Using Vocabularies in the Argument Editor

Terms and term categories from the vocabulary can explicitly be used in the argumentation. In order to do so, some mark-up is required. The mark-up is visible while editing text, otherwise it will not be.

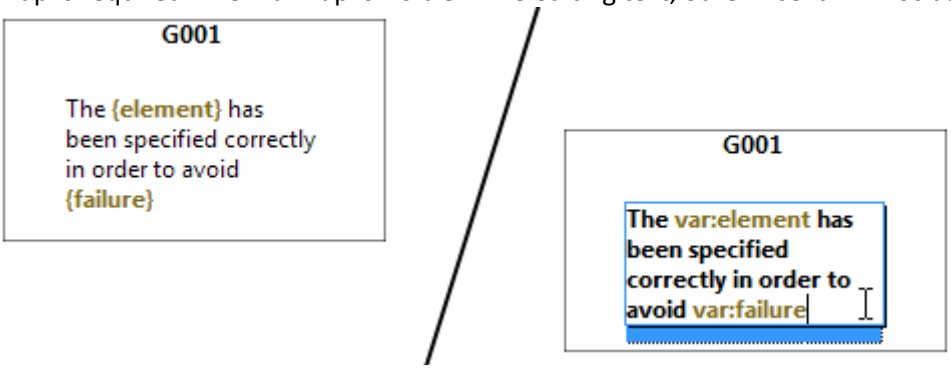


Figure 190 - Mark-up Rendering

One of the following mark-up variants can be used:

- voc:term
- voc:"term" – Usually for terms with spaces or when the sentence ends after the term.
- voc:term|terms – Provide a natural language expression to be rendered instead of the term name, e.g. the plural form of the term. Add quotes if the term or the expression contains spaces.

Categories use the "var" prefix instead of "voc".

Syntax highlighting and tooltips for vocabulary elements are available inside the argumentation editor.

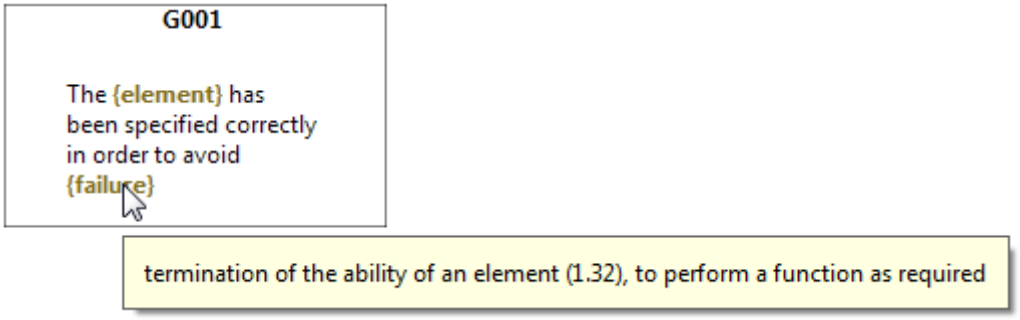


Figure 191 - Tooltip

While editing text, pressing Ctrl + Space will display a list of available vocabulary items. The list gets smaller when the user types the starting letters of the item he is looking for. Pressing enter inserts the item at the cursor.

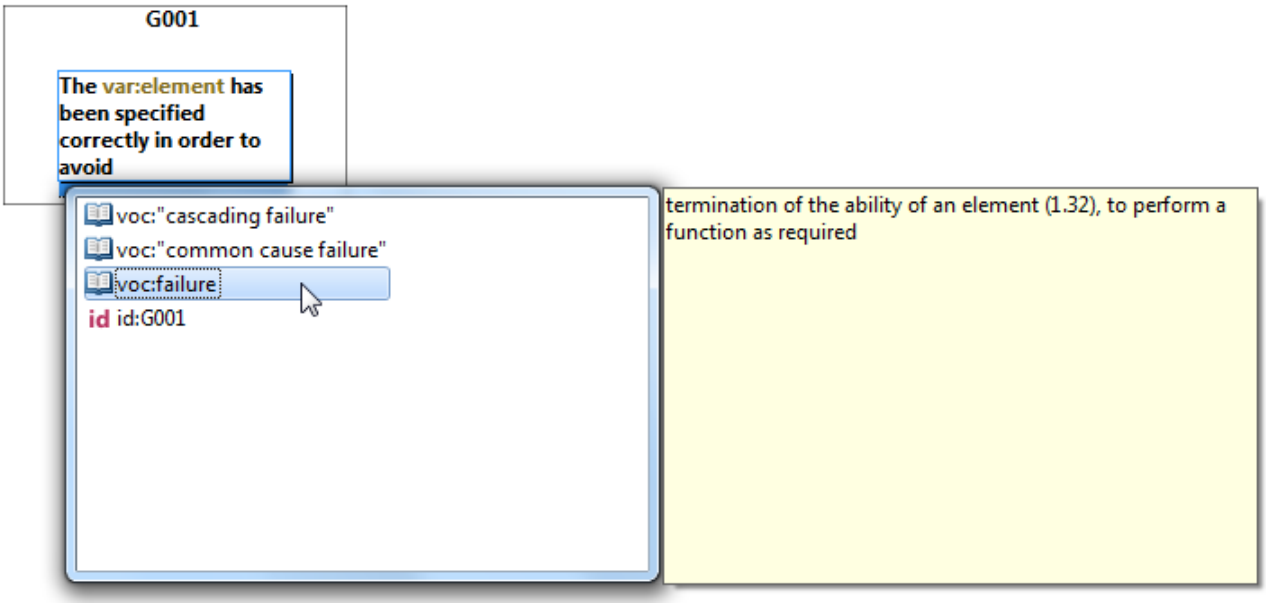


Figure 192 - Term Suggestions

8.5 Generating Argument Fragments

The Argument Generator generates a set of argument-fragments from the selected CHESS model and stores them in the corresponding destination assurance case in the CDO repository stated in the OpenCert preferences. The Argument Generator assumes that the CHESS model is enriched with contracts and that contract refinement has been performed such that contract status is updated to indicate if the contract is validated in the given context or not. Moreover, Argument Generator assumes that the analyzed model and the refinement check results are stored in the refinement analysis context. Furthermore, if there are any connections of CHESS model elements with OpenCert assurance information, as described in Section 7.19, the relevant information will be included in the argument-fragments based on the relevance of the CHESS elements for assurance. The attached screenshots (Figure 193 - Figure 197) illustrate how the generation of argument-fragments can be performed. First, you need to select the OCRA refinement analysis context. Then, you need to indicate to which assurance case on the corresponding CDO server should the argument-fragments be saved. The argument-generation is performed for each component and for each validated contract. The set of argument-fragments for each component can be viewed in the selected assurance case.

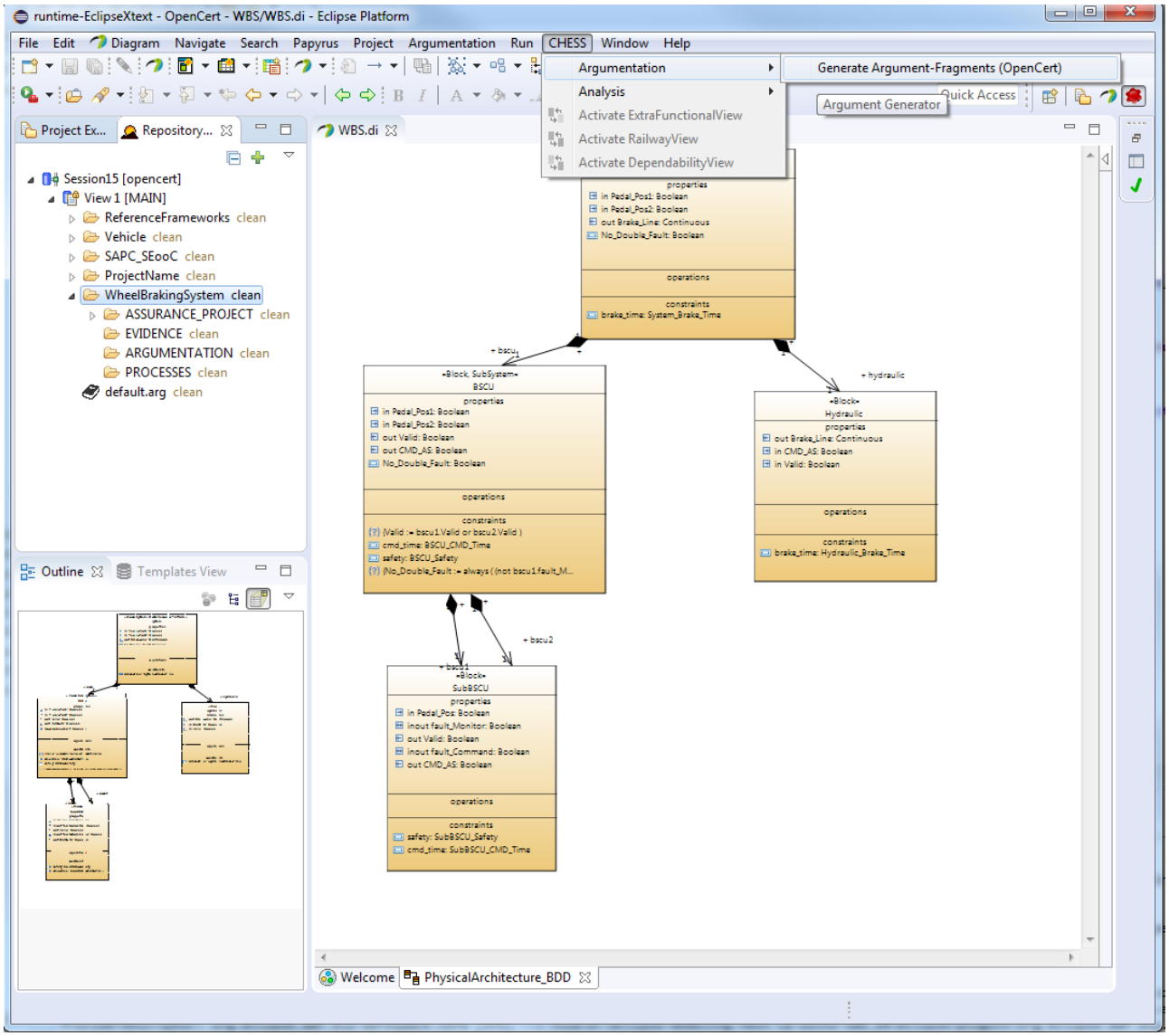


Figure 193 - Initiating the argument-fragment generation (Step 1)

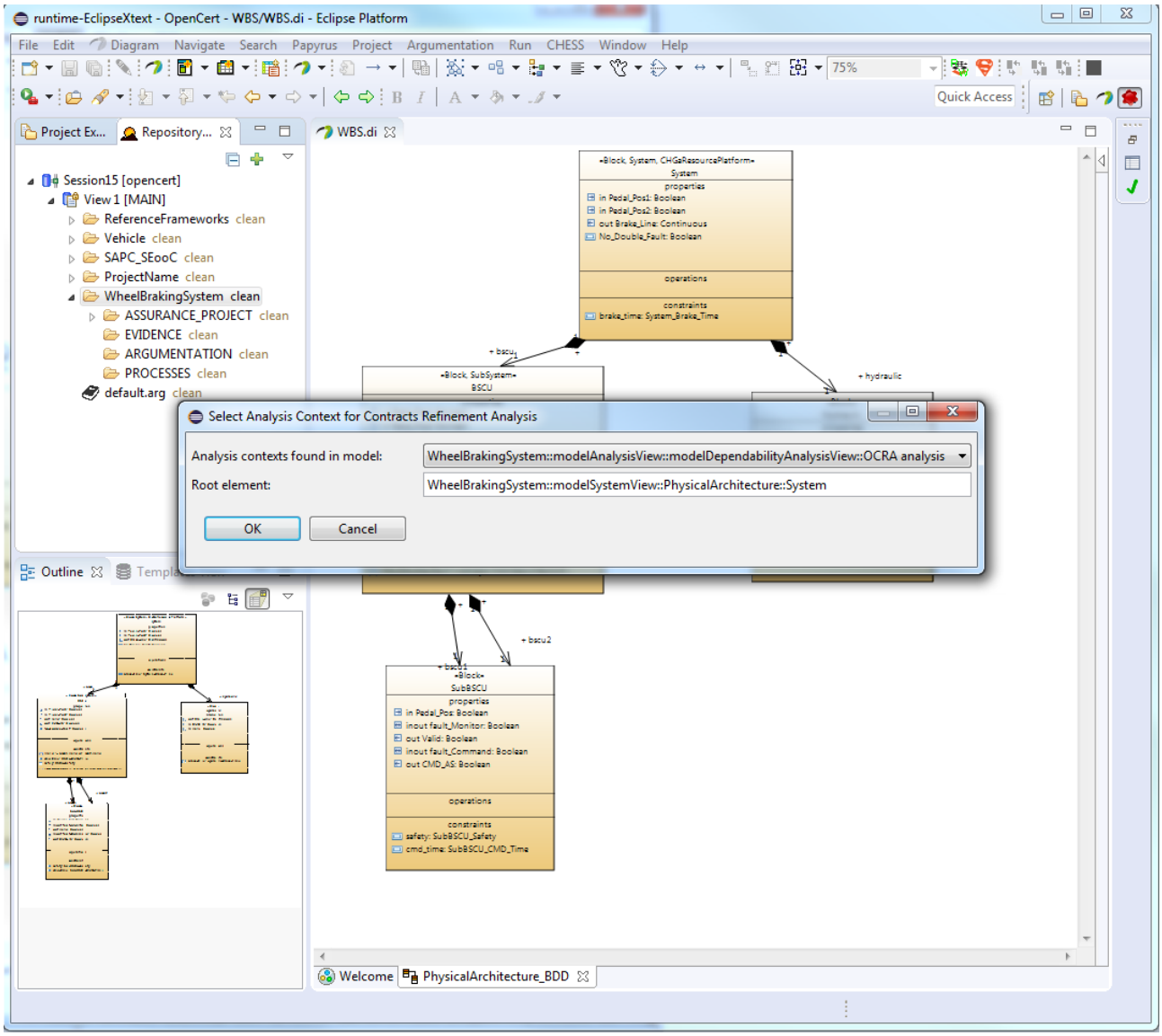


Figure 194 - Selecting the source analysis context (Step 2)

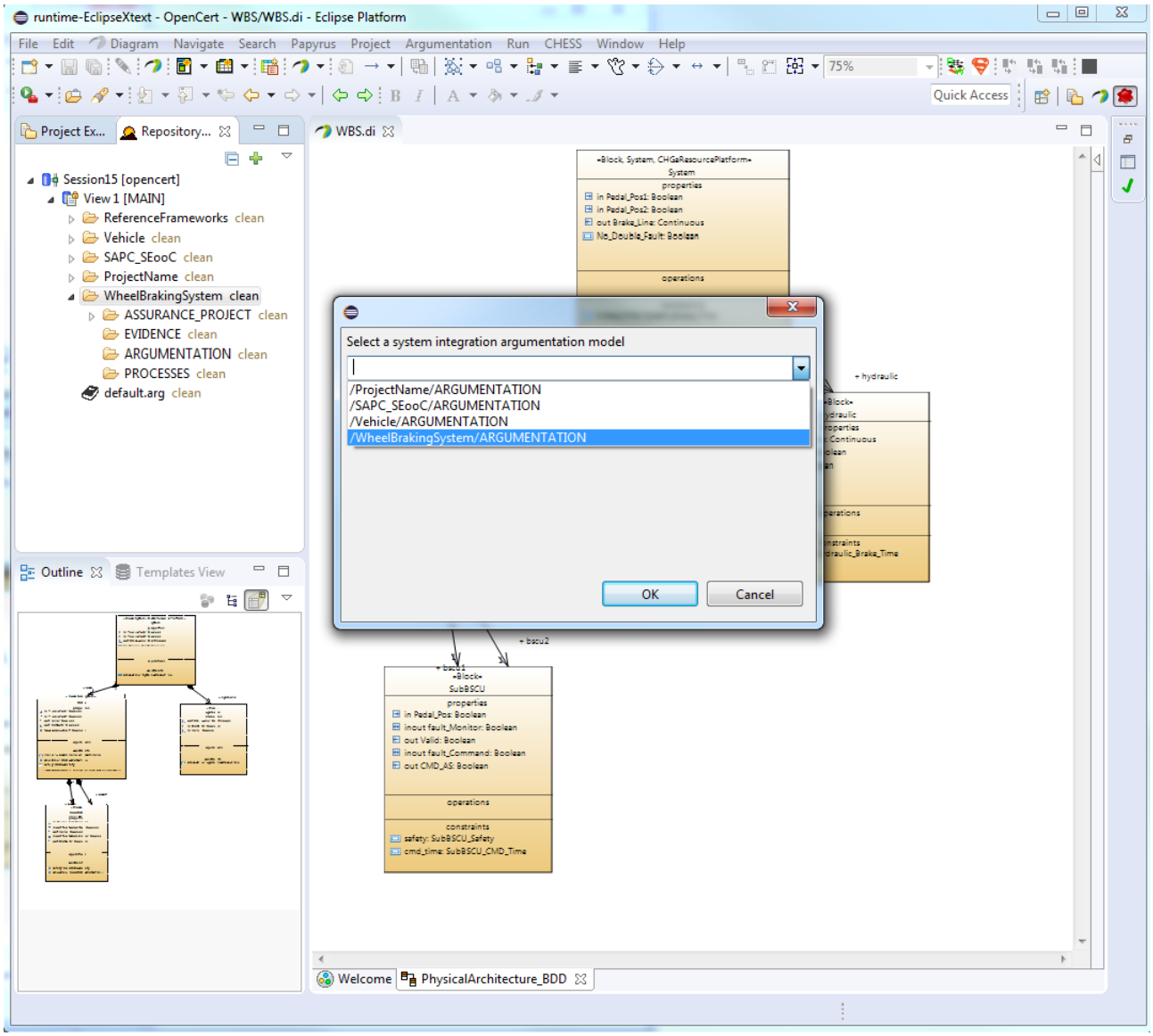


Figure 195 - Selecting the destination assurance case folder on the CDO repository (Step 3)

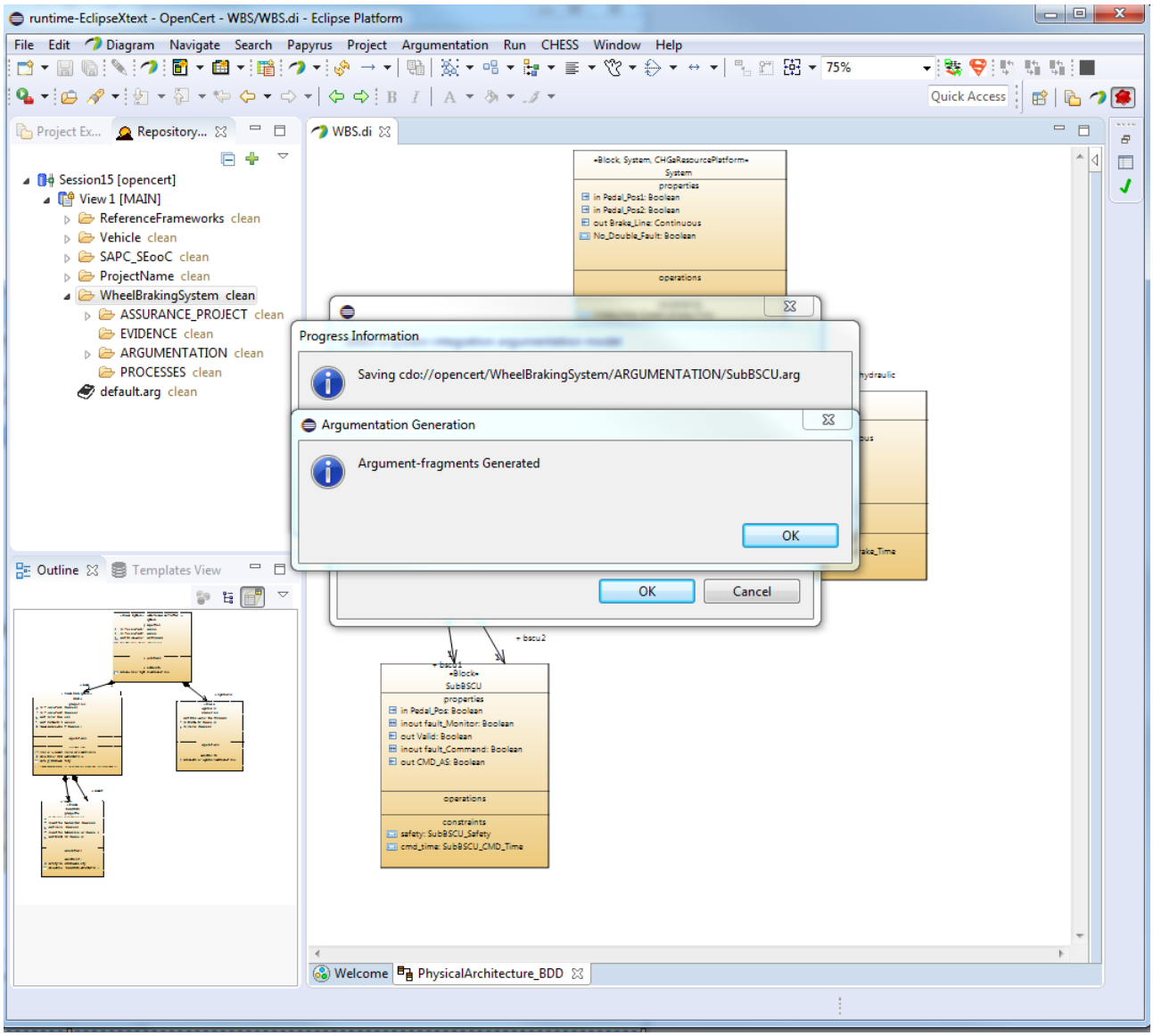


Figure 196 - Generation successfully completed with argument-fragments for each block

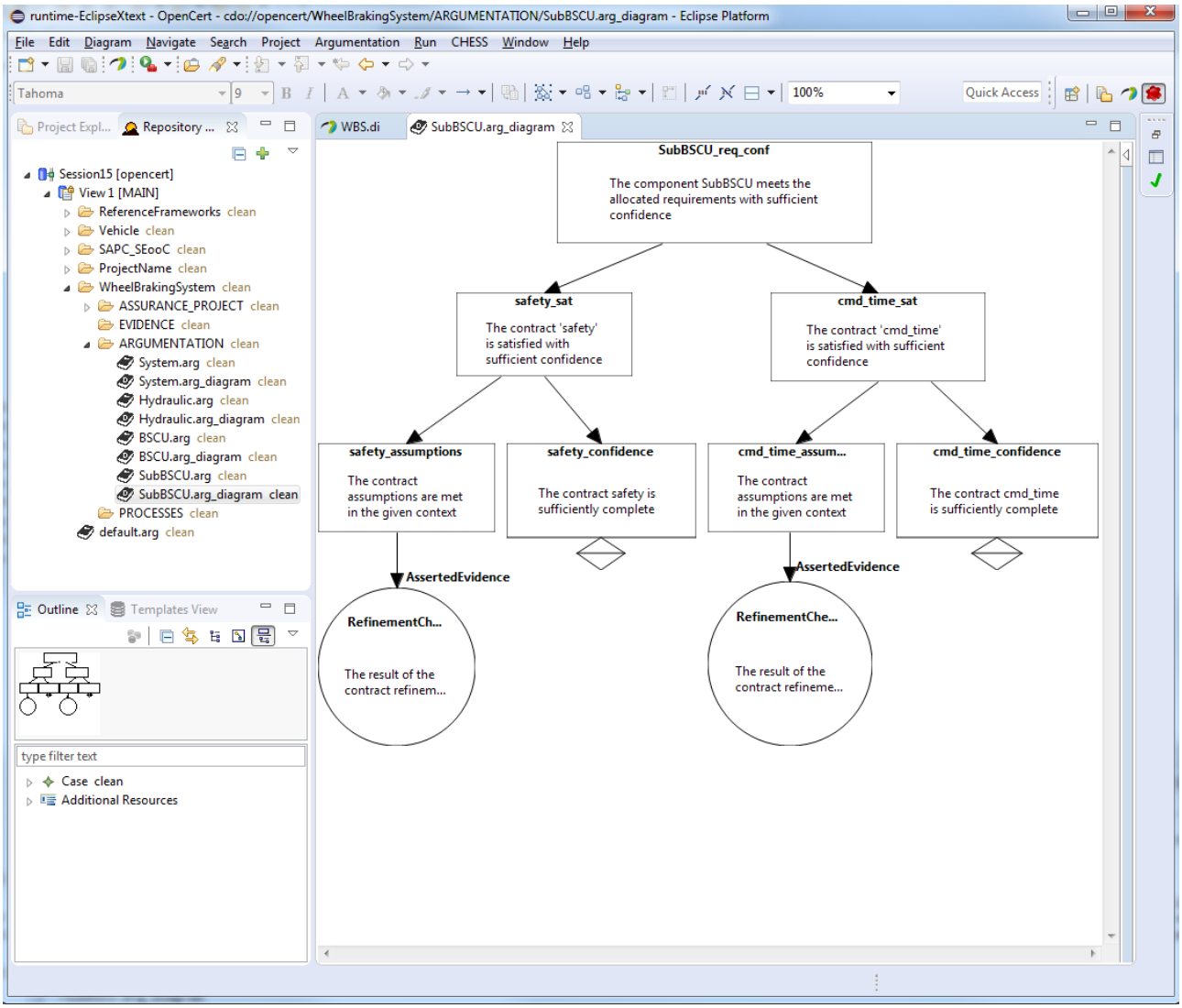


Figure 197 - An example of the generated argument-fragment

8.6 Printing

The following operations regarding printing are available under the “File” menu.
1. “Print Preview” item: Print preview
2. “Print...” item: Print
3. “Page Setup...” item: Print settings

8.7 Export an Argument Model

In order to export an argument model stored in the repository just select under the “File” menu, the Export option. A new window will appear and the user should select under the OpenCert category the Export argument model to File option.

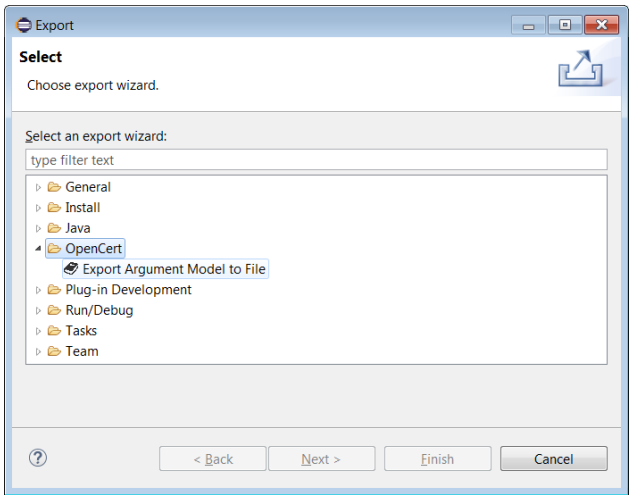


Figure 198 - Export wizard selection window

A new window appears where the user could select any argument folder stored in the repository and indicate the file destination location and name.

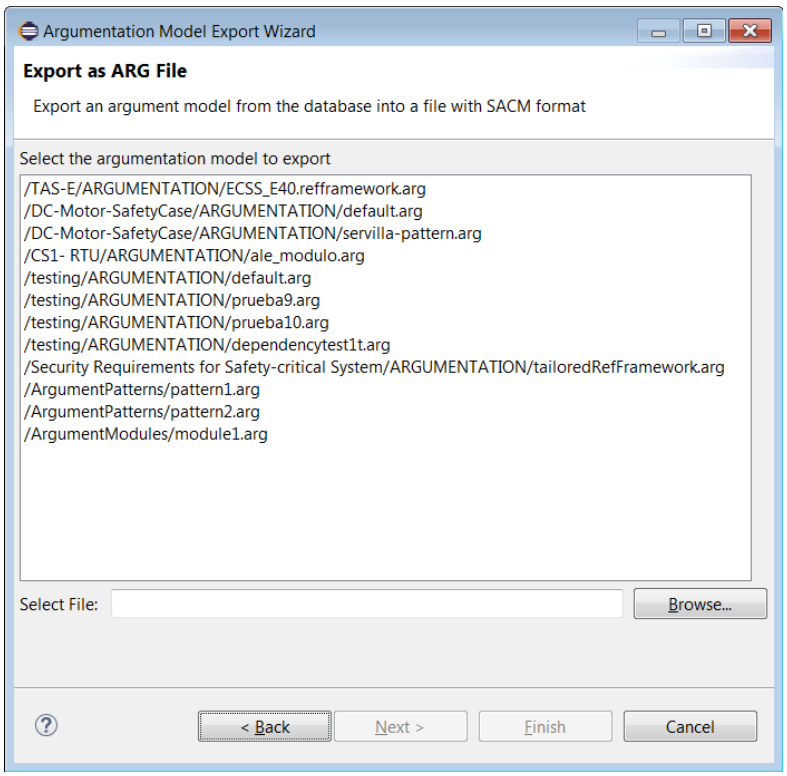


Figure 199 - Export argument model window

Note: The user can import any argument model exported with this functionality using the steps described in section 6.12.

9 Evidence Management

9.1 Define Artefact Repository Preferences

The first step before creating an Evidence Model is to indicate the SVN Repository configuration information to store the artefact files using the menu Windows→ Preferences

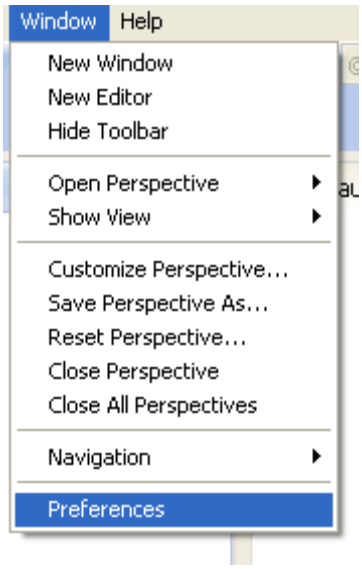


Figure 200 - Preference menu

Then select the Opencert→ Artefact Repository Preferences category and introduce the information required. If you want to use a local directory as Artefact Repository you have to check the "Use Local Repository" or uncheck it to use a SVN Server, the path of the local folder used as Local repository, the URL of the remote SVN server and the user and password of the SVN server.

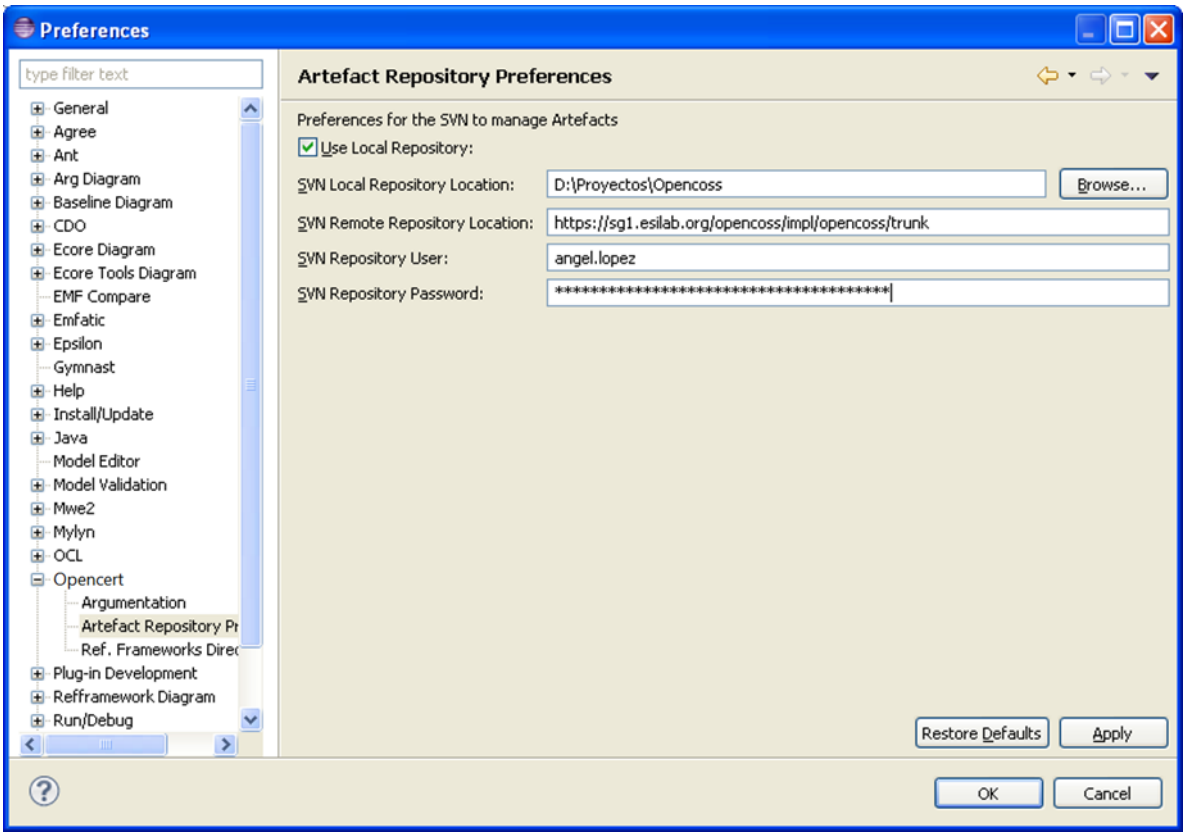


Figure 201 - Artefact Repository Preferences

The management of Evidences must be made through the creation of a new model of the type **Evidence Model**.

In order to generate a new Evidence Model, the following steps need to be done:

- First, select the entry of the menu *File -> New -> Other*:

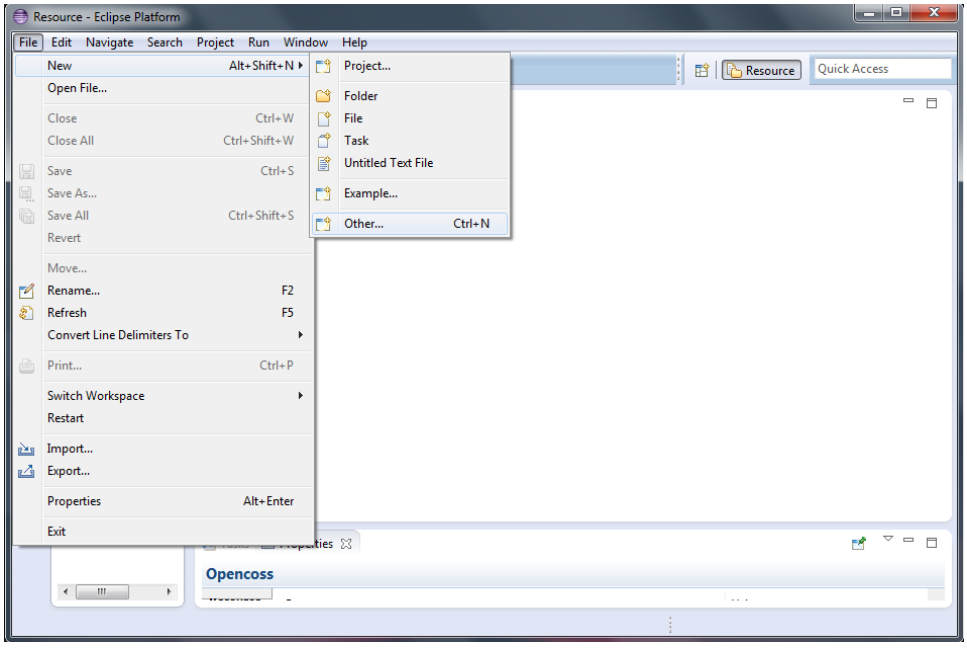


Figure 202 - New Evidence Model menu File -> New -> Other

- Inside the category of the wizard *Opencert*, select the *Evidence Model* to the Repository and press the *Next* button:

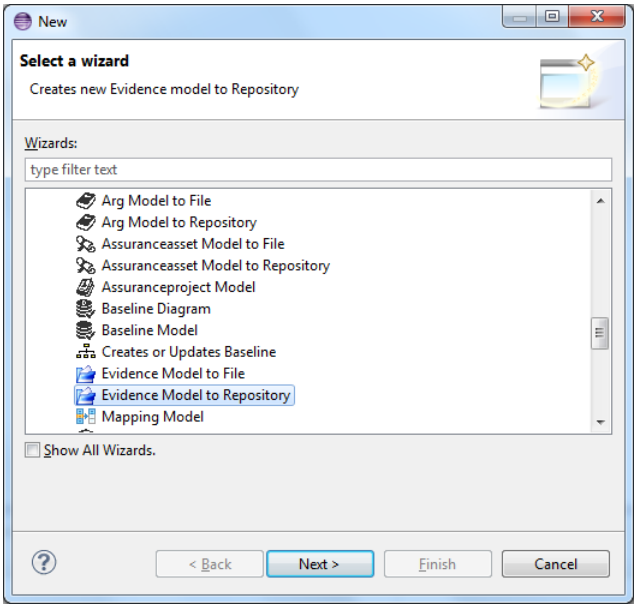


Figure 203 - New Evidence Model I

- Enter o select the parent folder, the resource name and press the Next button:

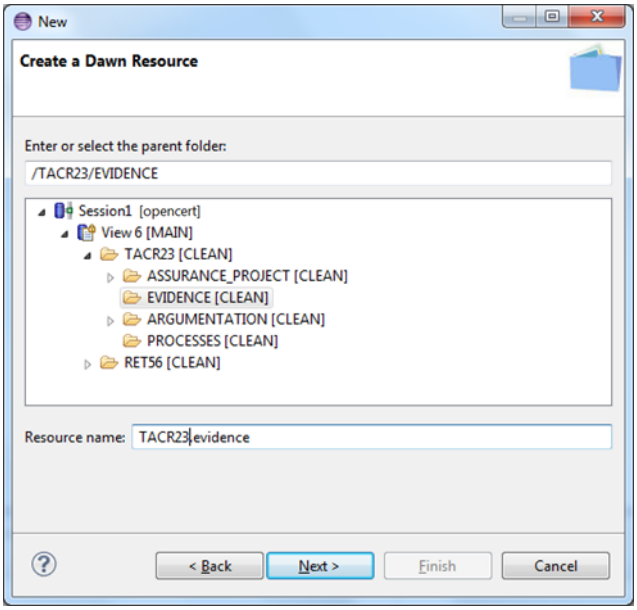


Figure 204 - New Evidence Model II

- And finally, select the “Artefact Model” object to create.

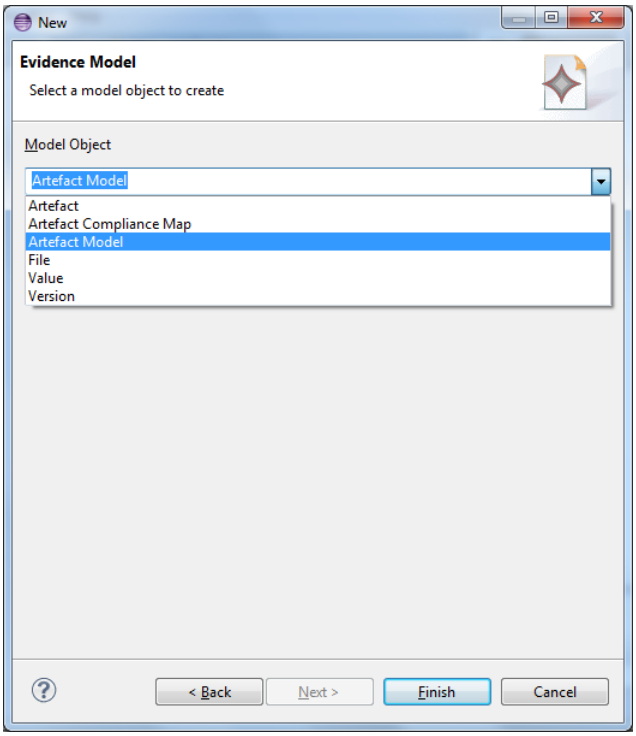


Figure 205 - New Evidence Model III

Once the Evidence Model has been created, the first item is presented to the user.

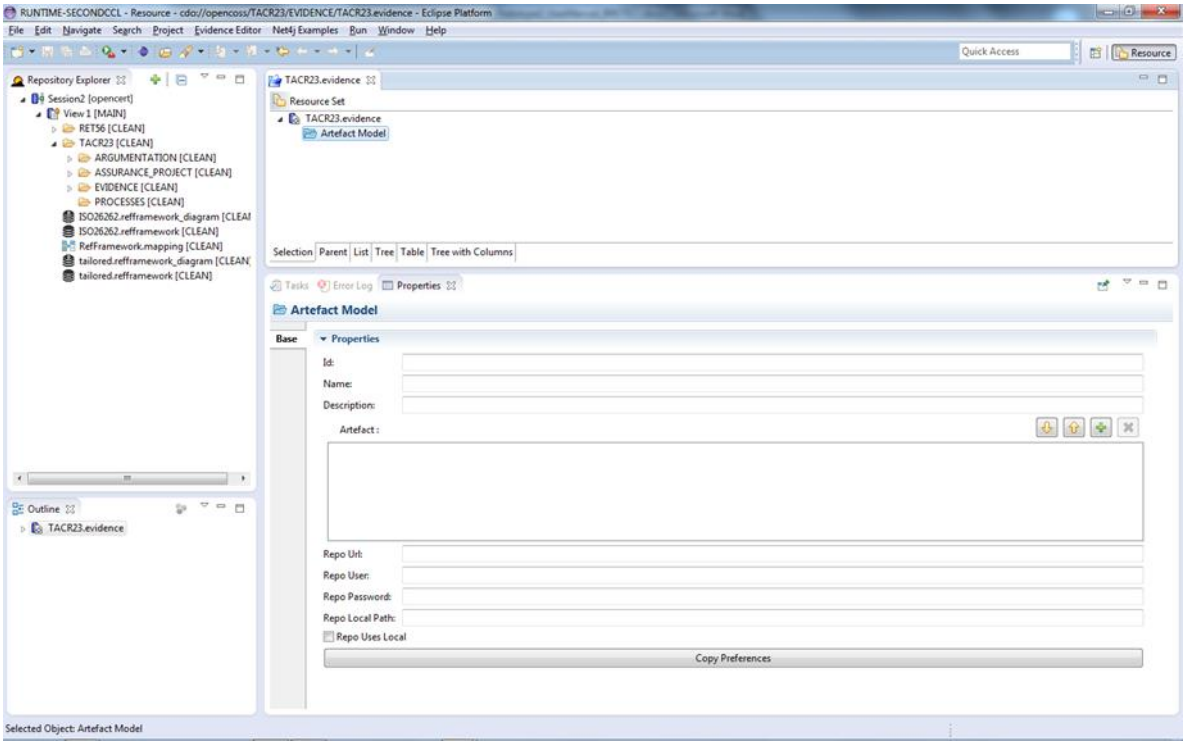


Figure 206 - Evidence Model

The “Copy Preferences” button will copy the Artefact Repository Preferences data to this model and will be saved in the model to be used to store the Artefact files of this evidence model. If this information is empty, then the data specified in the Artefact Repository Preferences will be used to store the artifact files.

9.2 Artefact Definition

9.2.1 Add an artefact definition

It is possible to add artefacts definition to an artefact model in two ways:

- One way, select the artefact model and press the button

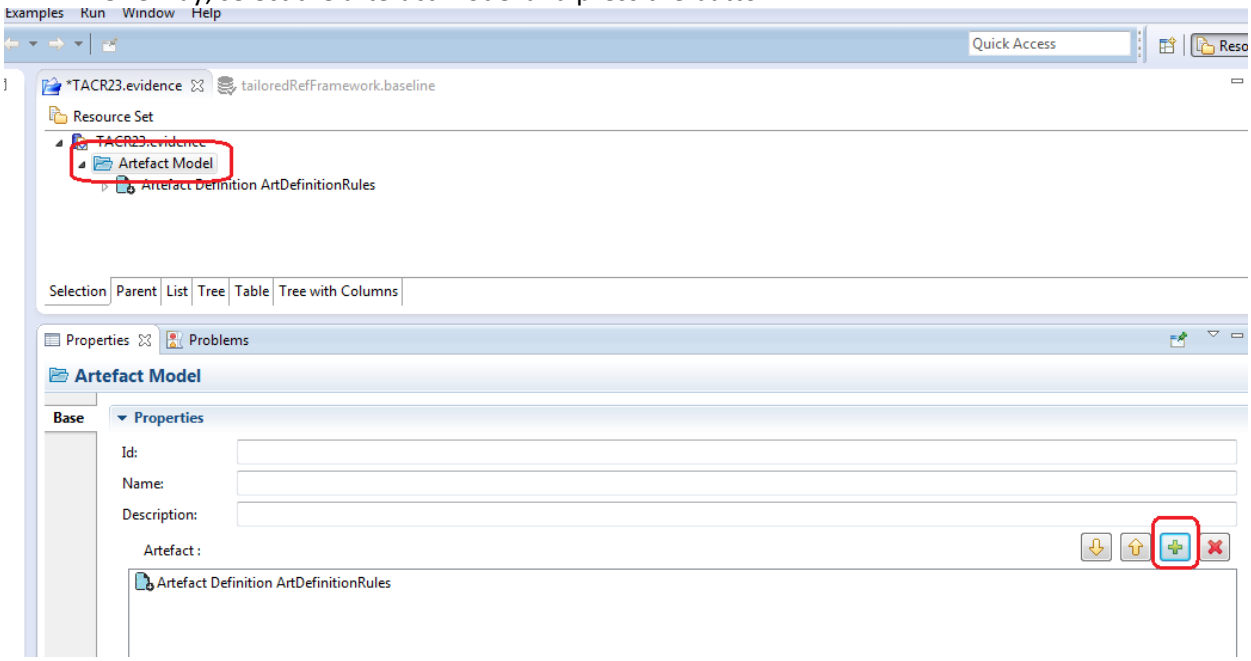


Figure 207 - Add New Artefact Definition (I)

- Another way, click on the branch Artefact Model, press the right mouse button and select the contextual menu New Child –> Artefact Definition.

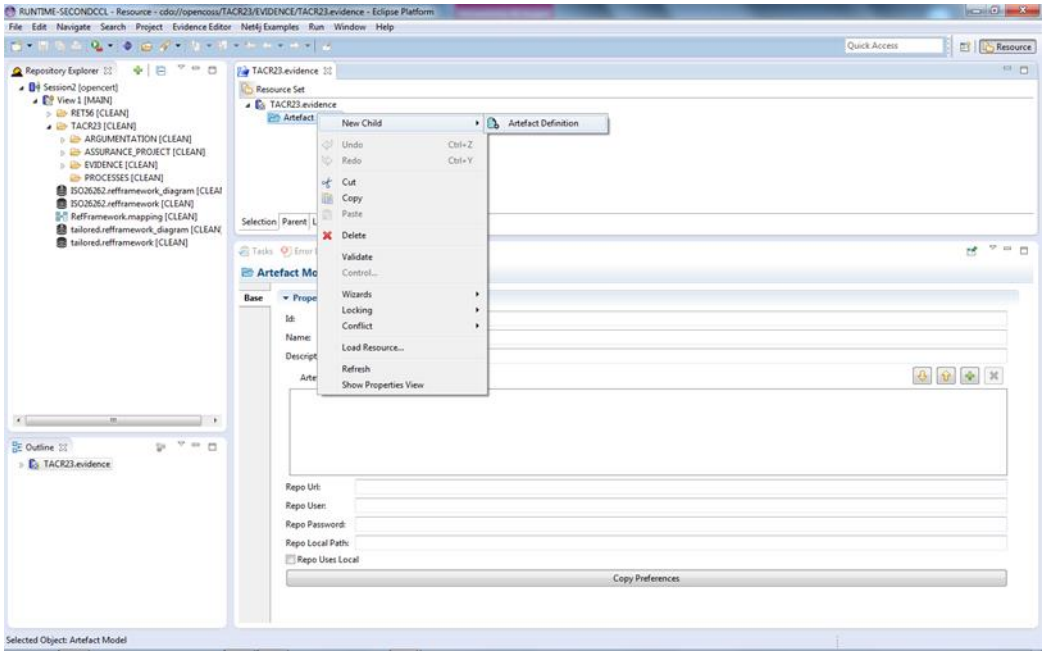


Figure 208 - Add New Artefact Definition (II)

In the properties zone, the framework presents several fields to describe the new Artefact Definition divided in tabs:

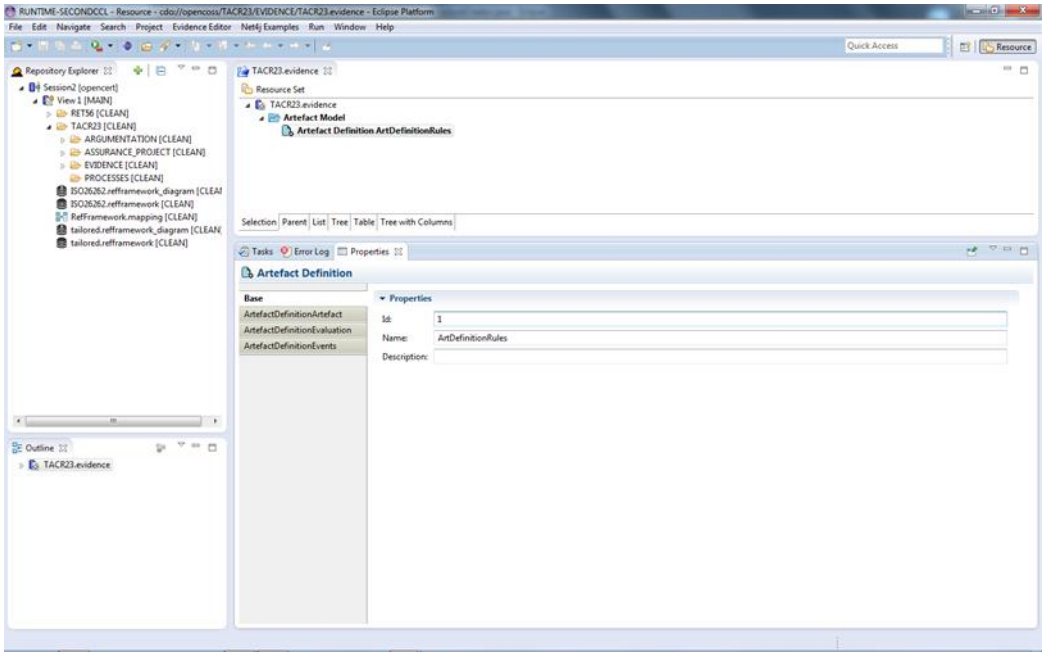


Figure 209 - Artefact Definition Description (I)

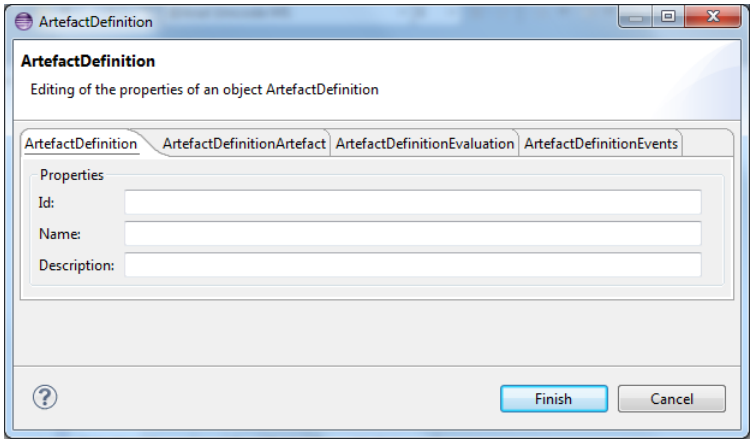


Figure 210 - Artefact Definition Description (II)

1. Artefact Definition (base)
    - Id: Artefact Definition identifier.
    - Name: Artefact Definition name.
    - Description: Artefact Definition description.
2. Artefact Definition Artefact
    - Name: Artefact name. This field is read-only.
    - Version ID: Identifier of the artefact version.
    - Date: Date of the artefact version.
    - Last Version: This field shows what artefact is the version in use.
    - File ID: Identifier of the file associated with the artefact.
    - Name: Name of the file associated with the artefact.
    - Description: Description of the file associated with the artefact.

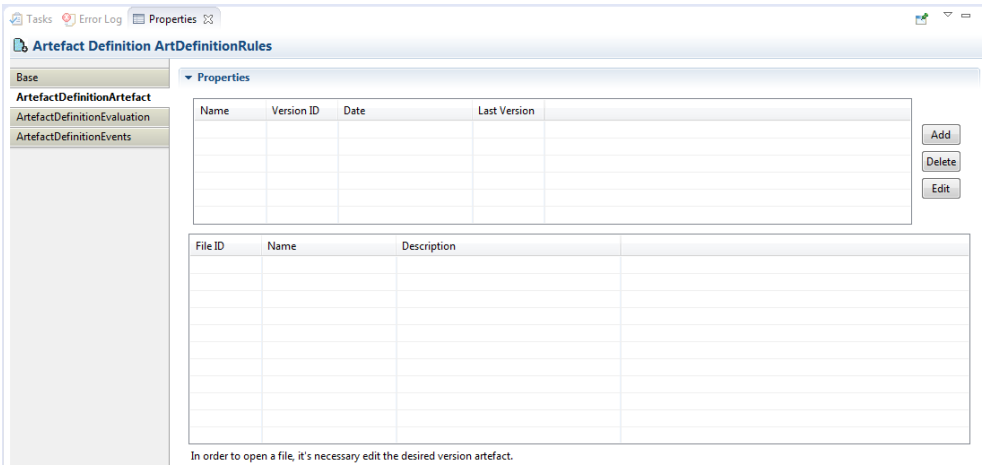


Figure 211 - Description Artefact Definition Artefact

3. Artefact Definition Evaluation
    - References to the assurance asset evaluations that specify the outcome of evaluating the artefact.

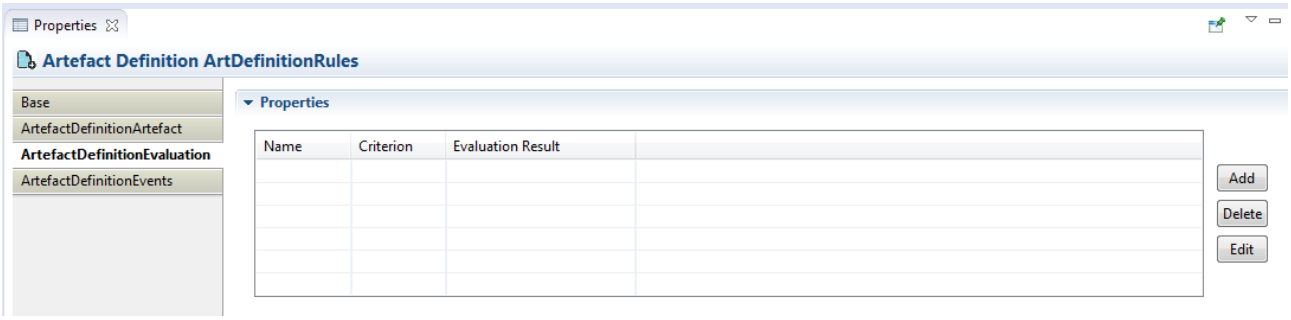


Figure 212 - Description Artefact Definition Evaluation

4. Artefact Definition Events:
    - References to the assurance asset events of which the lifecycle of the artefact consists.

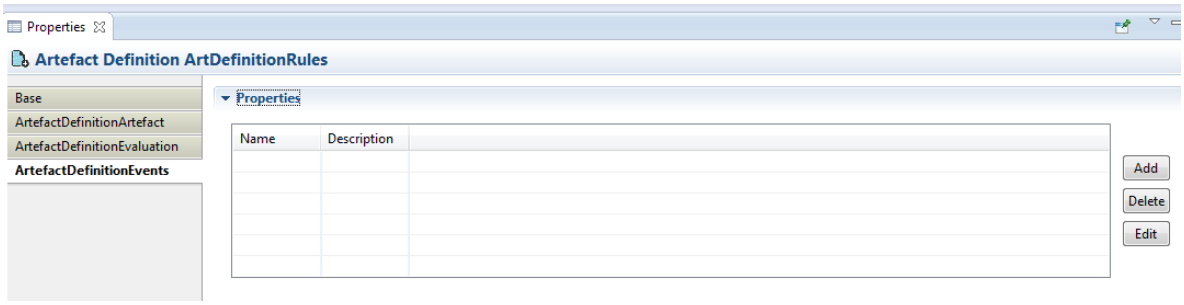


Figure 213 - Description Artefact Definition Events

9.2.2 Delete Artefact Definition.

To delete an artefact definition:

- Select the artefact definition, press the right mouse button and select the contextual menu *Delete:*

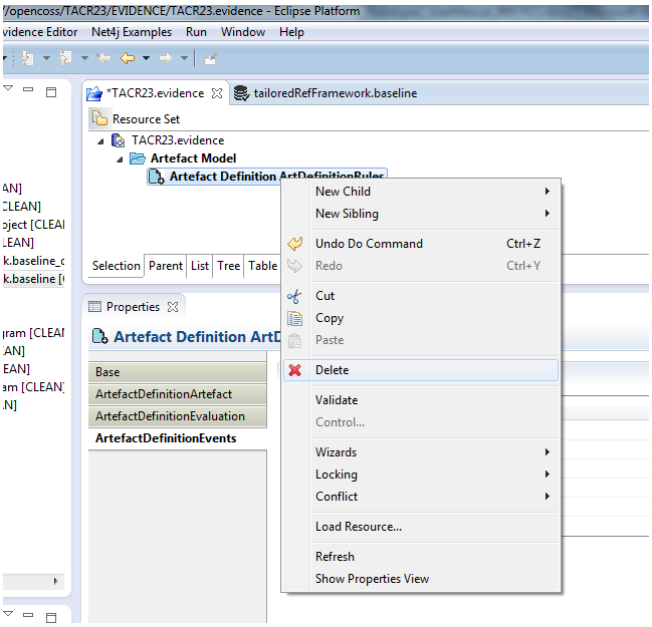


Figure 214 - Delete Artefact Definition (I)

- Or, select the artefact model, select the artefact definition model to delete and press the button *:*

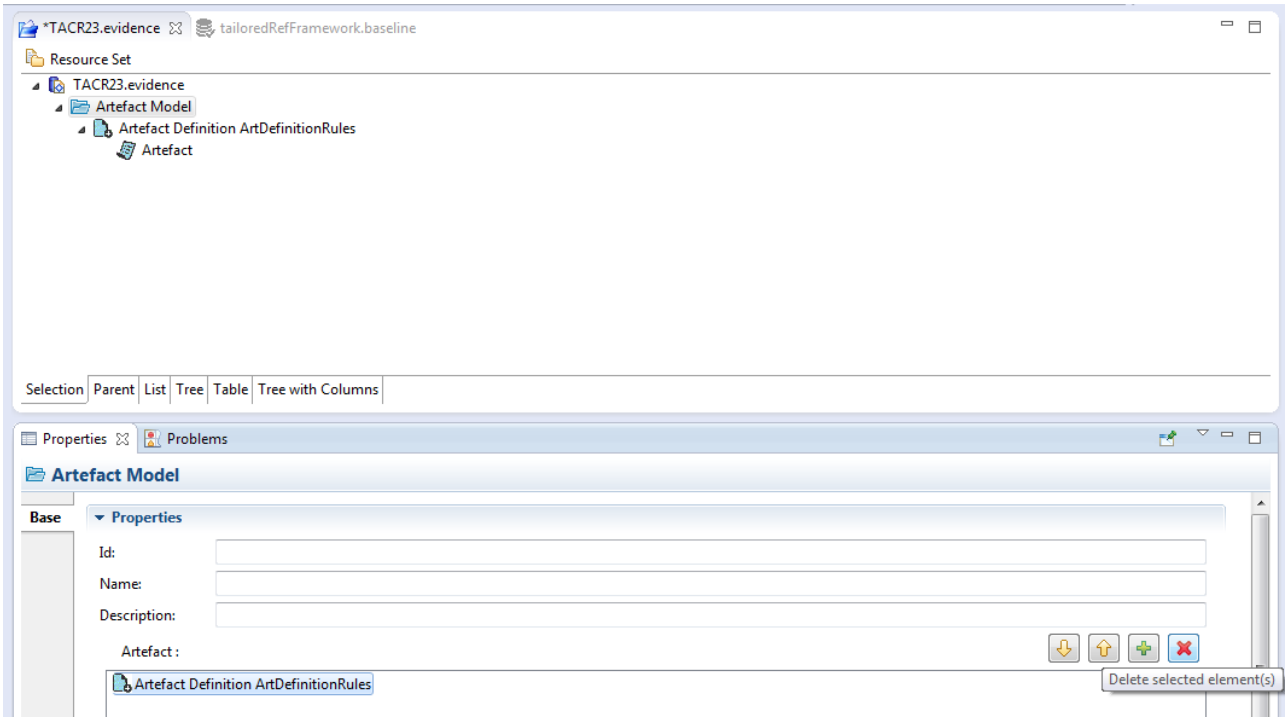


Figure 215 - Delete Artefact Definition (II)

9.3 Artefact

9.3.1 Add an artefact

It is possible to add artefacts to an artefact definition in two ways:

- Select the artefact definition, press the right button of the mouse and select the contextual menu *New Child –> Artefact*

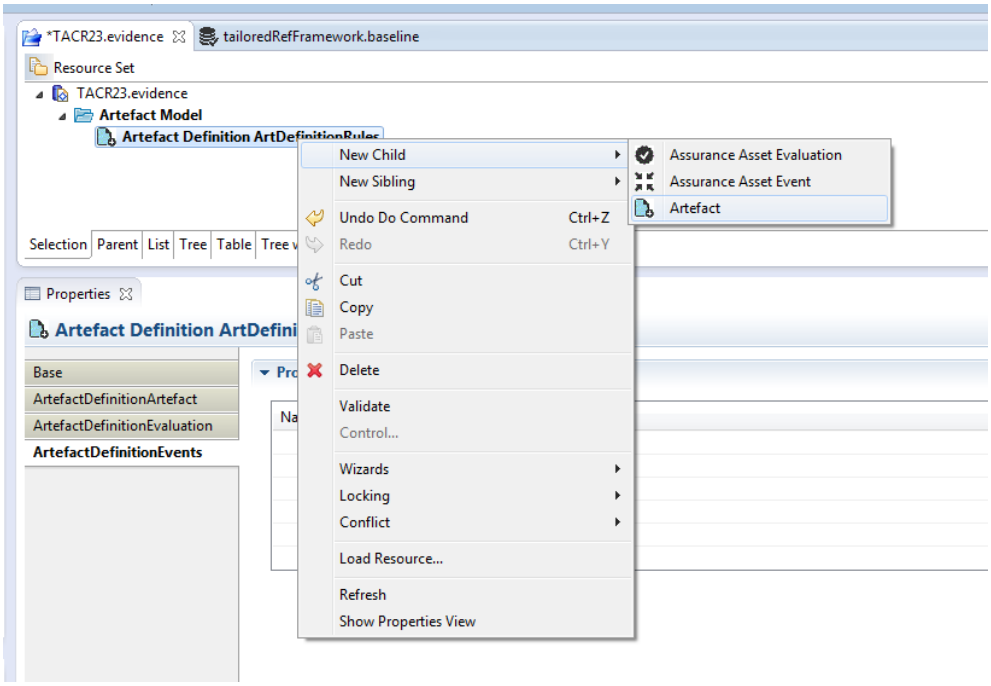


Figure 216 - Add New Artefact (I)

- Or, select the artefact definition, select the Artefact Definition Artefact tab Properties, and press the button *Add*

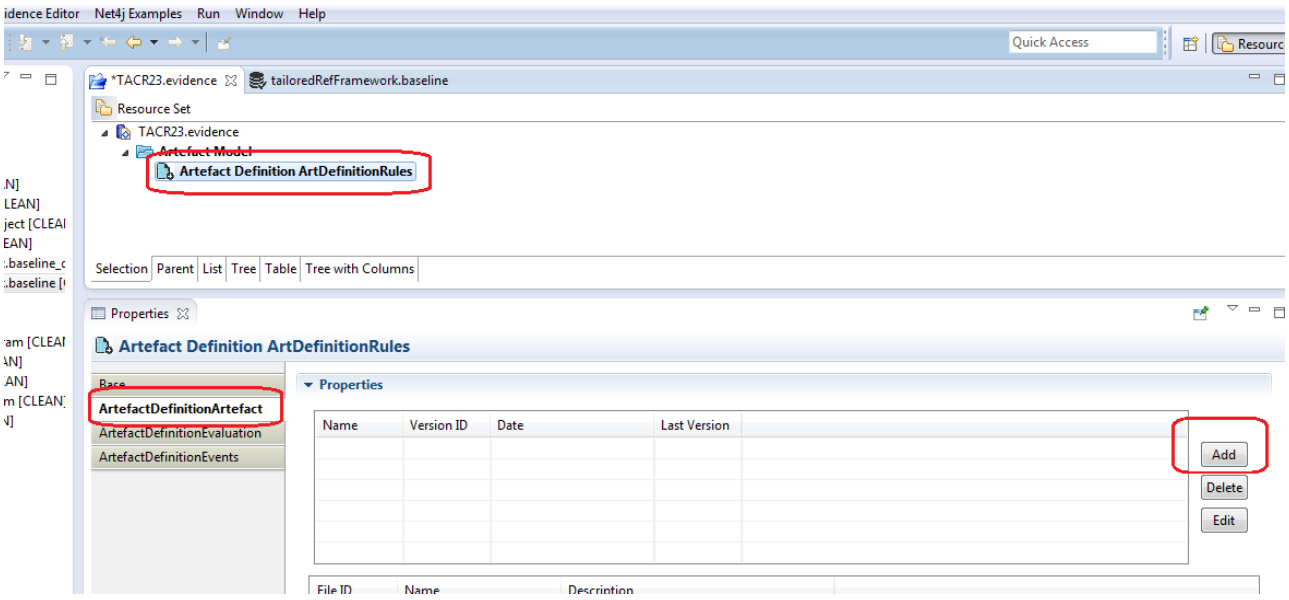


Figure 217 - Add New Artefact (II)

When user modifies one Artefact, the system automatically adds to it an AssuranceAssetEvent of type Modification.

In the properties zone, the framework presents several fields to describe the new Artefact divided in tabs:

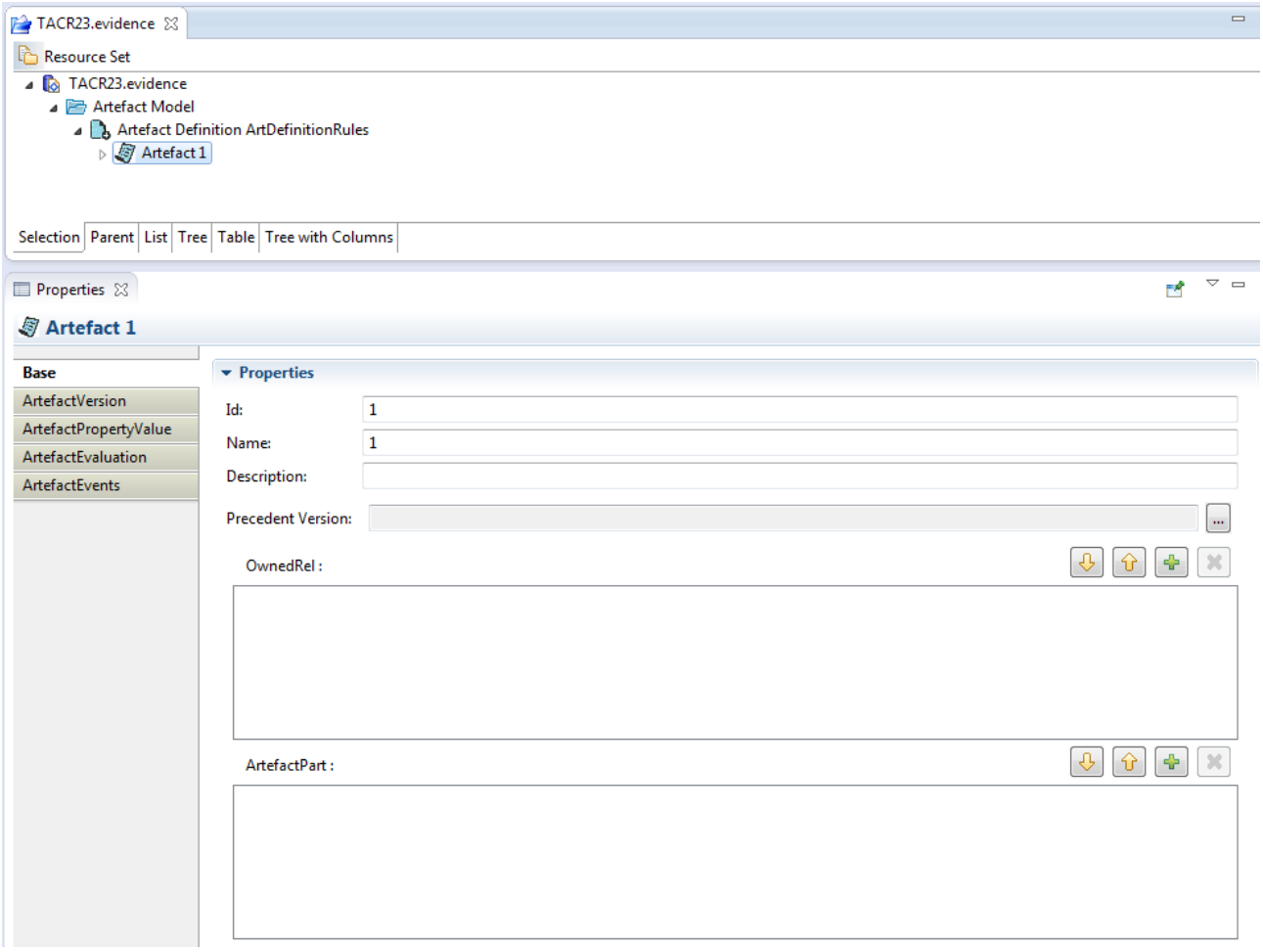


Figure 218 - Artefact Description

1. Artefact Definition (base)
    - Id: Artefact Definition identifier.
    - Name: Artefact Definition name.
    - Description: Artefact Definition description.
2. Artefact Version
    - Version ID: Identifier of the artefact version.
    - Date: Date of the artefact version.
    - Changes: Changes make in the artefact version.
    - Last Version: This field shows what artefact is the version in use.
    - Is Template: Check if the artefact is a template.
    - Is Configurable: Check if the artefact is configurable.
    - Resource: List of resources associated to the artefact,

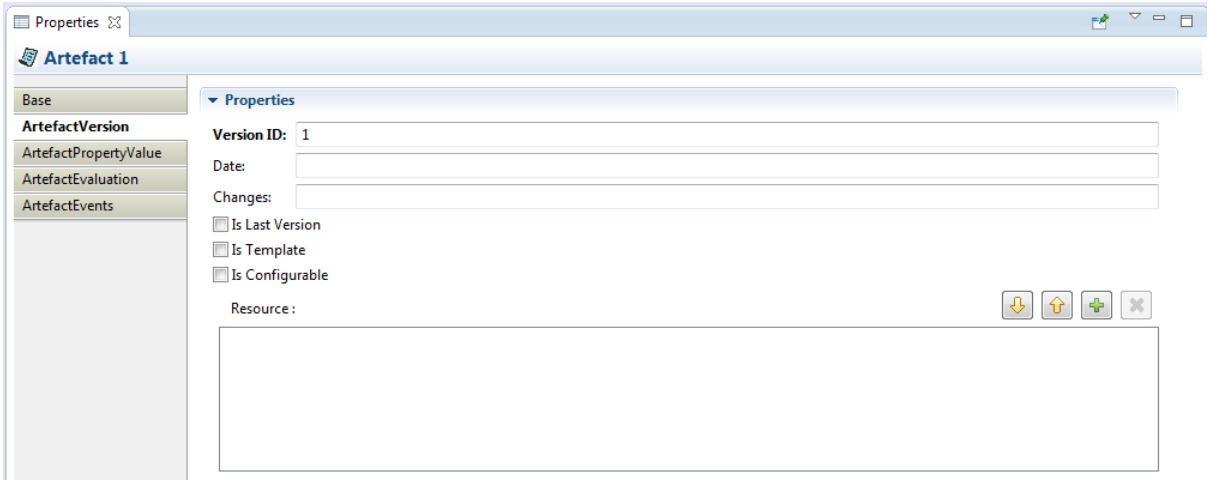


Figure 219 - Description Artefact Version

3. Artefact Property Value
    - Property: Property name.
    - Value: Property value.

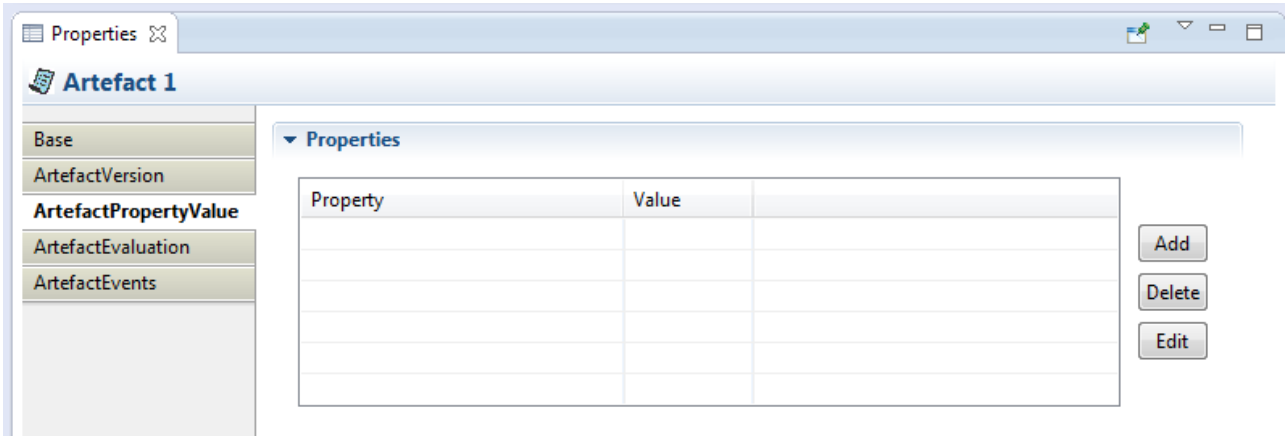


Figure 220 - Description Artefact Property Value

4. Artefact Evaluation
    - Evaluation: References to the assurance asset evaluations that specify the outcome of evaluating the artefact. When a user introduces Evaluation information to an Artefact, and AssuranceAssetEvent of type Evaluation is added automatically to the Artefact.

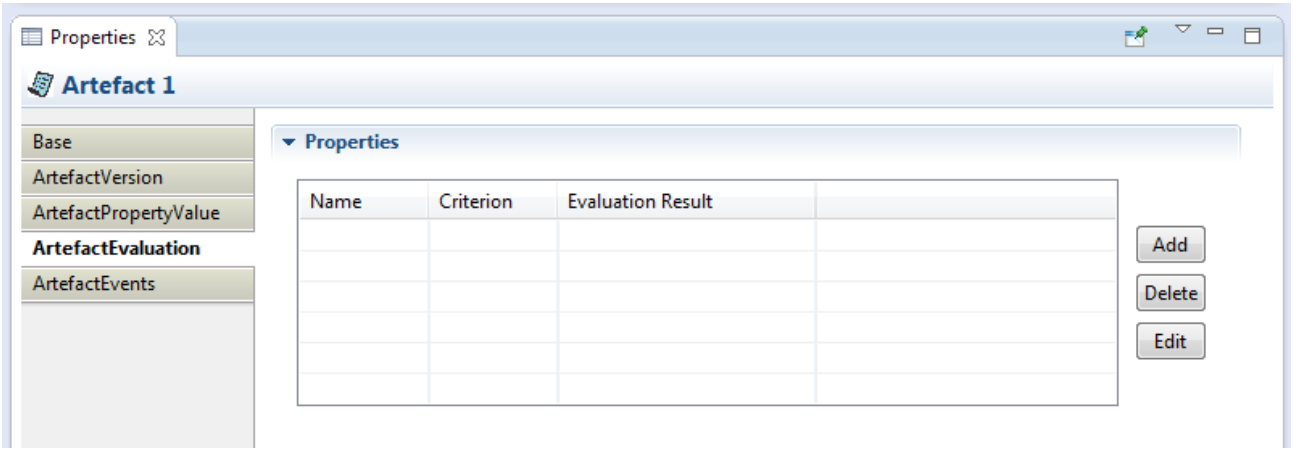


Figure 221 - Description Artefact Evaluation

5. Artefact Events
    - LifecycleEvent: References to the assurance asset events of which the lifecycle of the artefact consists.

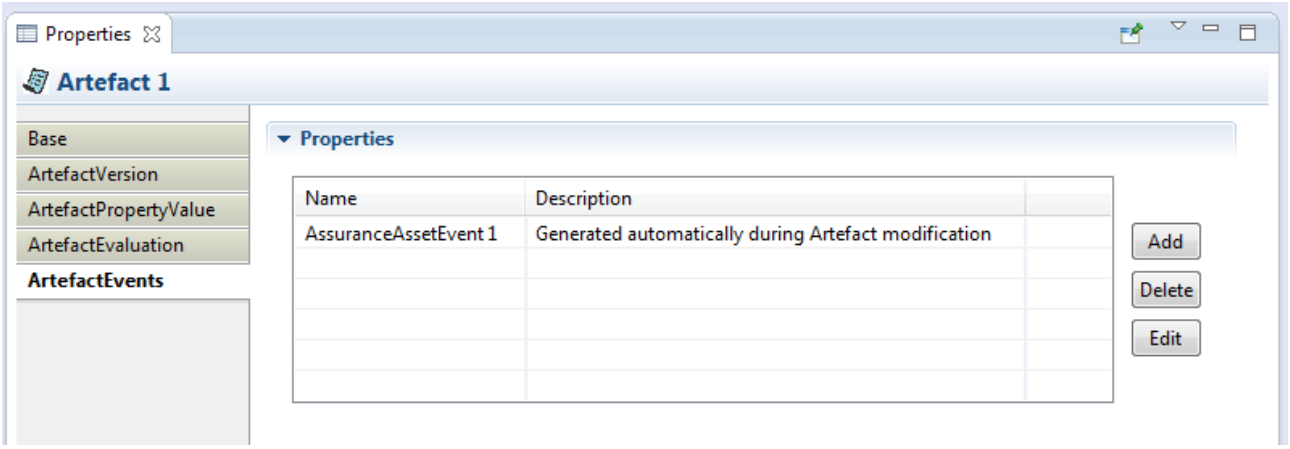


Figure 222 - Description Artefact Events

When adding child Artefact to other Artefact, it's created automatically *ArtefactRel* information with *modificationEffect=MODIFY* and *revocationEffect=MODIFY* with source= *parentArtefact* and *target=child Artefact*.

9.3.2 Delete an artefact.

To delete an artefact:

- Selecting the artefact, press the right mouse button and select the contextual menu *Delete*.

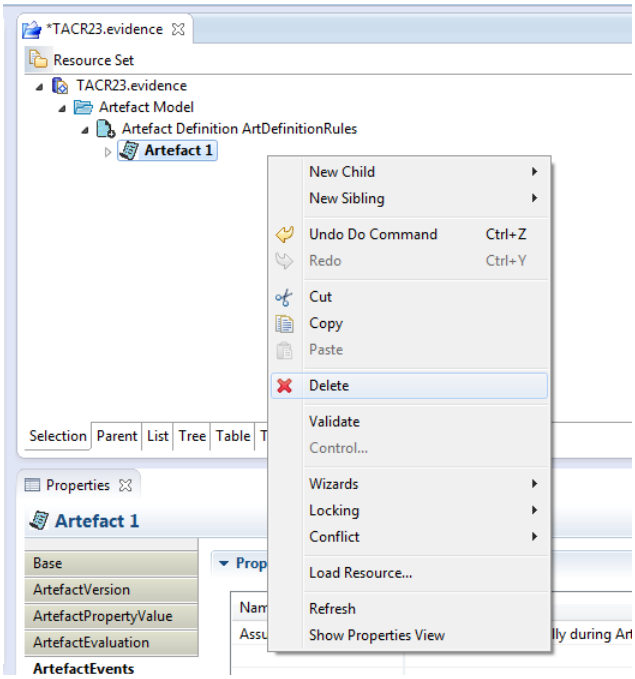


Figure 223 - Delete Artefact I.

- Or select the branch *Artefact Definition that contains the artefact to delete, select the ArtefactDefinitionArtefact tab,* select the artefact and press the button *Delete*

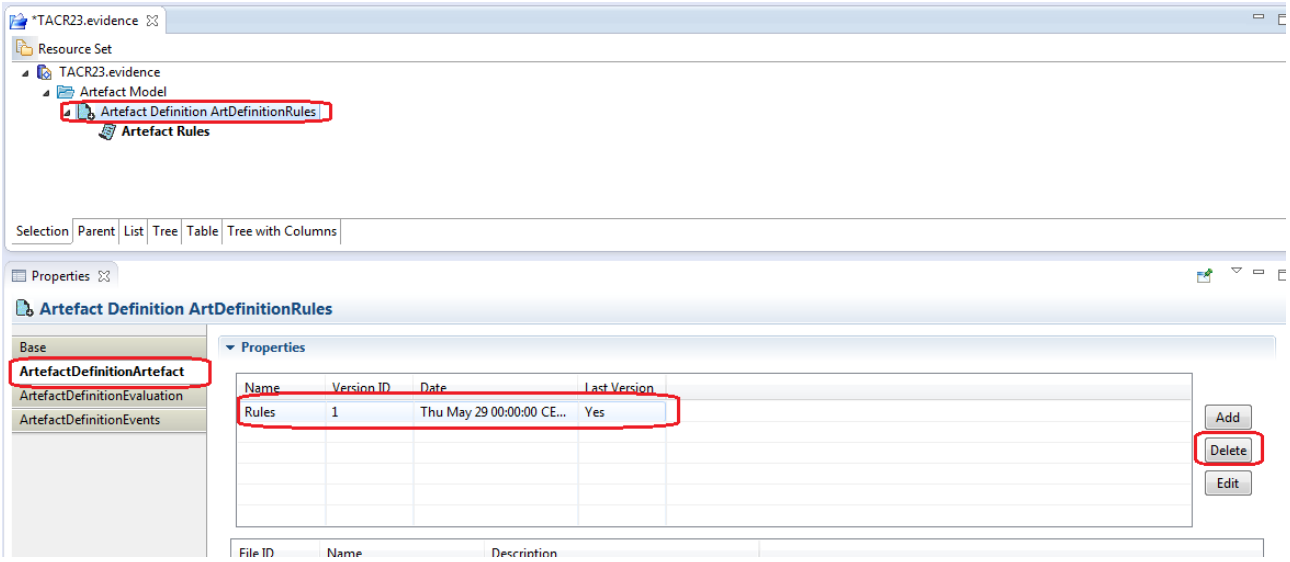


Figure 224 - Delete Artefact II.

9.4 Artefact Resource

9.4.1 Add an artefact resource to an artefact

Once selected the artefact:

- Press the right mouse button and select the contextual menu New Child -> Resource to bring up the Artefact File properties.

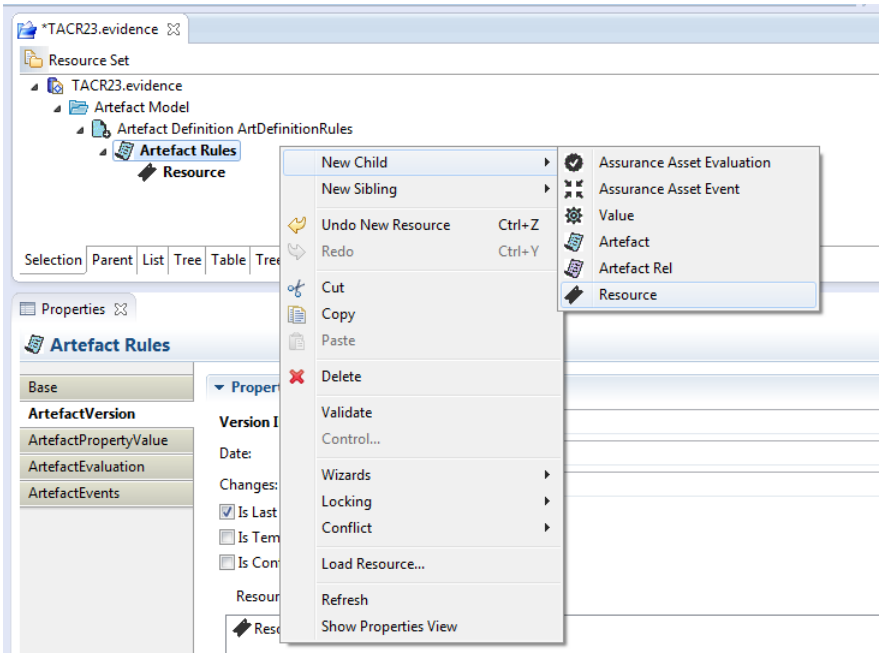


Figure 225 - Add Artefact Resource I

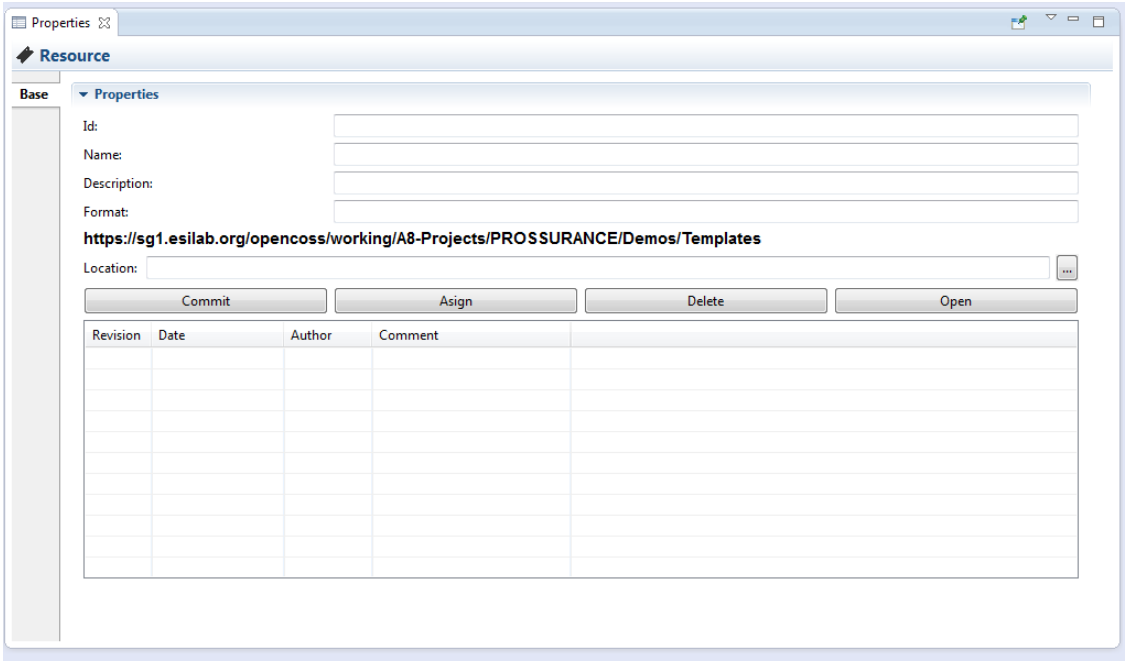


Figure 226 - Resource properties

- Or, select the Artefact Version tab and press the button .

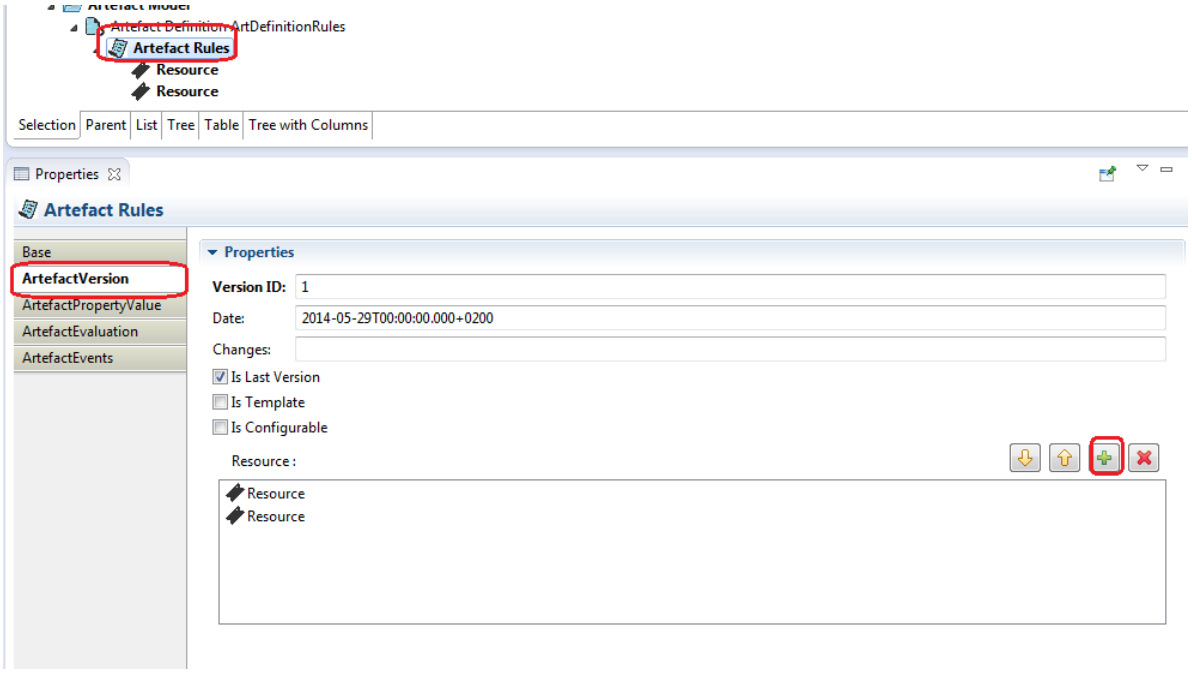


Figure 227 - Add Artefact Resource II

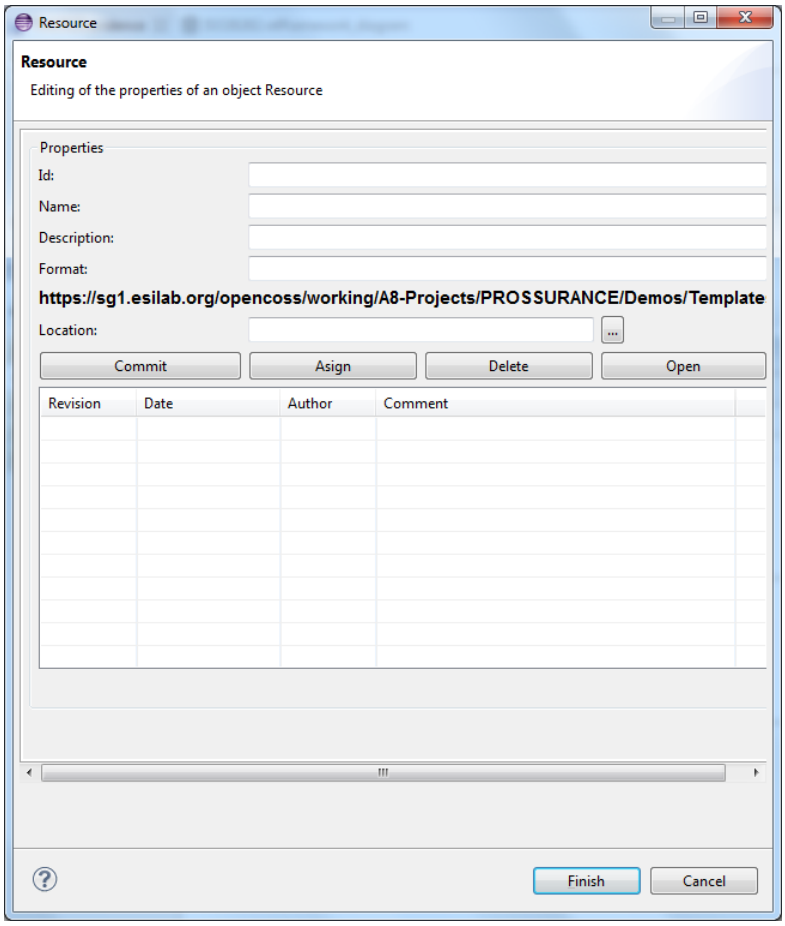


Figure 228 - Resource dialog box

In case of using a Local Repository to add the file, press the button Location or Assign, and select the file that will be added to the artefact resource from the local drive. The URL of the repository will be displayed in bold.

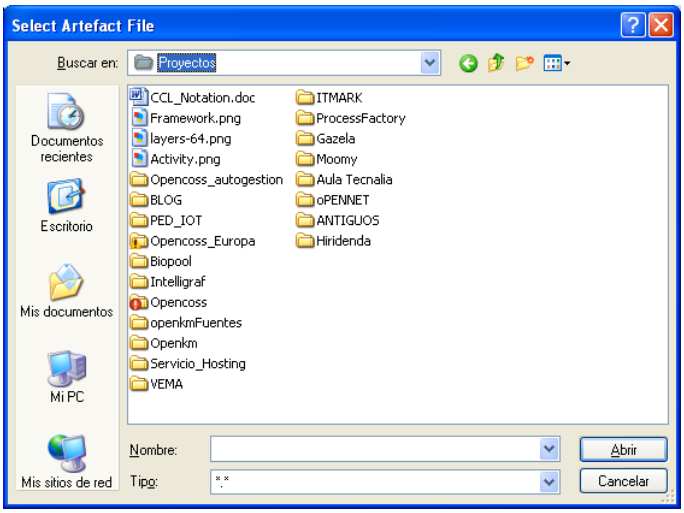


Figure 229 - Select Artefact from the local drive.

1. In case of using a Remote Repository to add the file, press the button Location to select the file that will be added to the artefact resource from the local drive and after press the “Commit” button to

upload it to the SVN server. If the already file exists in the SVN Server use the “Assign” button to select and assign it to the artifact version. Finally the SVN history of the file will be displayed in the table below.

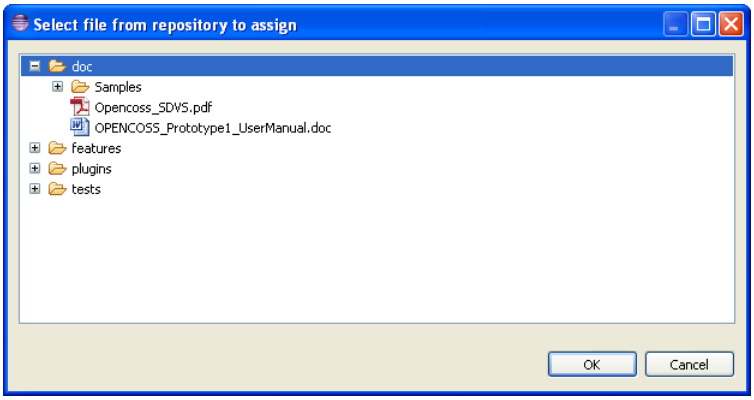


Figure 230 - Select Artefact from the SVN Remote Repository

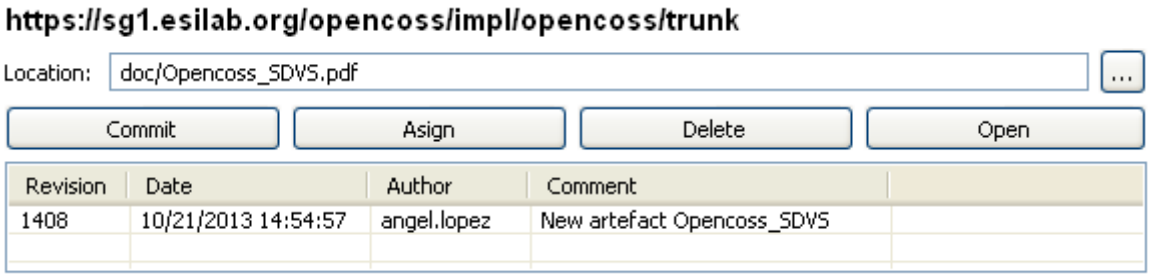


Figure 231 - SVN History table of a File

2. Clicking the “Open” button will launch the corresponding application to open de file. In case of remote repository, the file will be downloaded in a local temporally file.
3. Clicking the “Delete” button will delete the file from the repository, local or remote.

9.4.2 Delete an artefact resource.

To delete an artefact resource:

1. After selecting the artefact resource, press the right mouse button and select the contextual menu *Delete*.

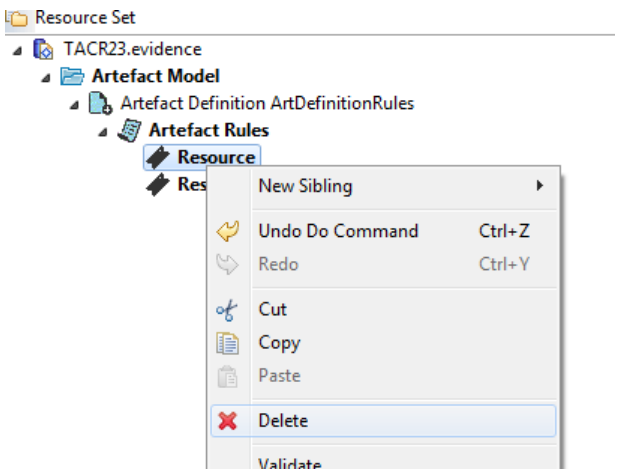


Figure 232 - Delete Artefact Resource I.

2. Or, select the artefact version tab, select the Resource to remove, and press the icon button

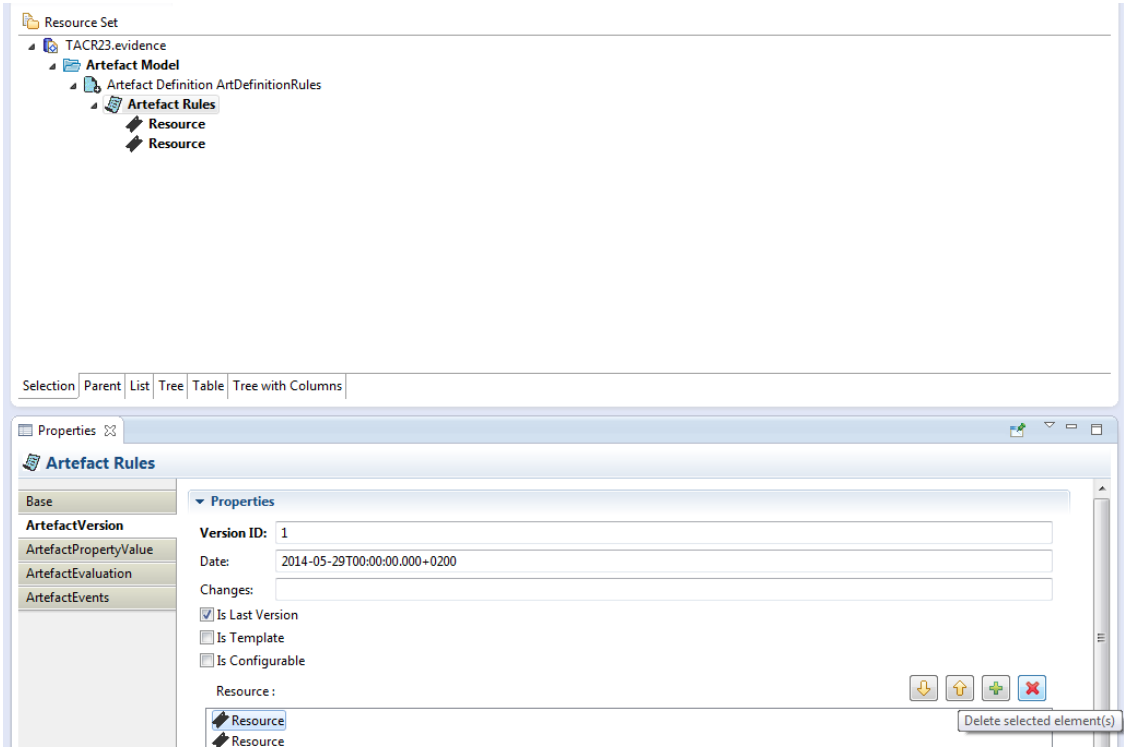


Figure 233 - Delete Artefact Resource II.

9.5 Artefact Property Value

Firstly, it's necessary to load the CDO resource property model (.property). So, press the editing window and select "Load Resource" in the context menu.

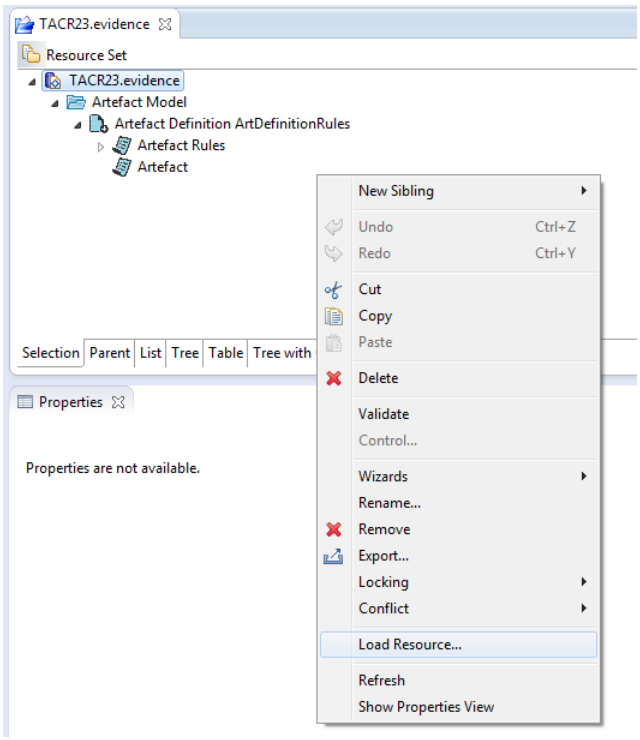


Figure 234 - Load Resource Property model.

Then introduce the URI of the property model.

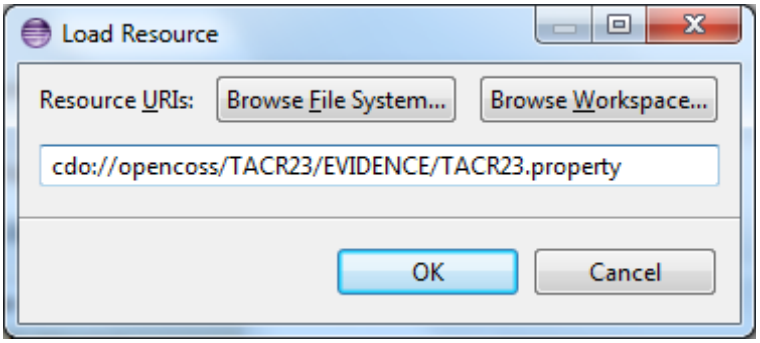


Figure 235 - Select Property model.

9.5.1 Add an artefact property value to an artefact

Once the artefact is selected, it is possible to add an artefact property in two ways:

1. One way, selecting the tab Artefact Property Value and pressing the button Add to bring up the Property Value dialog box:

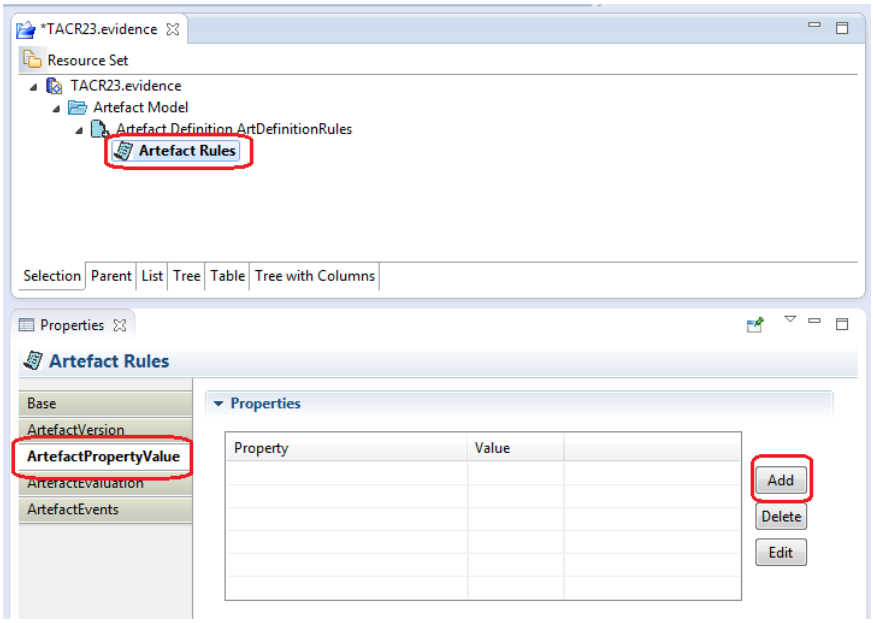


Figure 236 - Add Artefact Property Value I.

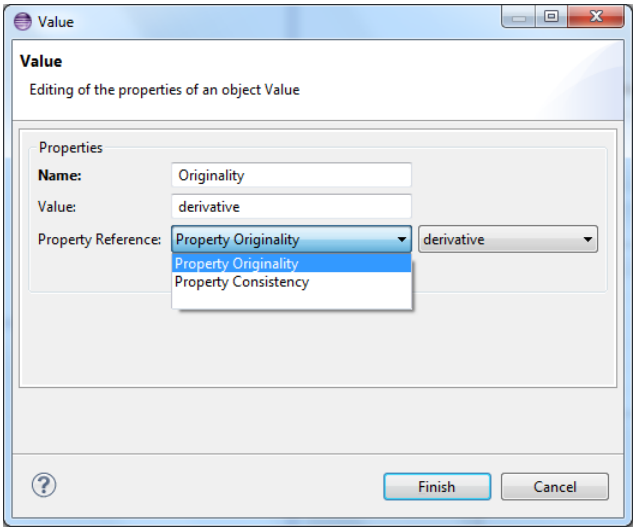


Figure 237 - Artefact Value dialog box

2. Another way, pressing the right mouse button and selecting the contextual menu New Child -> Value to bring up the Artefact Property properties.

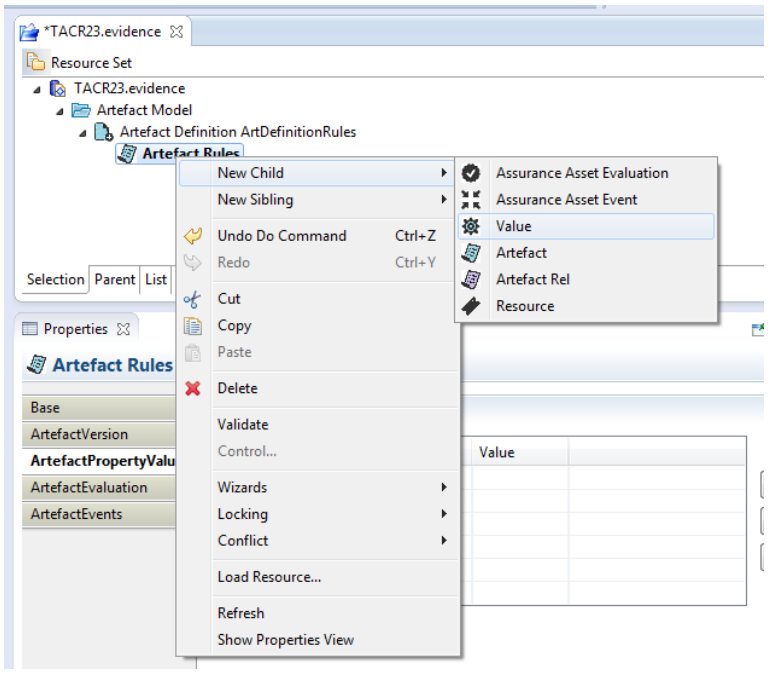


Figure 238 - Add Artefact Property Value II

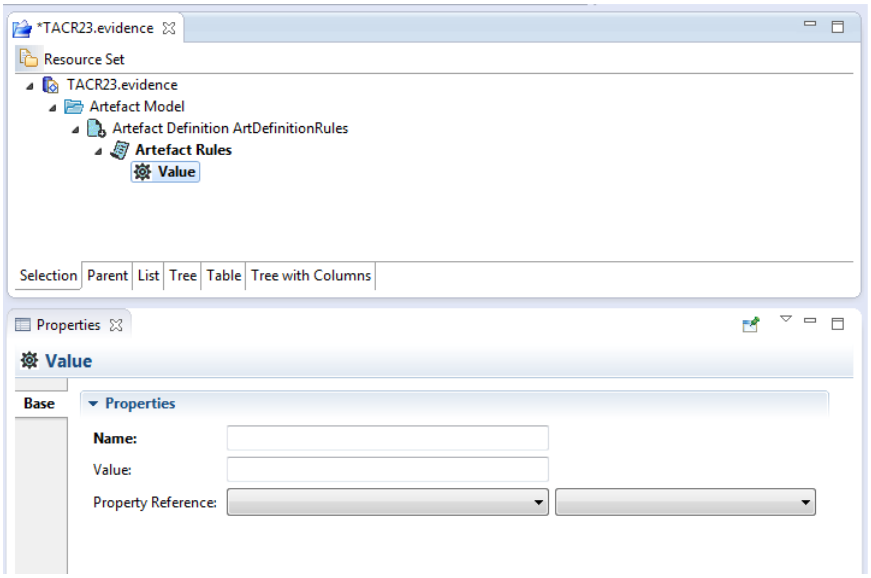


Figure 239 - Artefact Property properties

9.5.2 Delete an artefact property value

It is possible to delete an artefact property in two ways:

1. One way, select the artefact property in the tree, press the right mouse button and select the contextual menu *Delete*.

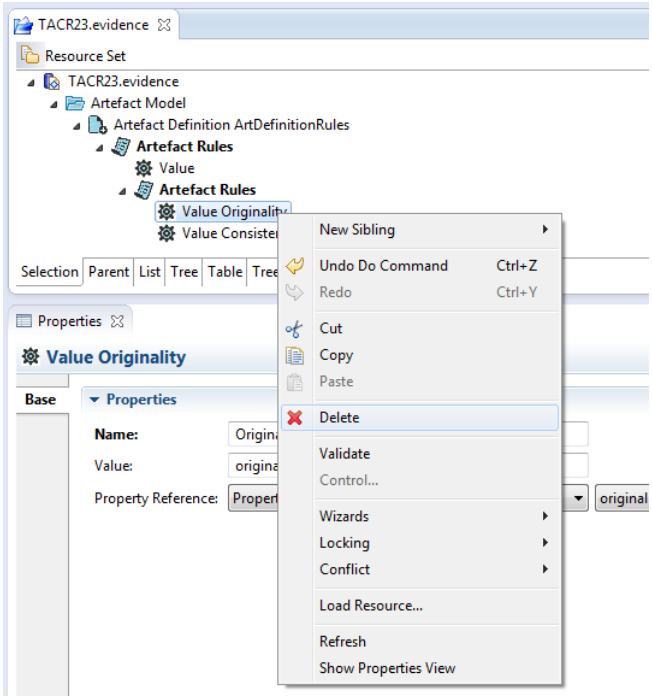


Figure 240 - Delete Artefact Property Value I

2. Another way, select the parent artefact of the artefact property to remove in the tree, select the Artefact Property value tab, select the artefact property and select the button Delete.

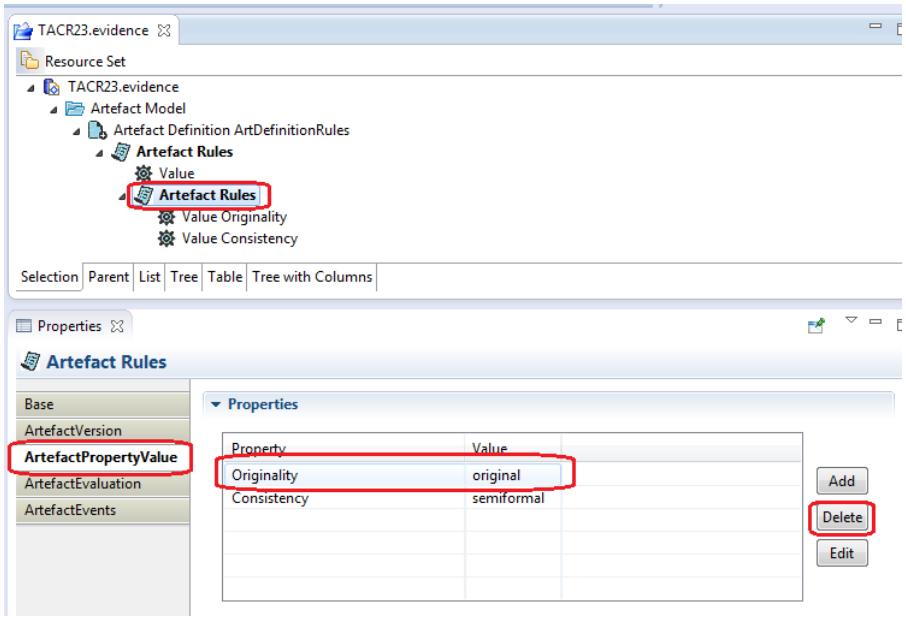


Figure 241 - Delete Artefact Property Value II

9.6 Artefact Assurance Asset Evaluation

9.6.1 Add an artefact assurance asset evaluation to an artefact

Once the artefact is selected, it is possible to add an assurance asset evaluation in two ways:

1. One way, selecting the tab Artefact Evaluation and pressing the button Add to bring up the an Assurance Asset Evaluation dialog box:

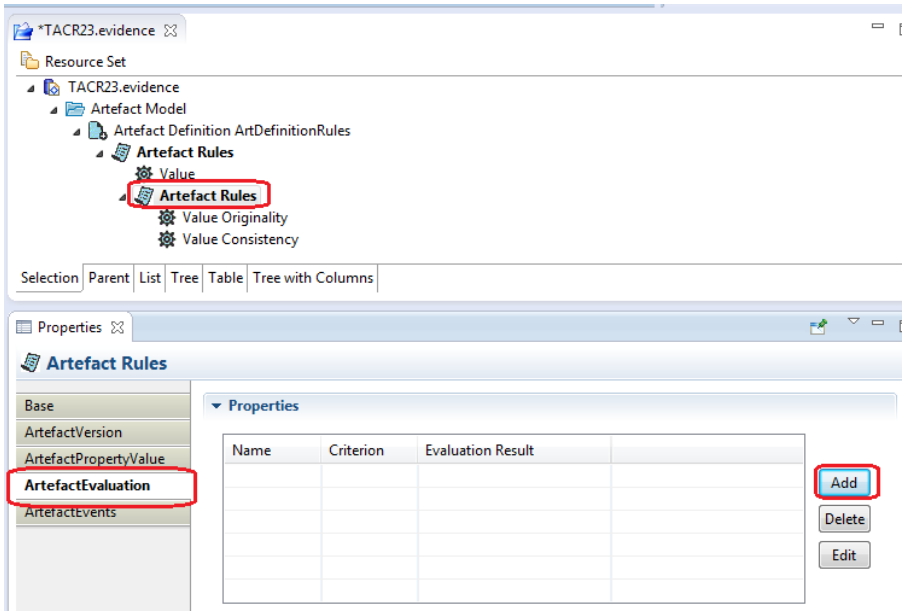


Figure 242 - Add Artefact Assurance Asset Evaluation I

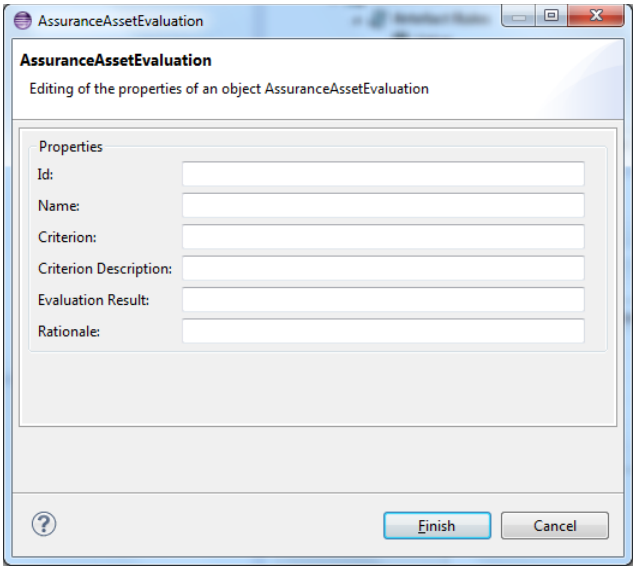


Figure 243 - Artefact Assurance Asset Evaluation dialog box

2. Another way, pressing the right mouse button and selecting the contextual menu New Child -> Assurance Asset Evaluation to bring up the Assurance Asset Evaluation properties.

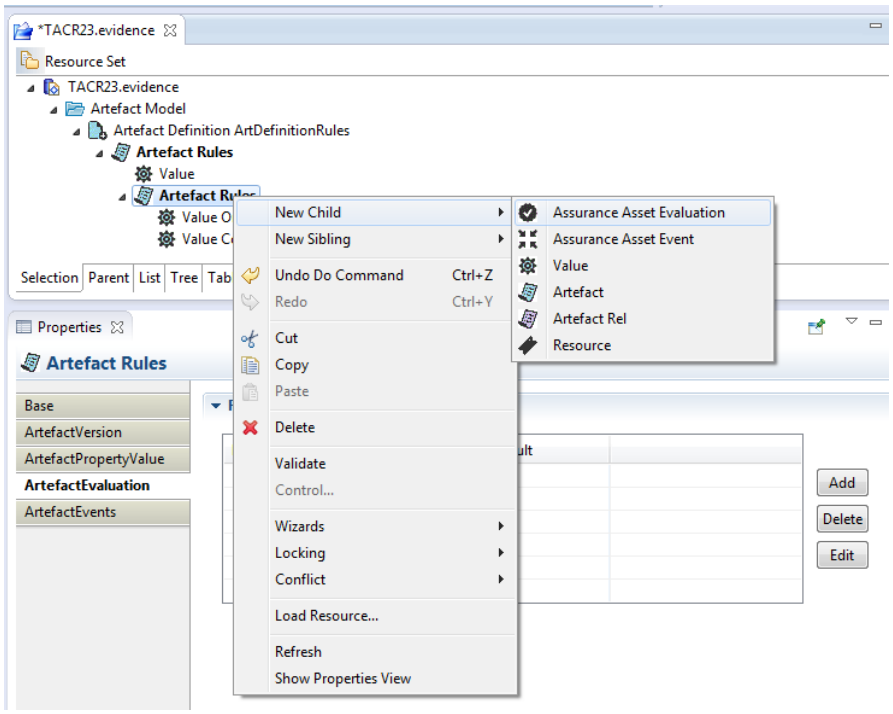


Figure 244 - Add Artefact Assurance Asset Evaluation II

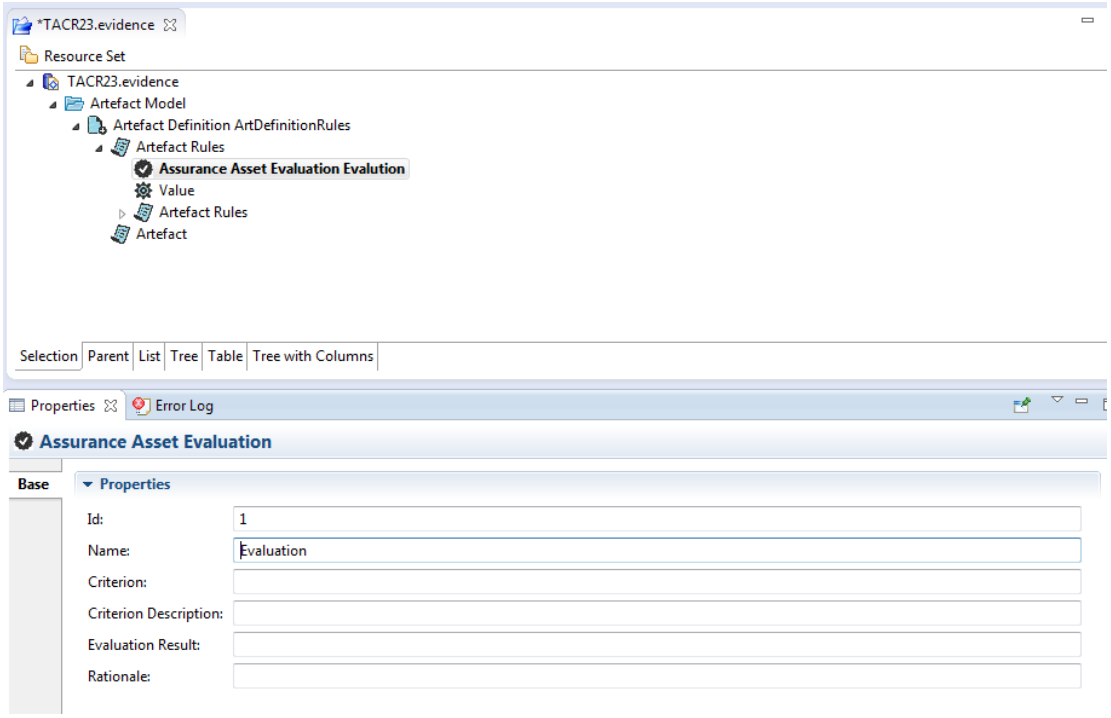


Figure 245 - Artefact Assurance Asset Evaluation properties

9.6.2 Delete an artefact assurance asset evaluation

It is possible to delete an assurance assets evaluation in two ways:

1. One way, select the assurance assets evaluation in the tree, press the right mouse button and select the contextual menu *Delete*.

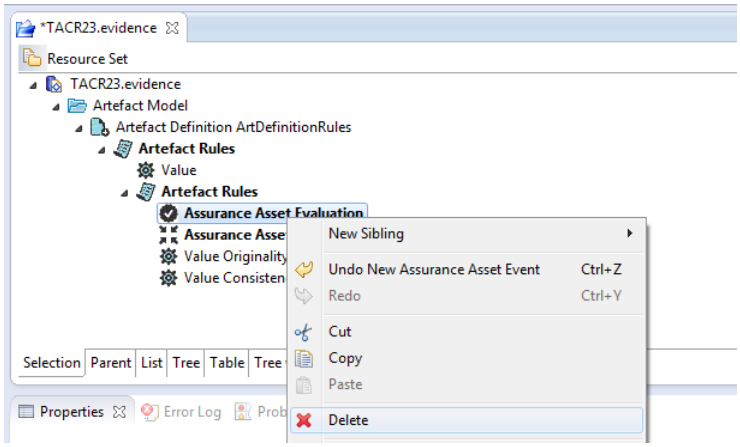


Figure 246 - Delete Artefact Assurance Asset Evaluation I

2. Another way, select the parent artefact of the assurance asset evaluation to remove in the tree, select the Artefact Evaluation tab, select assurance asset evaluation and select the button Delete.

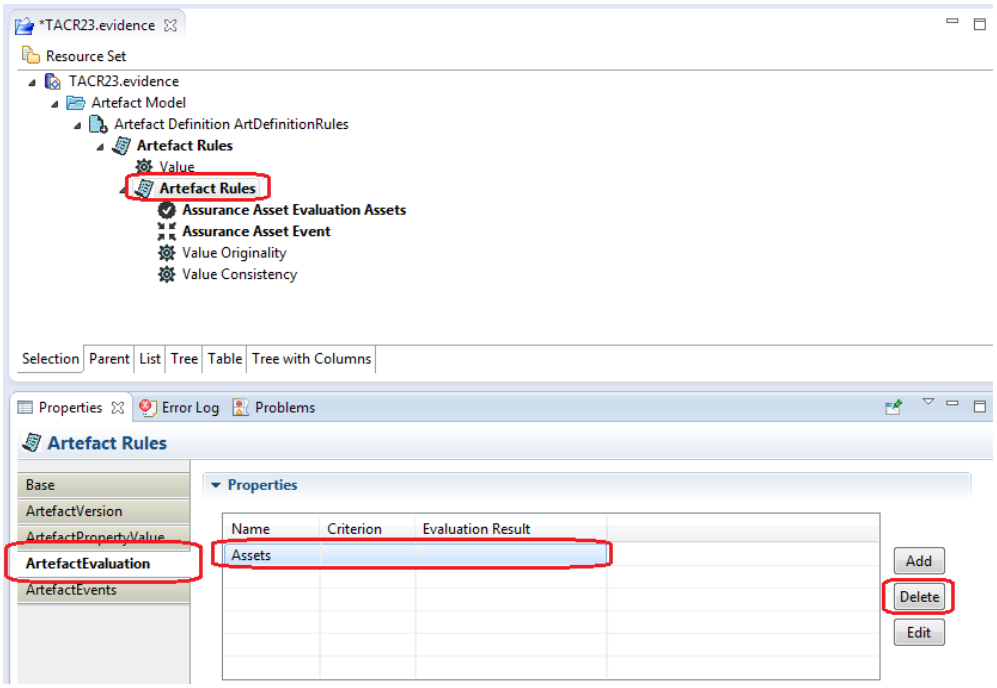


Figure 247 - Delete Artefact Assurance Asset Evaluation II

9.7 Artefact Assurance Asset Events.

9.7.1 Add an artefact assurance asset event to an artefact

Once the artefact is selected, it is possible to add an assurance asset event in two ways:

1. One way, selecting the tab Artefact Events and pressing the button Add to bring up the an Assurance Asset Event dialog box:

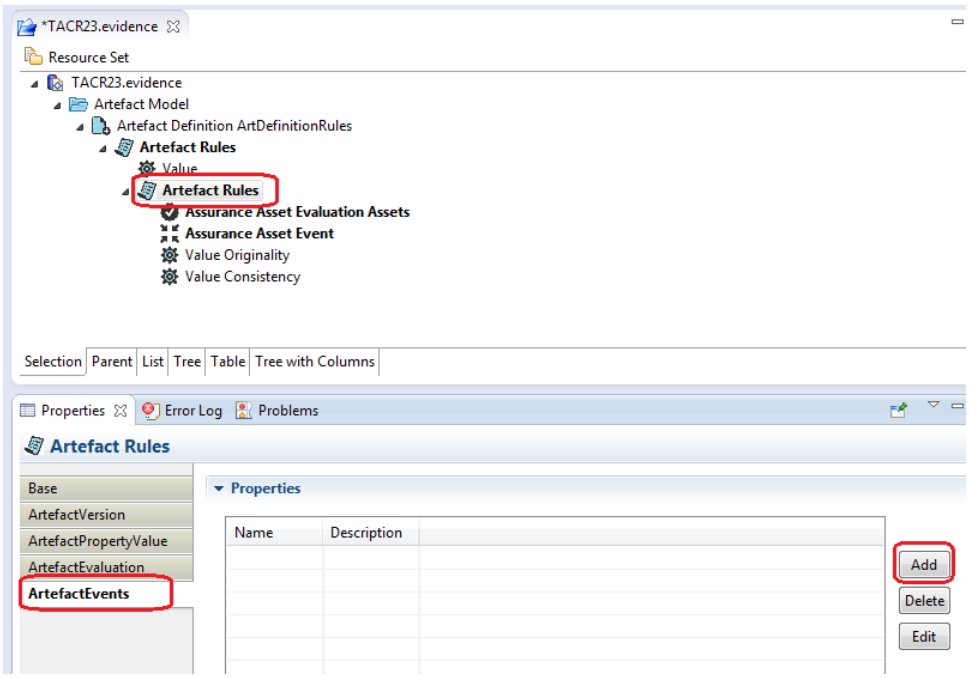


Figure 248 - Add Artefact Assurance Asset Event I

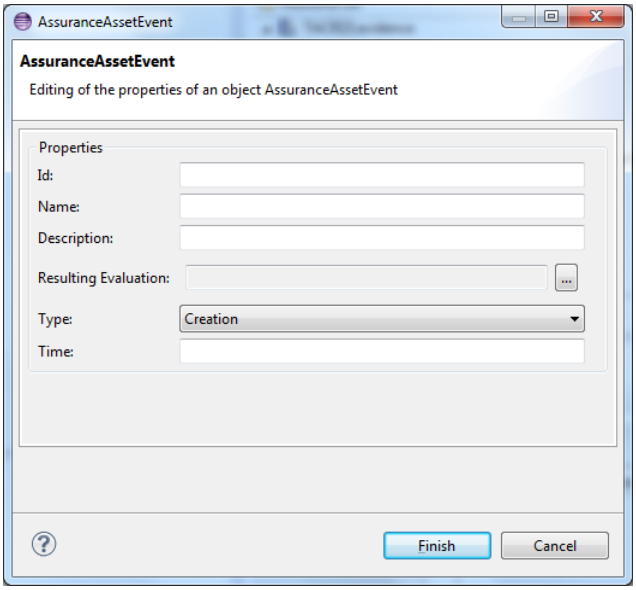


Figure 249 - Artefact Assurance Asset Event dialog box

2. Another way, pressing the right mouse button and selecting the contextual menu New Child -> Assurance Asset Evaluation to bring up the Assurance Asset Evaluation properties.

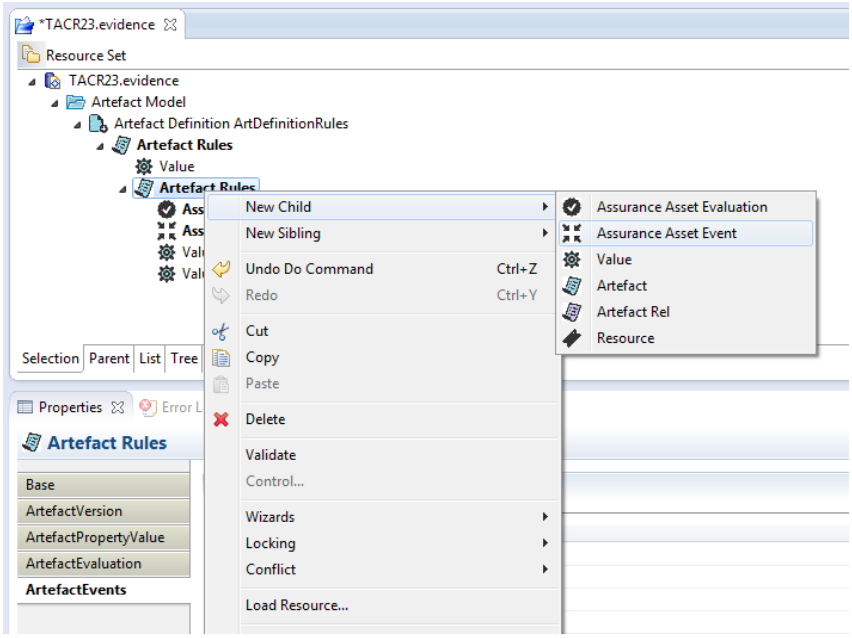


Figure 250 - Add Artefact Assurance Asset Event II

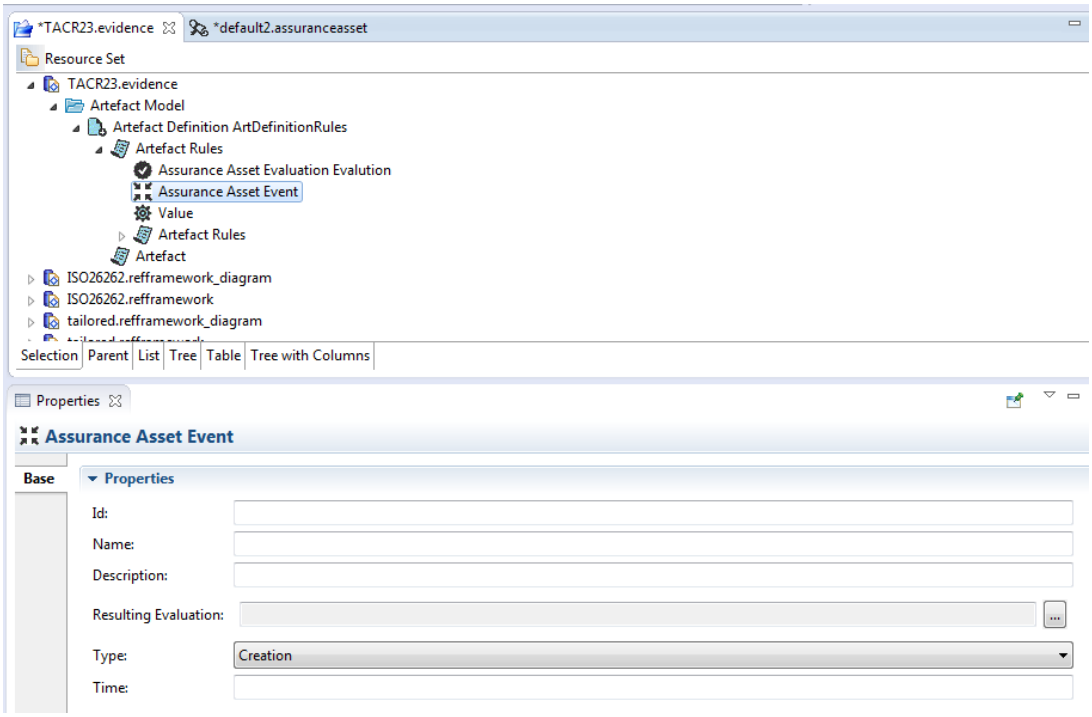


Figure 251 - Artefact Assurance Asset Event properties

9.7.2 Delete an artefact assurance asset event

It is possible to delete an assurance assets event in two ways:

1. One way, select the assurance assets event in the tree, press the right mouse button and select the contextual menu *Delete*.

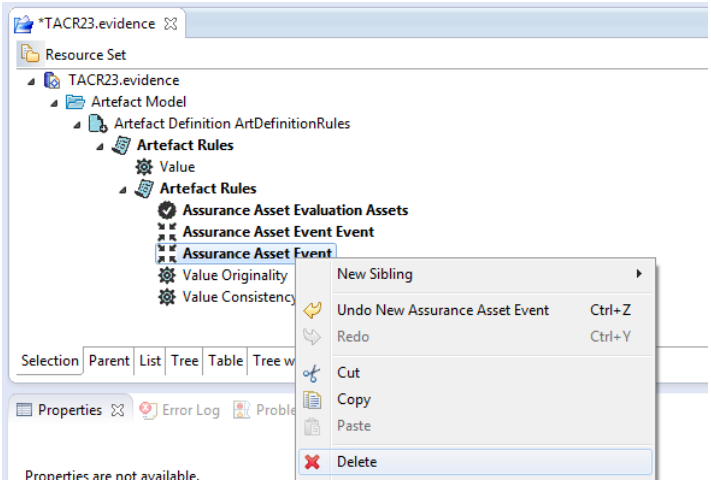


Figure 252 - Delete Artefact Assurance Asset Event I

2. Another way, select the parent artefact of the assurance asset event to remove in the tree, select the Artefact Event tab, select assurance asset event and select the button Delete

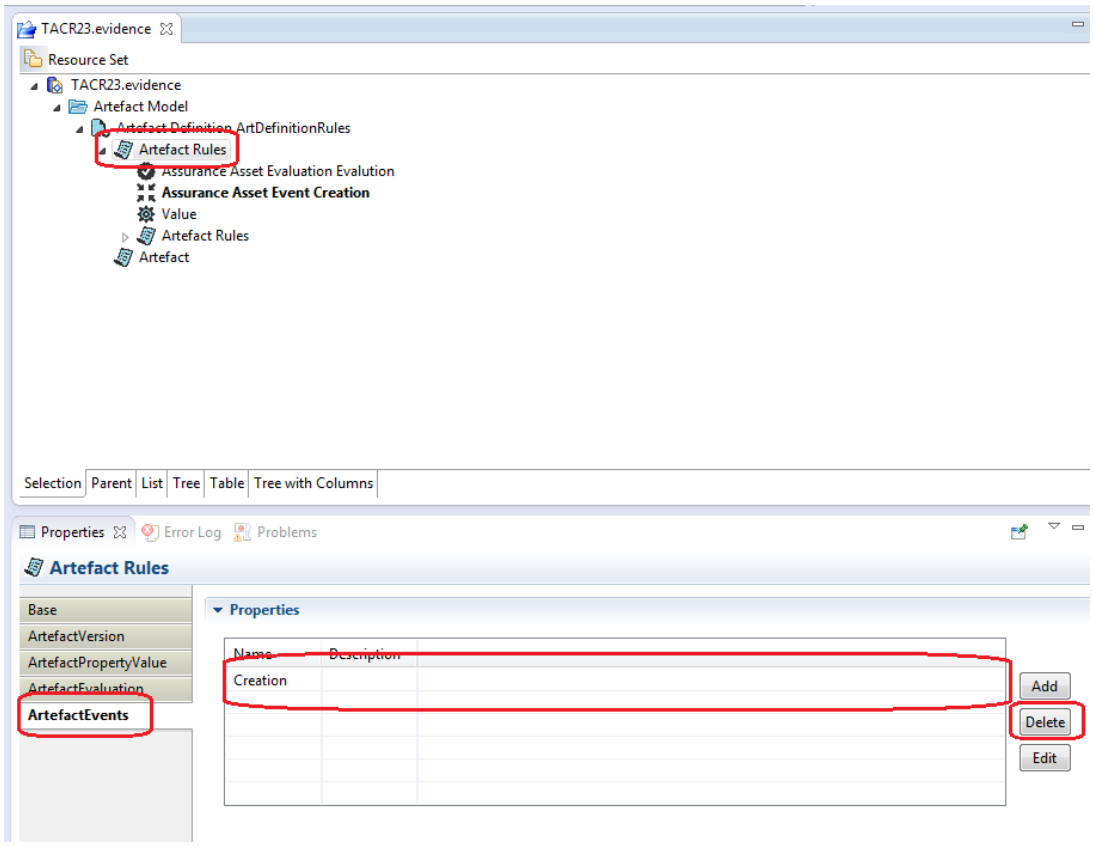


Figure 253 - Delete Artefact Assurance Asset Event II

9.8 Impact analysis

This functionality informs to the user about the impact of the changes in one Artefact that affect to others taking into account the relations between the modified and the impacted artefacts ones.

When the user clicks the save button to store the modifications made in one evidence model, for each Artefact modified with impact to another, the user will be asked in a confirmation dialog if he is agree or not with the showed impact information in form of tree or not.

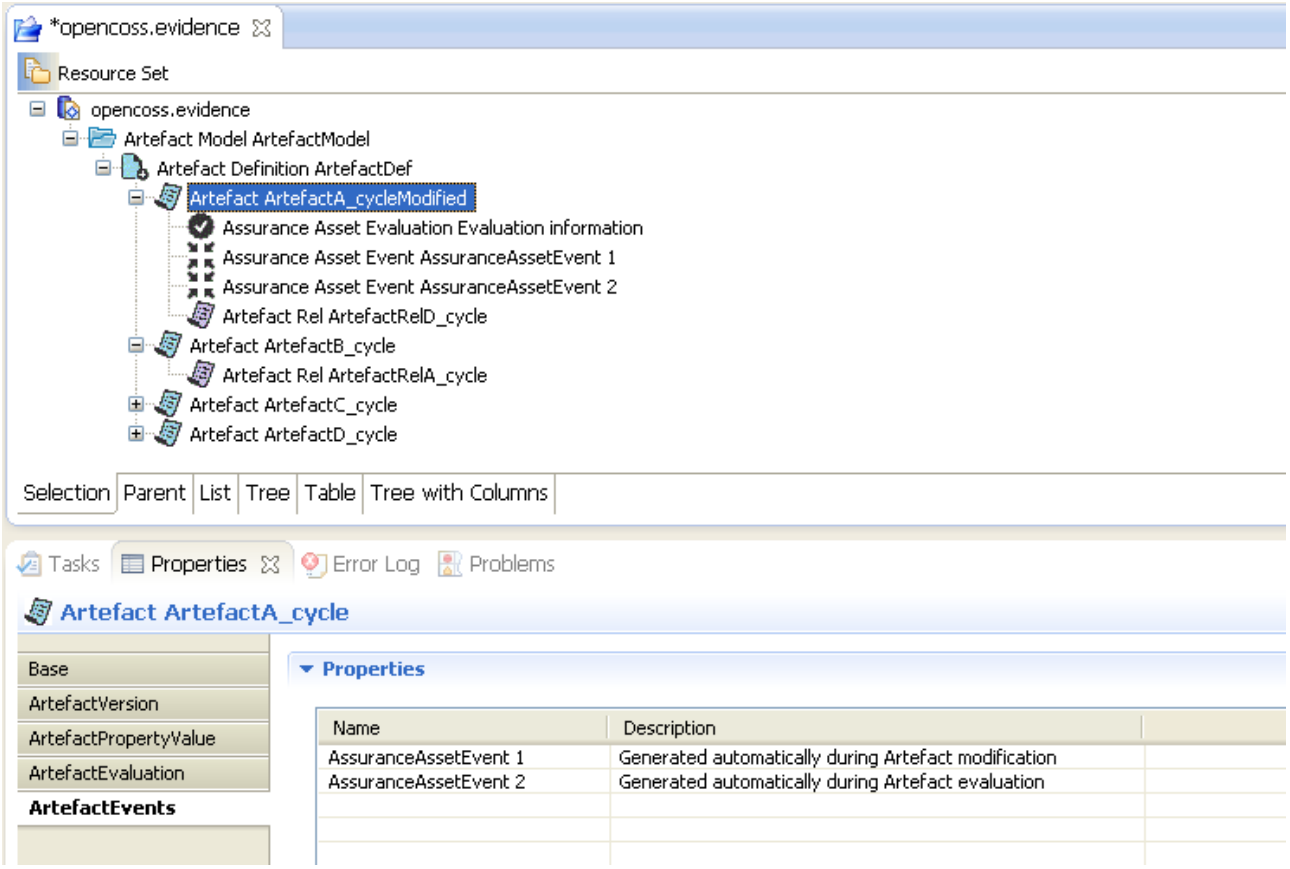


Figure 254 - Artefact modified with automatically generated events

If the user accepts the impact showed new assurance assents events will be generated to the modified and impacted artefacts.

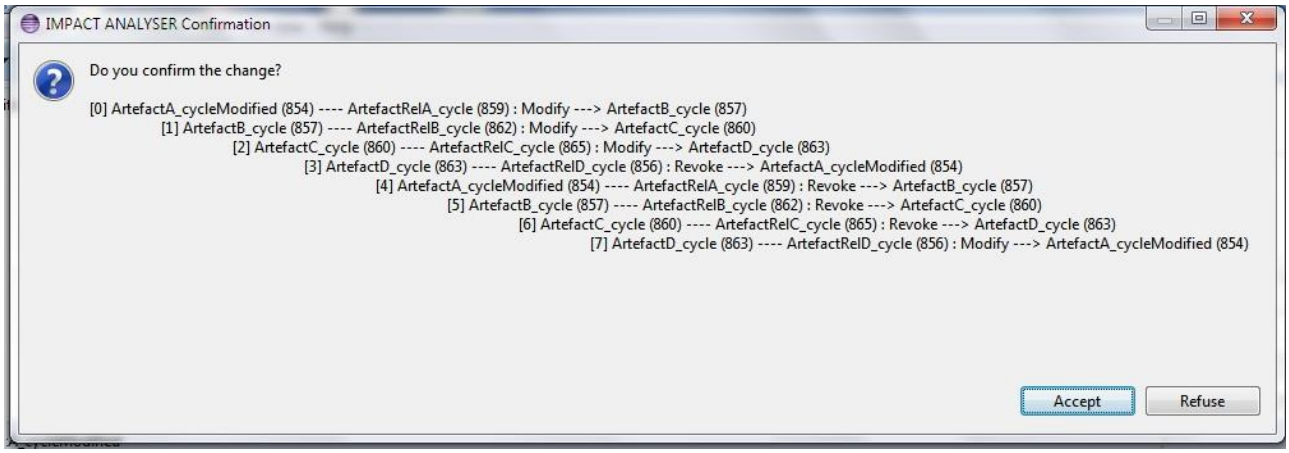


Figure 255 - Artefact analyser confirmation windows

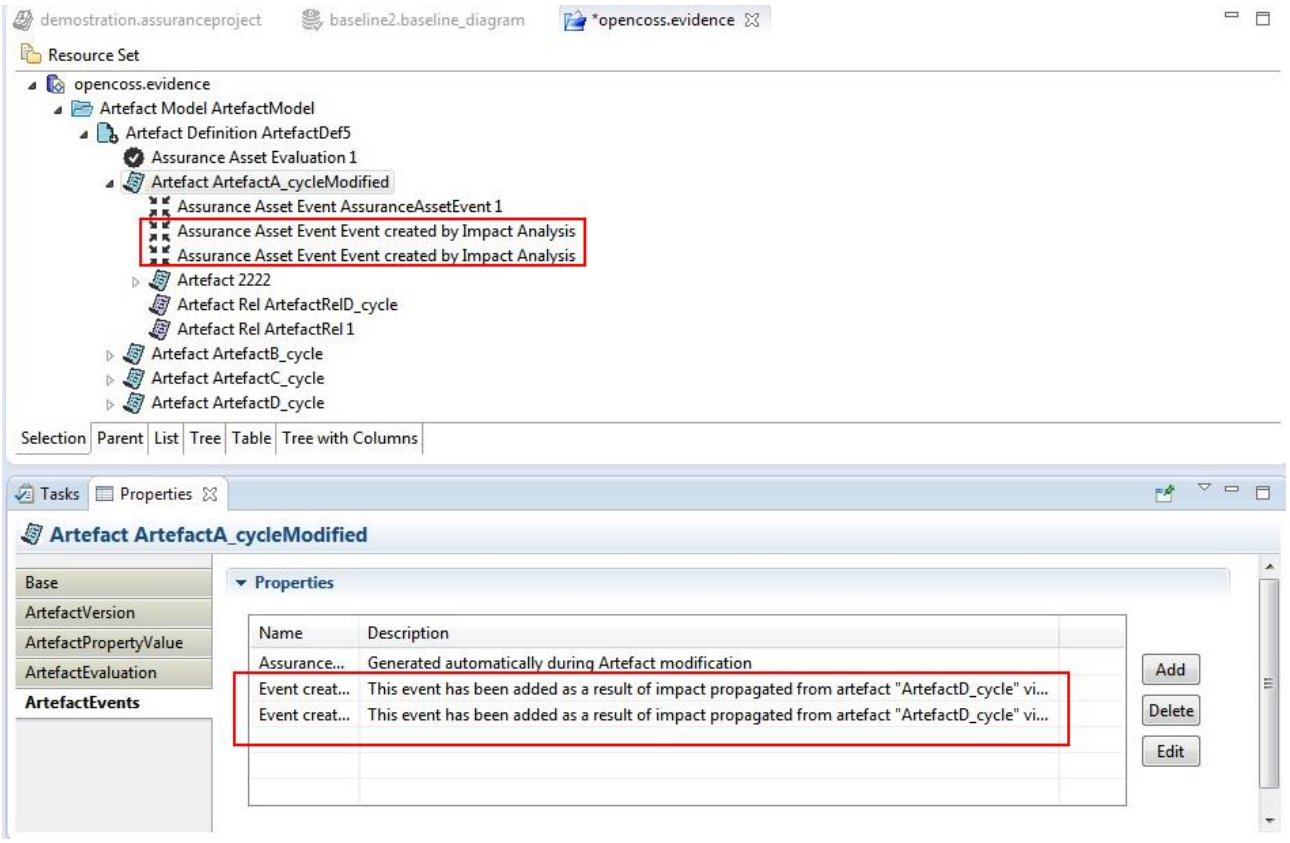


Figure 256 - Artefact events created by Impact Analyser

See also Impact Analysis described in server section in Change Impact Analysis chapter.

9.9 Create new Executed Process

Executed process models can be created by importing EPF information into OpenCert (see Section 6.7 for further details). Alternatively, it is possible to create a new model of the type **Process Model**.

In order to generate a new Process Model, the following steps need to be done:

- First, select the entry of the menu *File -> New -> Other*.
- Inside the category *Opencert*, select the *Process Model* and press the *Next* button.

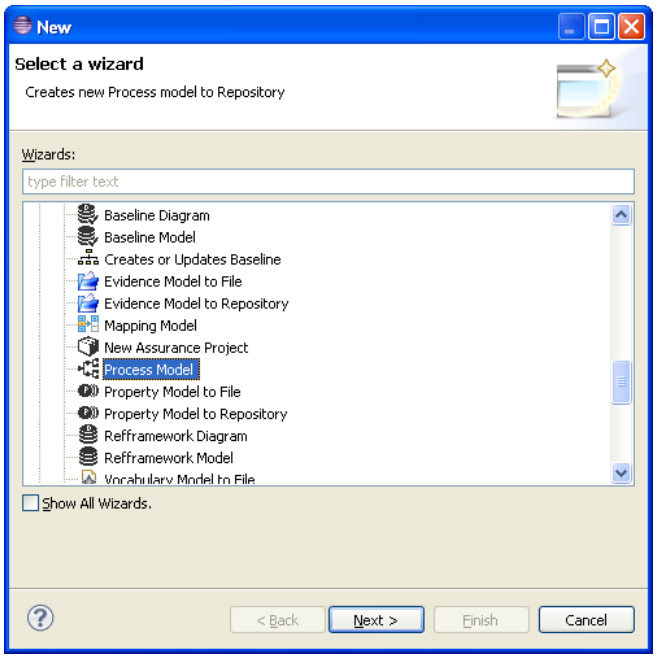


Figure 257 - New Process Model I

- Enter o select the parent folder, name the model and press the Next button.

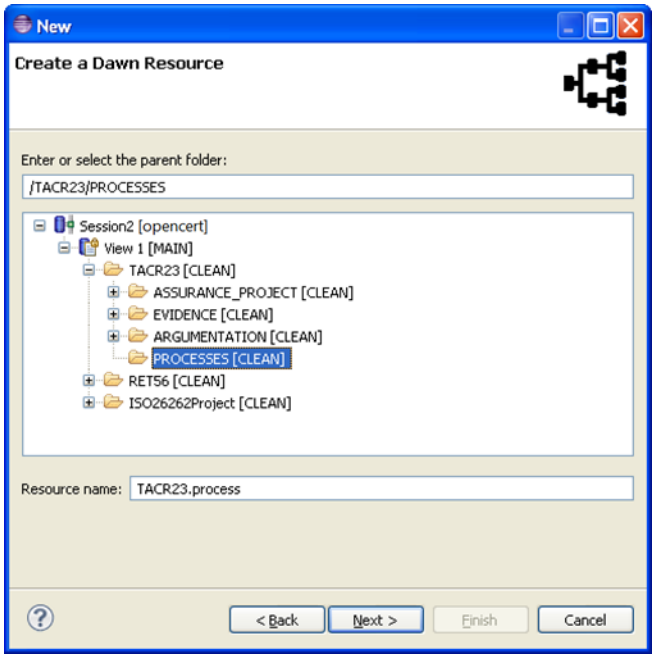


Figure 258 - New Process Model II

- And finally, select the “Model” object to create

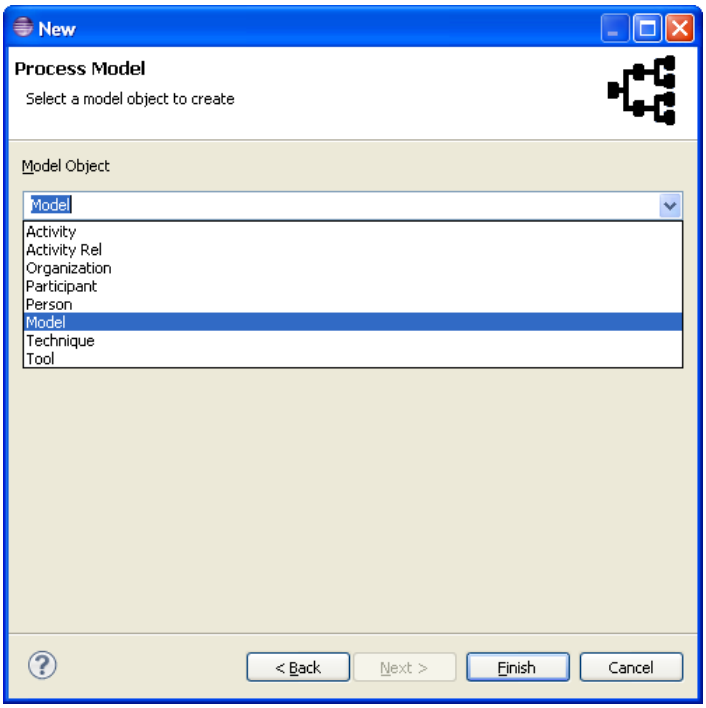


Figure 259 - New Process Model III

Once the Property Model has been created, the first item is presented to the user

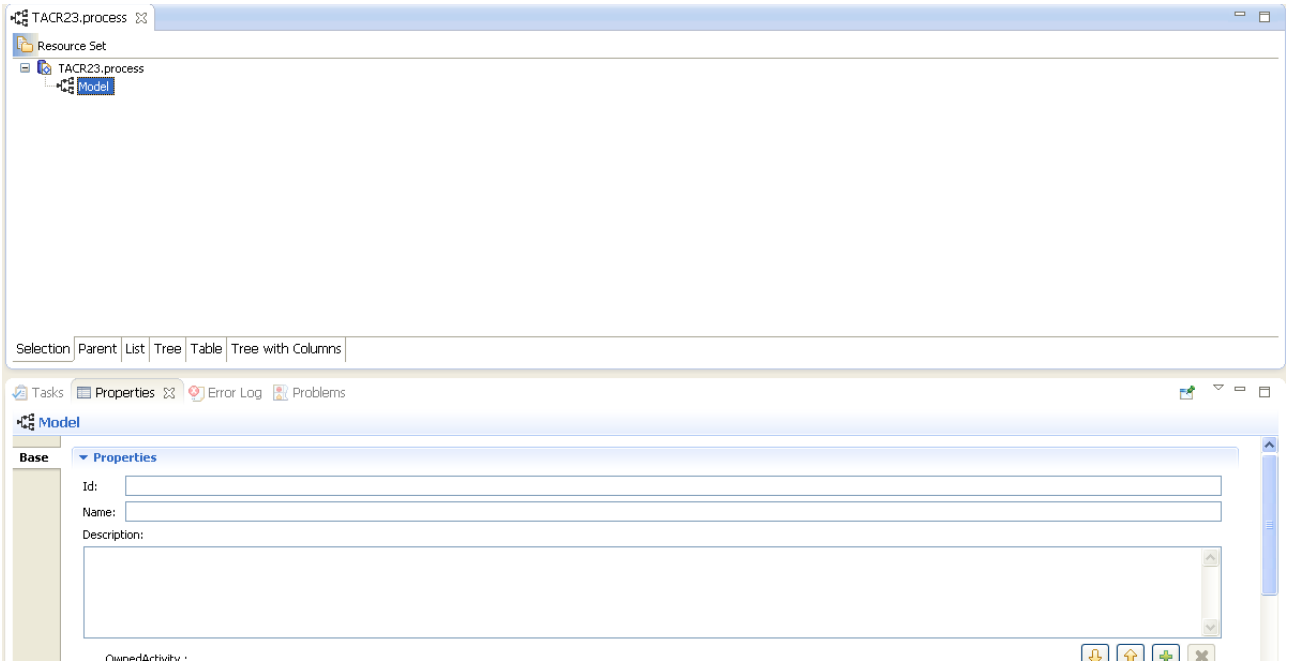


Figure 260 - Process Model

9.10 Creating Process Assurance data

The Process Model allows defining activity, participant, person, tool, organization or technique objects.

To create these objects, in the Model zone, click on the branch *Model* and press the right mouse button and select the contextual menu New Child or use its properties view:

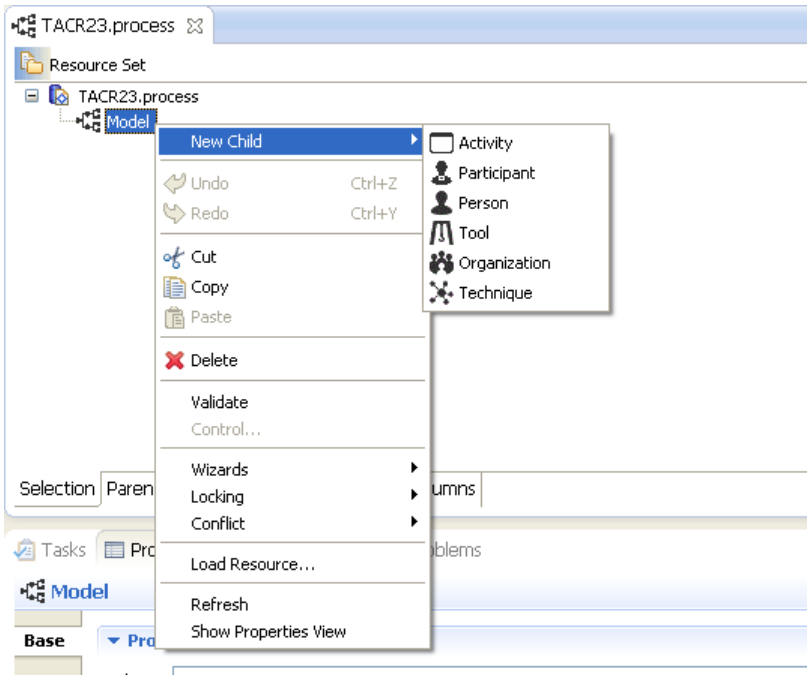


Figure 261 - Create Process Model data using context menu

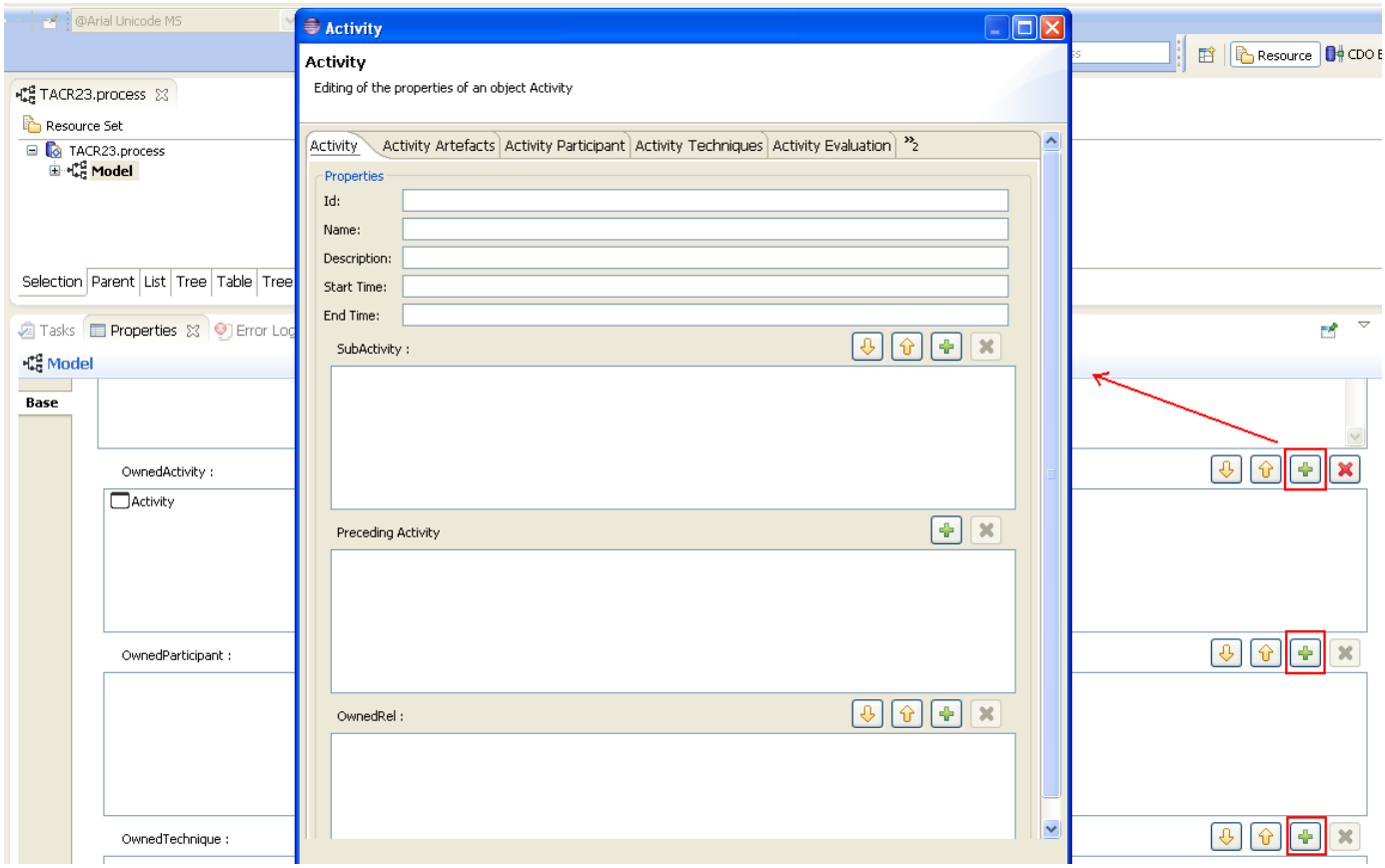


Figure 262 - Create Process Model data using properties View

9.11 Deleting Process Assurance Objects

To delete a Process Assurance Object, select the object to remove, press the right mouse button and select the contextual menu *Delete or select the information to delete using the properties view:*

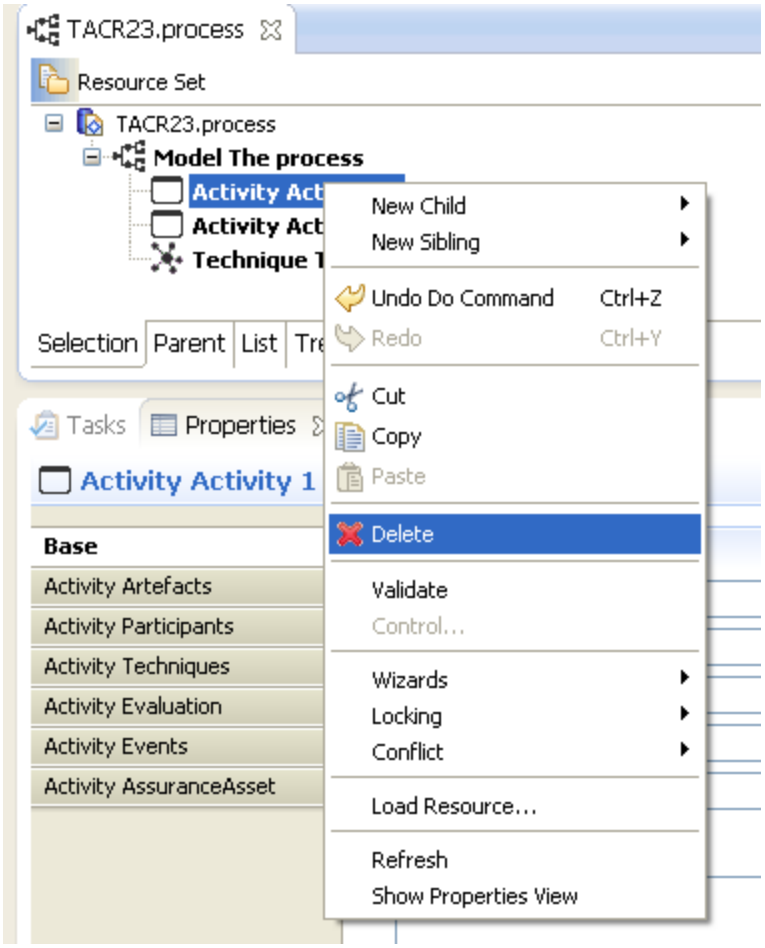


Figure 263 - Delete Process Model data using context menu

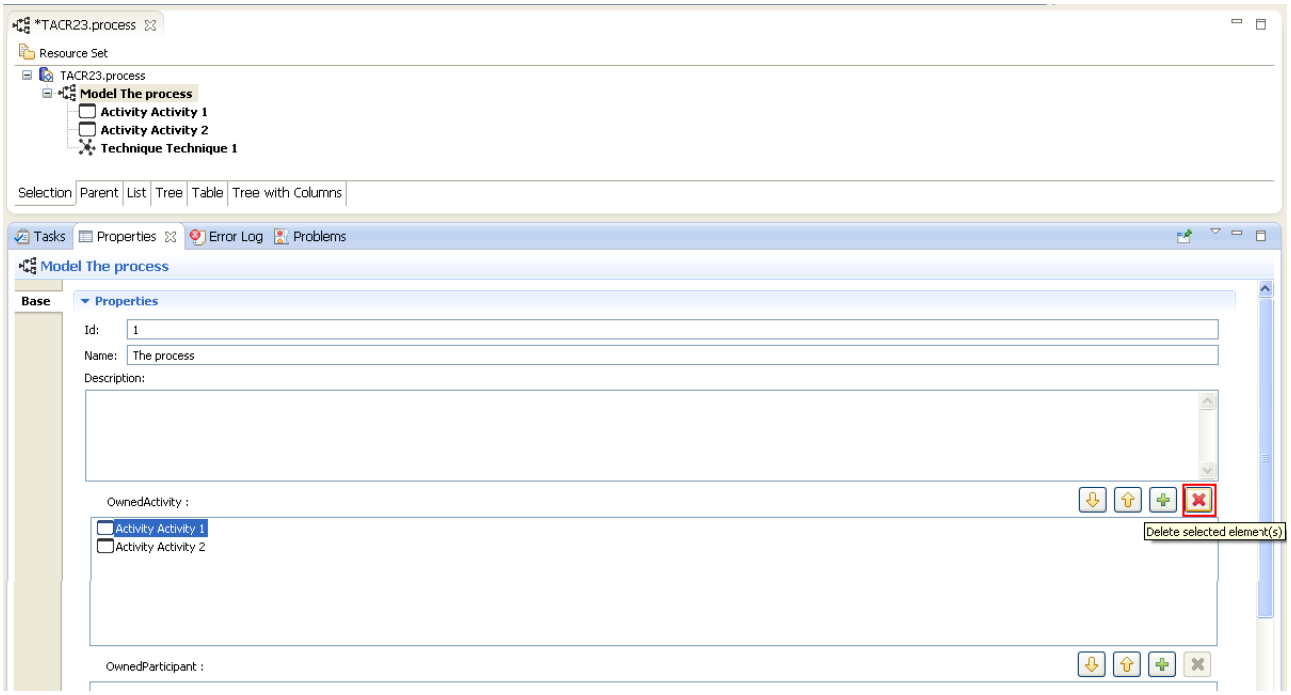


Figure 264 - Delete Process Model data using properties view

9.12 Creation of Property Model

The management of Properties must be made through the creation of a new model of the type **Property Model**.

In order to generate a new Property Model, the following steps need to be done:

- First, select the entry of the menu *File -> New -> Other*:

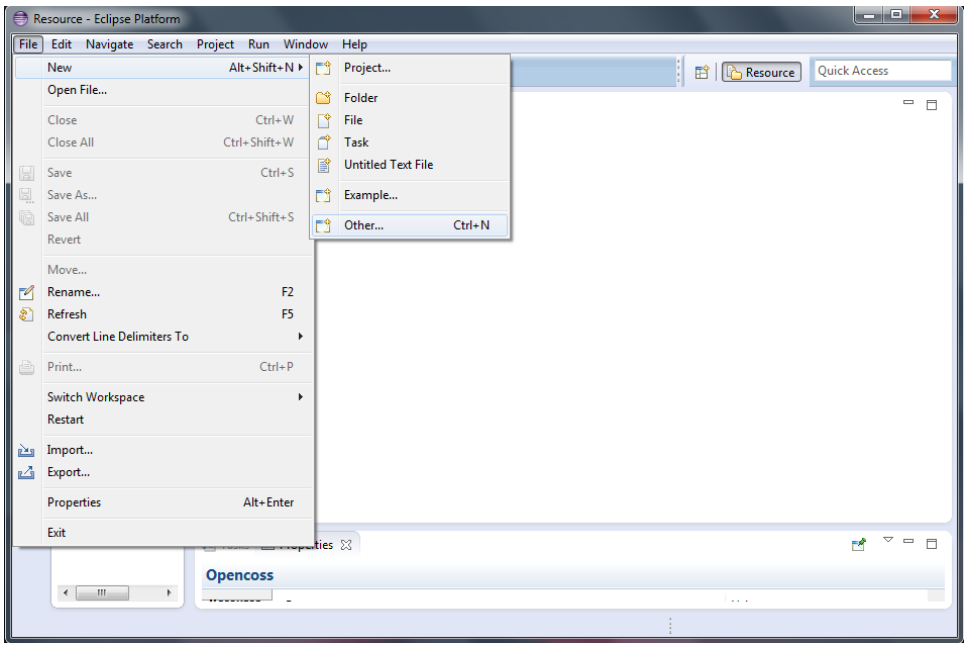


Figure 265 - New Property Model menu File -> New -> Other

- Inside the category of the wizard *Opencert*, select the *Property Model* to the Repository and press the *Next* button:

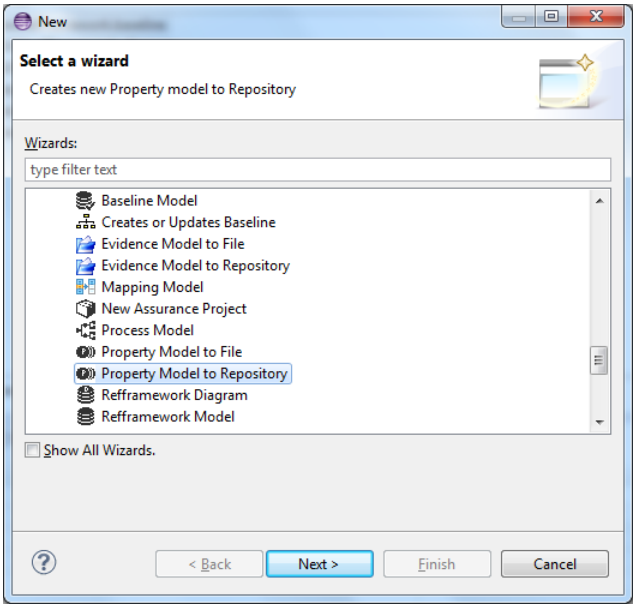


Figure 266 - New Property Model I

- Enter o select the parent folder, the resource name and press the Next button:

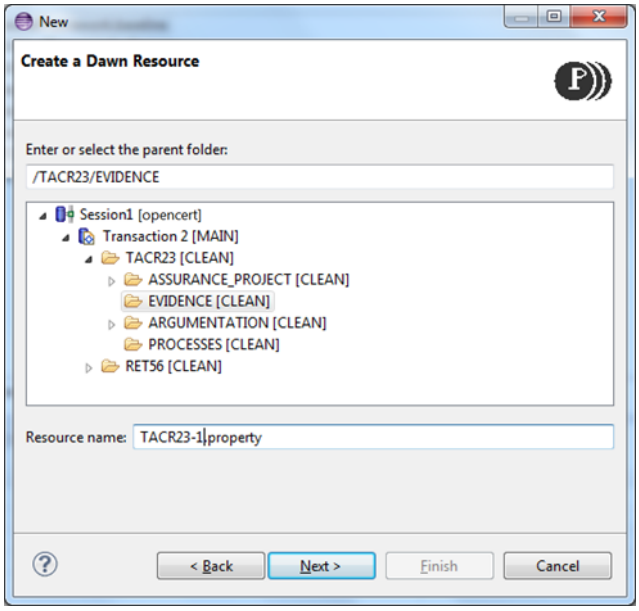


Figure 267 - New Property Model II

- And finally, select the “Model” object to create.

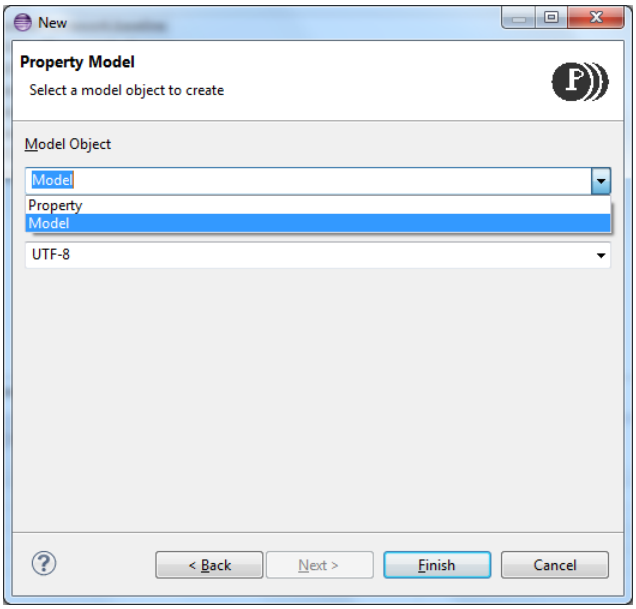


Figure 268 - New Property Model III

Once the Property Model has been created, the first item is presented to the user.

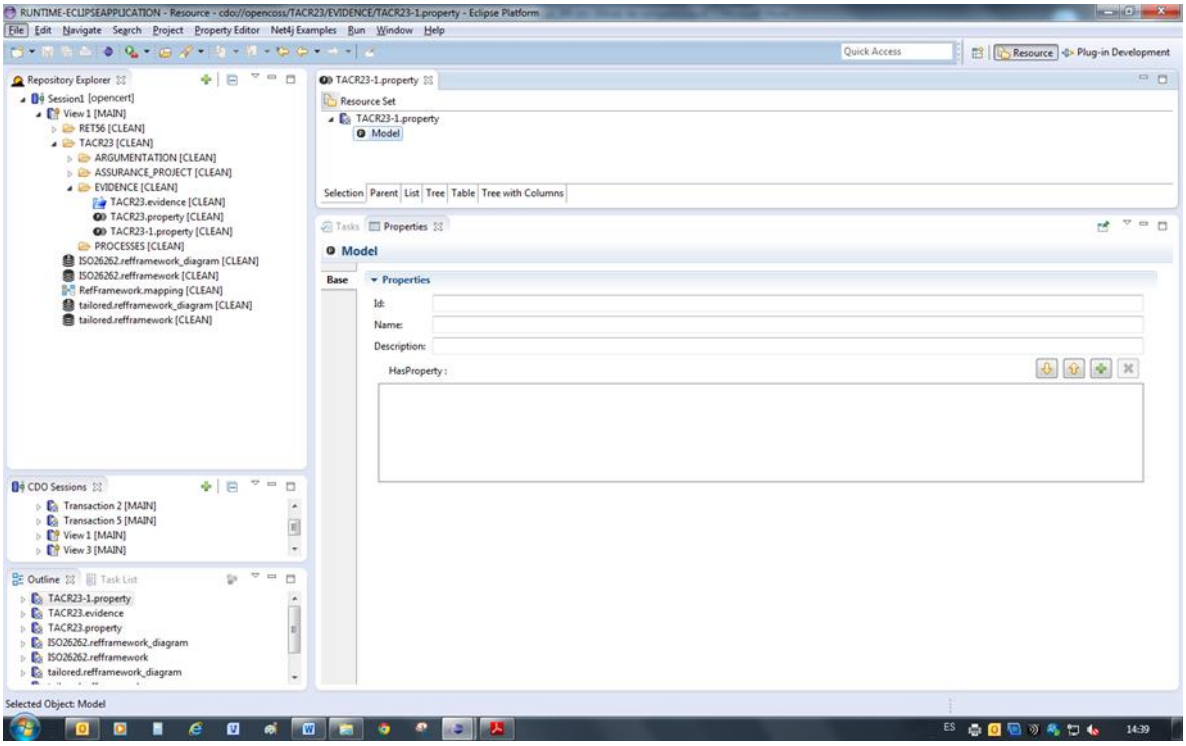


Figure 269 - Property Model

9.13 Edition of Properties

9.13.1Add a property

It is possible to add properties to a property model in two ways:

- Select the model element, press the right button of the mouse and select the contextual menu *New Child –> Property*

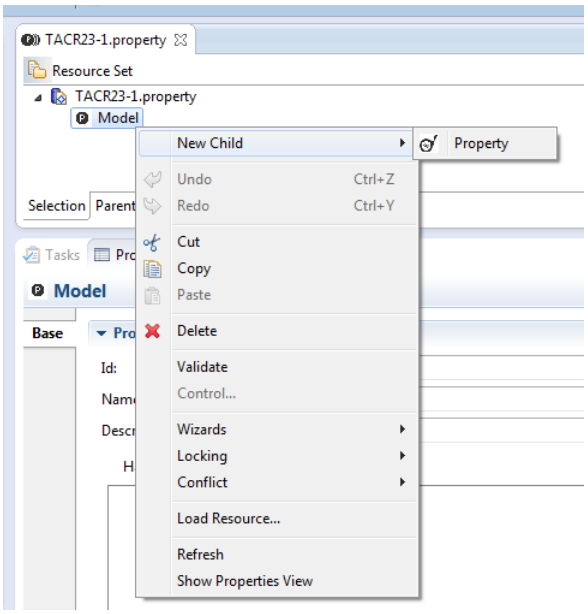


Figure 270 - Add New Property (I)

- Or, select the model element, and press the icon button *in the base tab*

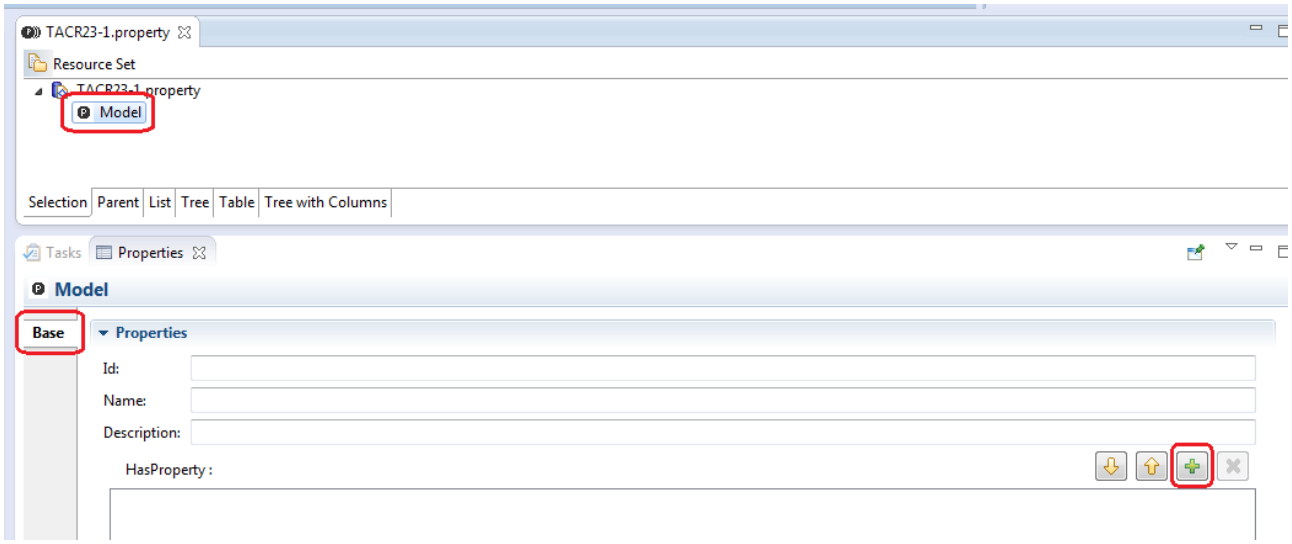


Figure 271 - Add New Property (II)

After these actions, in the properties zone, the framework presents several fields to describe the new property:

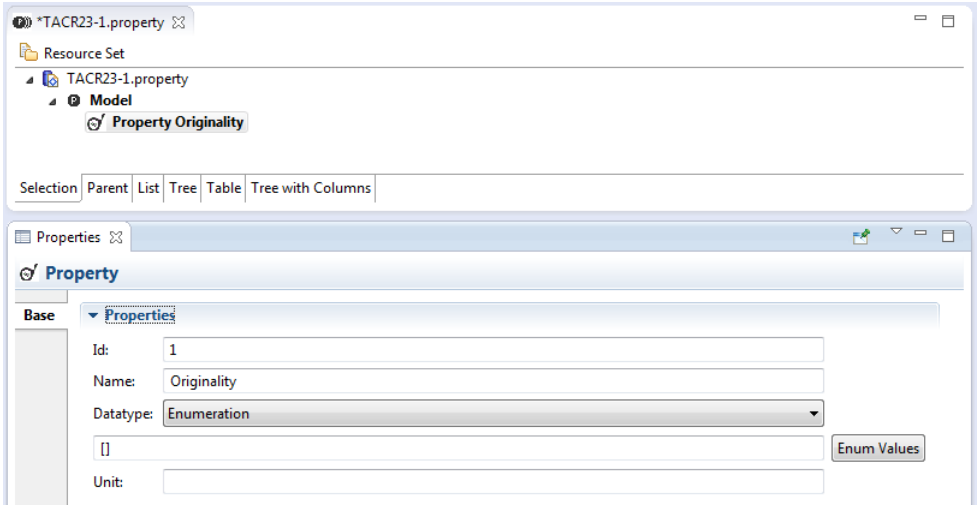


Figure 272 - Property properties

- Id: Property identifier.
- Name: Property name.
- Datatype: Property data type. Possible values: enumeration, string, integer and float.
- Enum values: values of an enumeration data type property. To add this values, press the button "Enum value"

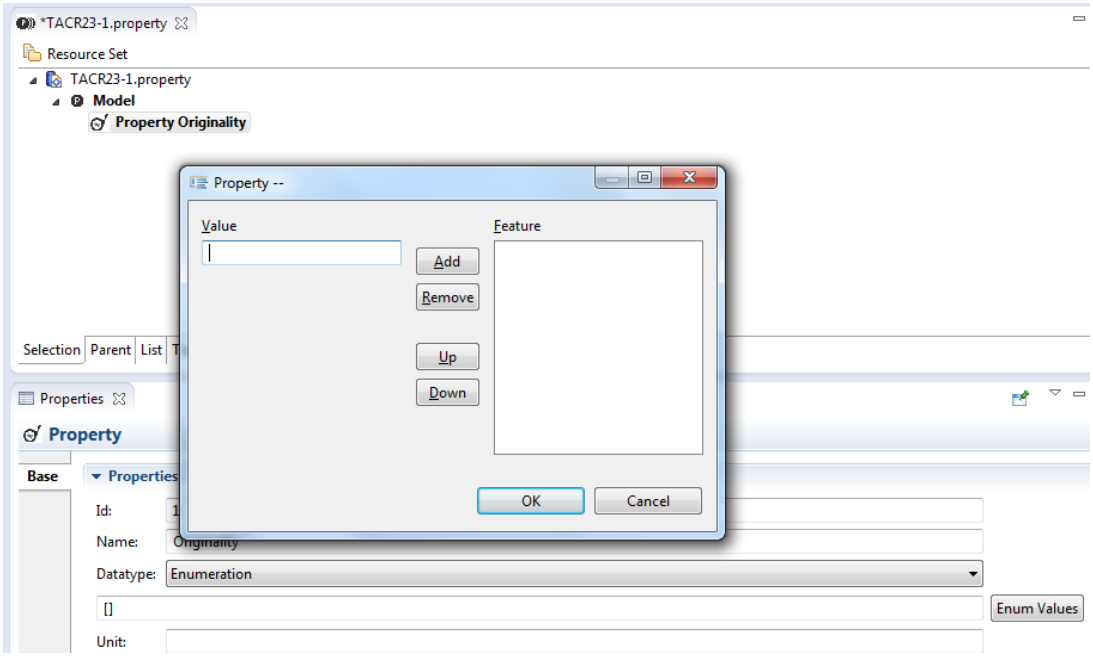


Figure 273 - Add enum values.

- Unit: unit value

9.13.2Delete a property.

To delete a property:

- Select the property, press the right mouse button and select the contextual menu *Delete*.

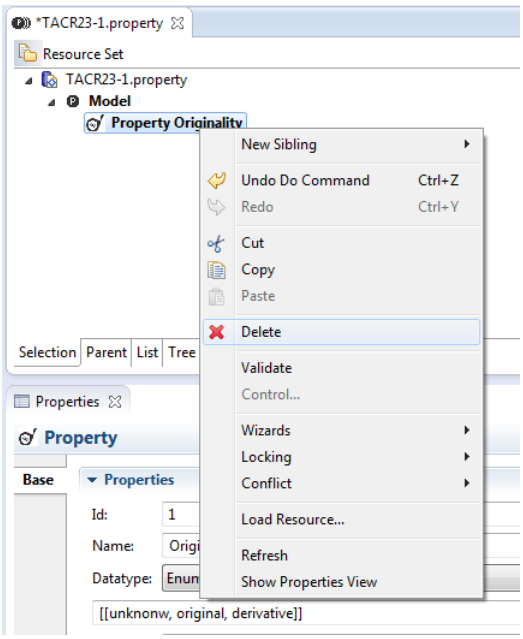


Figure 274 - Delete Property I.

- Or select the branch *Model that contains the property to delete, select the property and press the icon button*

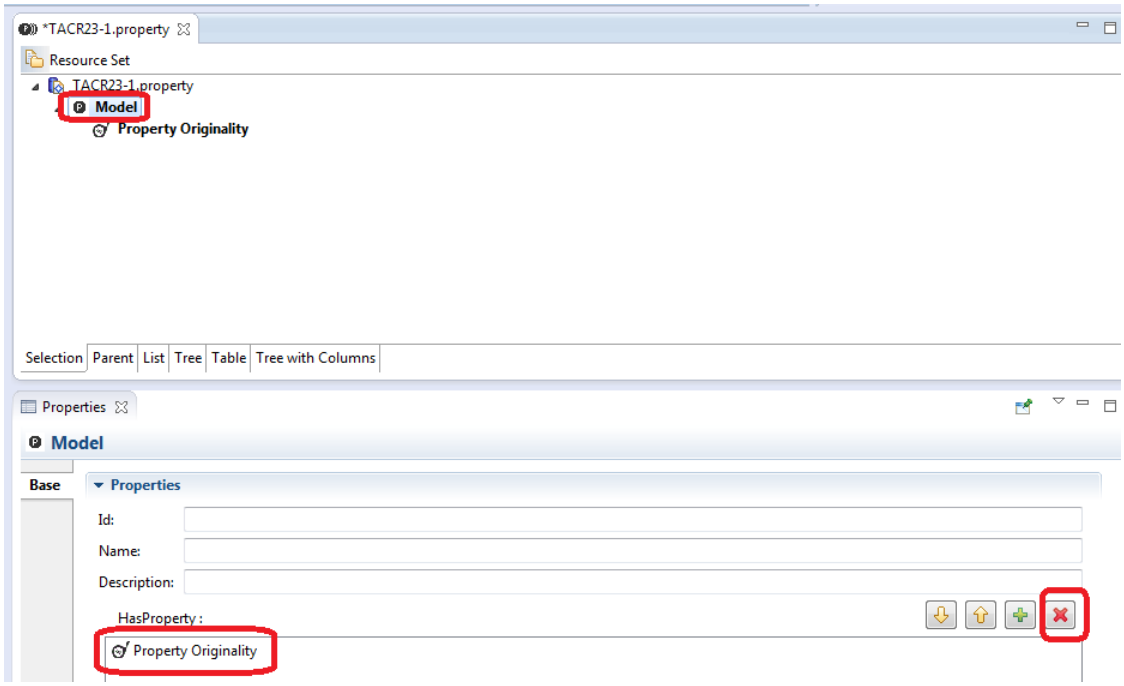


Figure 275 - Delete Property II.

10 OpenCert Web Client

As described in "**¡Error! No se encuentra el origen de la referencia.**" chapter, OpenCert tools consist of:

- OpenCert server - installed in a central host machine.
- One or many OpenCert clients - each of which installed on specific user machines.

The role of the central OpenCert server is threefold:

- It hosts CDO server, which facilitates a **common storage** for OpenCert server applications and OpenCert clients.
- It provides web interface with **OpenCert reports** presenting common storage assurance data from different angles
- It hosts **OpenCert API** services, e.g. evidence service or process service.

This chapter describes functionality provided by OpenCert server web reports - which facilitate a OpenCert server front-end for AMASS platform users.

10.1 OpenCert Web interface layout

OpenCert server web pages are served at 8080 port by default. If the navigation is done behind a proxy the port 8080 shall be open.
In order to view the web pages, please run your web browser and go to the following location:

http://<AMASS-SERVER-HOST-NAME>:8080/

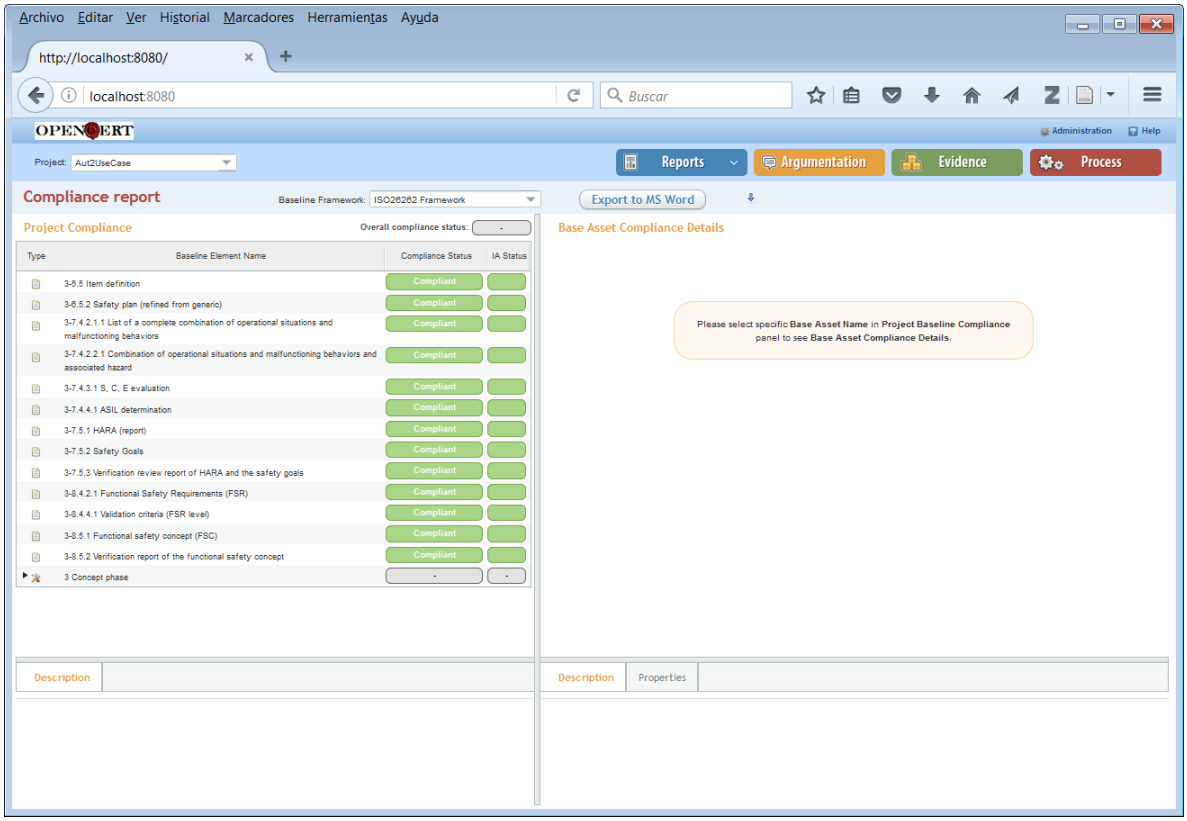


Figure 276 - Web interface layout

A typical OpenCert server web page consists of the following panels:

- Top panel
  It contains links to User Manual documents and links to server administration pages.
- Project and Menu panel
  It contains:
  - A select box with assurance projects which have been created in OpenCert platform.
  - Main menu with links to OpenCert server reports.
    There are several reports presenting analytical view from assurance data stored in OpenCert platform.
    Each of the reports is described in the subsequent chapters.
- Main panel
  It presents the main content of the page - depending on the current report or page selected from the menu and the given assurance project.

10.2 Compliance report

10.2.1 Goal of the report

Compliance report provides extensive functionality which helps OpenCert platform users to assess the current **compliance** of their project **to the selected safety standard** (i.e., baseline).
The functionality is intended to be used by:

- Project **team members**, for example developers, when the project is in progress, in order to have up-to-date insights into which of the baseline framework items are already satisfied and to what extent.
- Project **safety manager** in order to monitor the project general compliance, observe the compliance details and add, assign, or un-assign specific evidence resources to/from the given requirement of the safety standard which is followed by the project.
- Independent **safety assessor**, when the project draws to an end, in order to browse the assigned safety evidence, evaluate it and independently assess the actual project compliance to the specific safety standard.

Two modes of the report can be distinguished:

- An **interactive mode**, where user can actively browse the report, select the specific baseline items, view their properties, their compliance mapping, and the associated evidence, and add or remove the evidence resources mapped to the specific baseline element.
- A **printer** friendly **report** - which is a textual output presenting all the information of the current compliance of the selected project.

The compliance report can be accessed via the following OpenCert web server menu item.

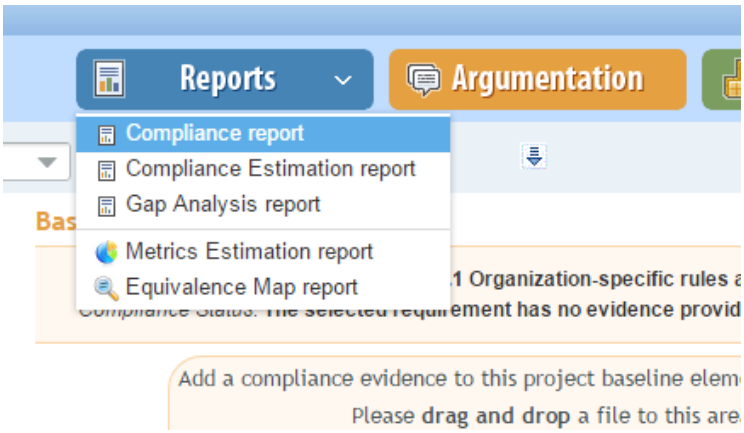


Figure 277 - Menu item directing to "Compliance report"

10.2.2 Viewing compliance data on the report

The compliance report allow users to see the overall compliance of the selected project to the specific safety standard.
When a specific OpenCert assurance project is selected in the top panel, its defined baselines are presented in the middle panel select box.

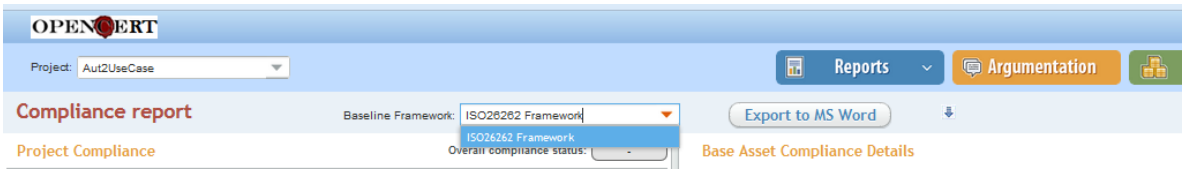


Figure 278 - Baseline Frameworks combo box for the specific project

The report data section is divided into 4 panels.

Figure 279 - 4 panels of "Compliance report"

The "**Project Compliance**" table, which is placed in the left, presents base artefacts and base activities of the selected safety standard. The most important column is the "**Compliance Status**" one, which presents the overall compliance status of a project to the specific safety standard item. The column can be sorted by value, thus allowing user to assess the project compliance at one glance.
In case base activities or base artefacts are defined to have a parent-child hierarchy, this relation is presented accordingly in a tree structure of the table.

Note:

"IA Status" column presents the current status of specific baseline element from Impact Analysis point of view. This functionality has been described in a separate chapter: Impact Analysis result presentation on OpenCert server reports.

When a specific baseline element item (i.e. table row) is selected, its description and properties, as defined in OpenCert storage, are presented in the bottom-left panel.

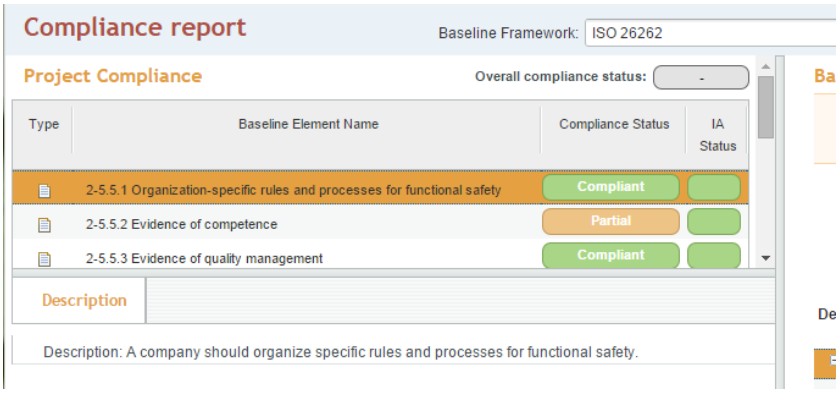


Figure 280 - Description of the selected baseline element presented at the bottom panel

Upon the selection of the specific safety standard item in the "**Project Compliance**" table on the left of the screen, the compliance mapping details are presented in the "**Base Asset Compliance Details**" panel at the right side of the page.

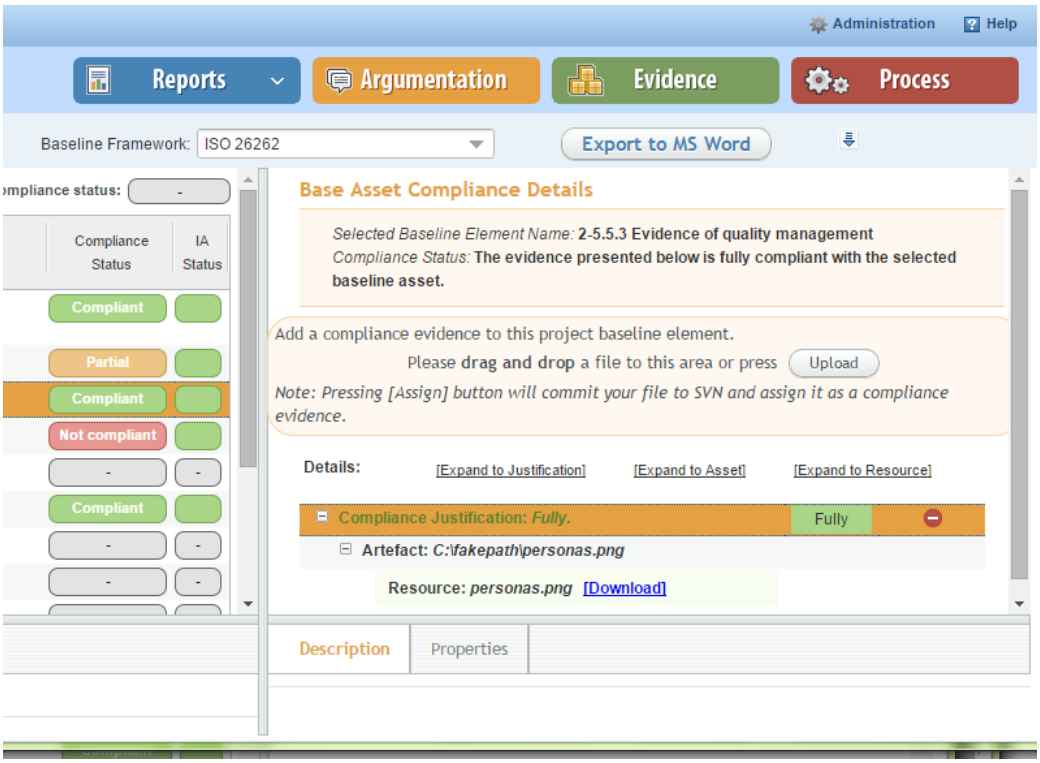


Figure 281 - Details of Justification and mapped evidence

The extensive compliance information is presented, including:

- **Compliance justification** explanation (as specified in OpenCert client editor or on this report).

- For the specific justification: the **associated artefact** or **activity**.
- For the specific artefact, its associated evidence **resource files**. These resource files are committed to the appropriate **SVN repository**. Users can press the ***[Download]*** link next to each resource tree node in order to download the specific file from the SVN and view it.

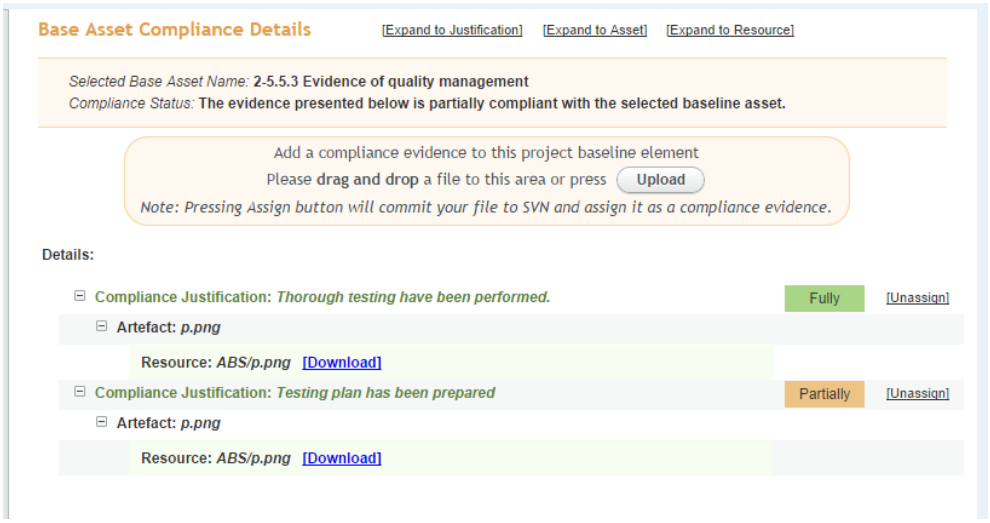


Figure 282 - Compliance evidence of the specific baseline asset

The above tree can be expanded or collapsed quickly to the desired level by pressing buttons above it, allowing tailoring the presented details to the level needed by a user at a given moment. When any of the above tree levels is selected (justification, artefact, or resource), its description and properties are presented in the right-bottom panel of the report.

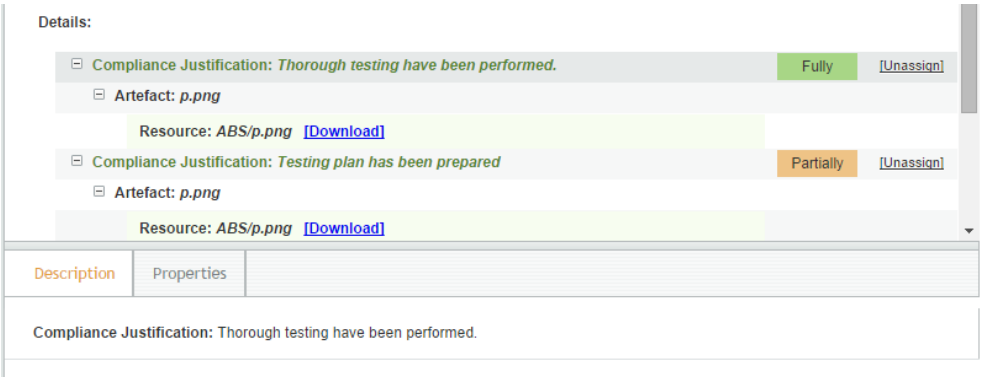


Figure 283 - Specific evidence details description presented at the bottom

10.2.3 Adding evidence and compliance data

Additionally to browsing the project evidence pieces, the report allows users to **add, modify and remove evidence** resources and **define a compliance mapping**.

"**Base Asset Compliance Details**" panel on the right-hand side of the report contains an **Upload** panel which allows users to add a specific file resource (containing the evidence), specify the associated artefact and define compliance justification text and its type.

After user presses the upload button or drag and drops the file resource to the panel, the following "**New Resource Definition**" dialog appears:

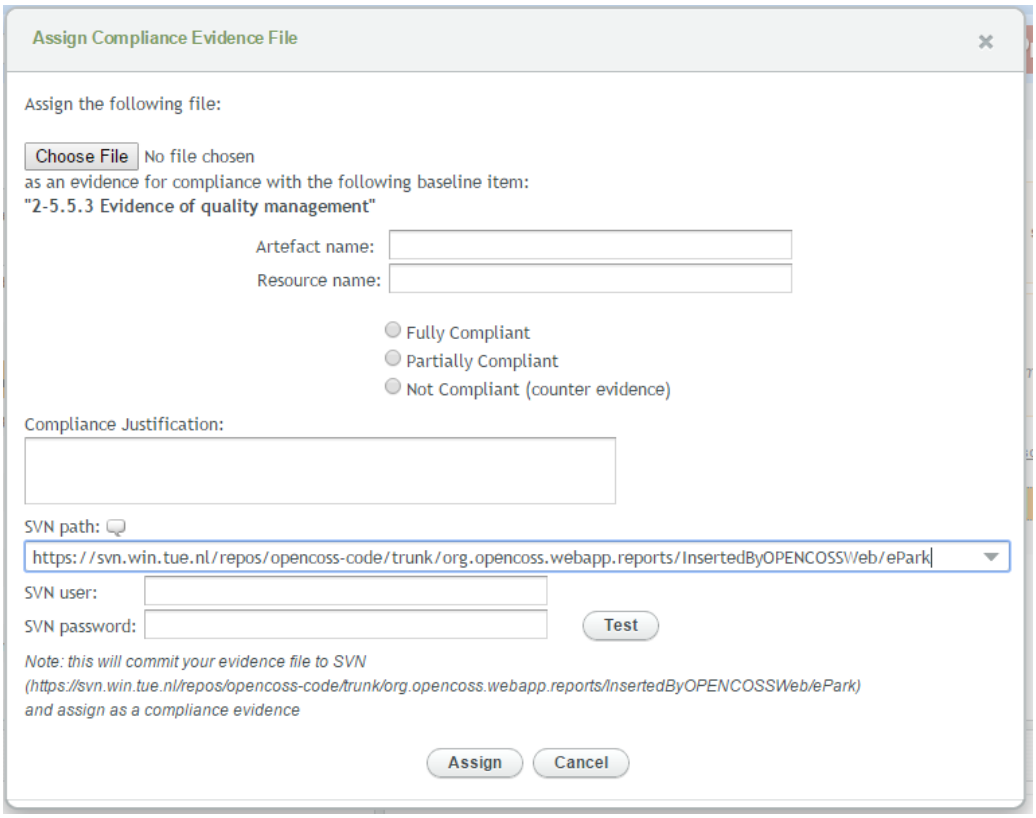


Figure 284 - A window allowing to assign and describe evidence to the given baseline item

User can enter the desired **compliance justification** in the text area, change the names to be created (default names are suggested) and define the **compliance mapping type**.
Additionally it is possible to specify a **SVN URL** location where the evidence file will be committed. User has a possibility to add new location or select already defined one from the select-box.
After pressing **Assign** button, the following actions are performed by the OpenCert platform:

- The resource file gets **committed to** the given **SVN** repository so that it is securely stored and can be retrieved on demand.
- A **resource** CCL object (associated with the above file) is created with the specific name.
- An **artefact** CCL object (associated with the specific resource) gets created.
- A **compliance justification**, which maps the artefact to the selected baseline framework items, gets created in OpenCert storage.

Additionally, there are [**Modify**] and [**Unassign**] buttons, which allow user to update or revoke the evidence file and compliance mapping created above.

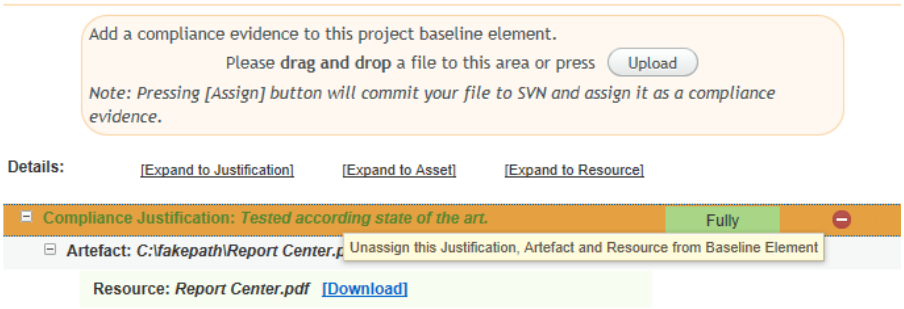


Figure 285 - Unassign button allowing to disassociate evidence from the given baseline item

10.2.4 Generation of summary textual report

The **interactive mode** presented in the preceding chapters is very comfortable for users to browse and filter data, and view their details. Upon each user selection, appropriate details are presented.
However, there is often a need to **generate an overall report**, containing **all the information** visualized in one place.
This can be easily done using **Export to MS Word** button.
Upon pressing it, a default an docx template report gets filled with the **all the Compliance report data** presented for the specific safety project.

Note for OpenCert administrators:
The template docx used for textual report generation can be changed on OpenCert server side in order to adjust it the given company standards.

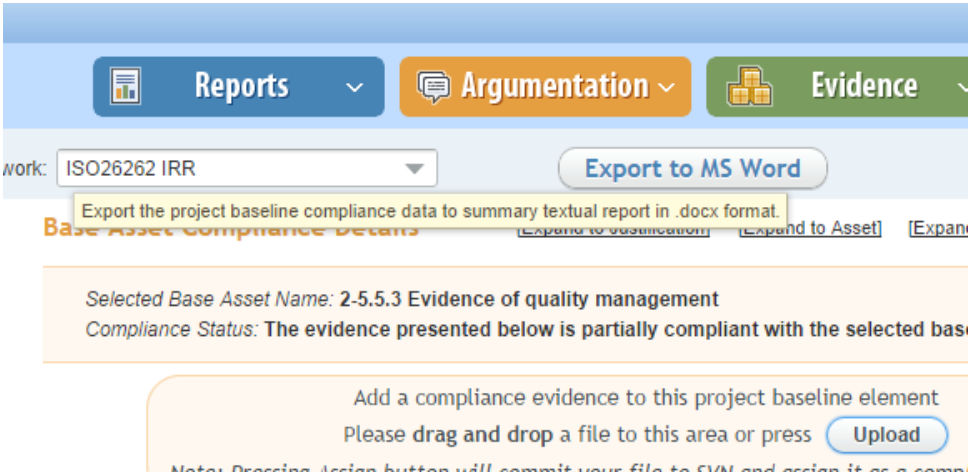
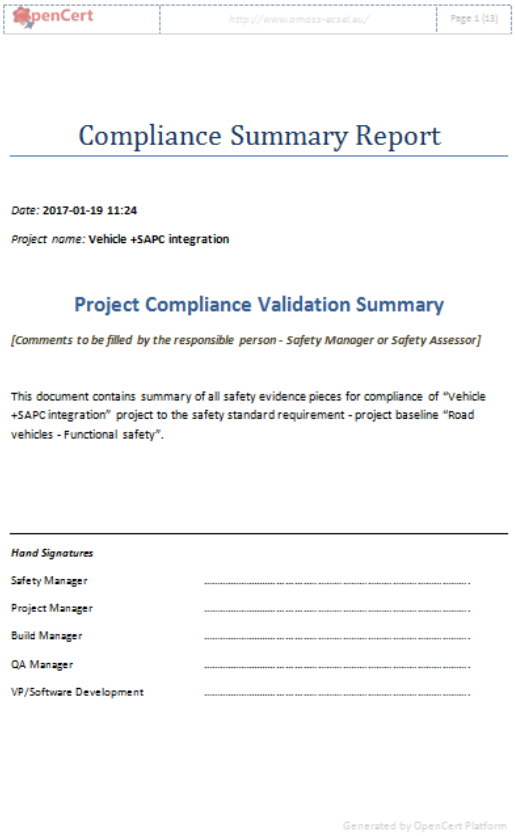


Figure 286 - "Export to MS Word" button which generates textual overall detailed report of Project Compliance to the safety standard

The textual report can then be **printed to pdf or on paper**, signed digitally or manually, and stored for future reference.


OpenCert http://www.amass-ecsel.eu/ Page 1 (13)

Compliance Summary Report

Date: **2017-01-19 11:24**

Project name: **Vehicle +SAPC integration**

Project Compliance Validation Summary

[Comments to be filled by the responsible person - Safety Manager or Safety Assessor]

This document contains summary of all safety evidence pieces for compliance of "Vehicle +SAPC integration" project to the safety standard requirement - project baseline "Road vehicles - Functional safety".

Hand Signatures

Safety Manager .....

Project Manager .....

Build Manager .....

QA Manager .....

VP/Software Development .....

Generated by OpenCert Platform

Figure 287 - First page of the generated textual report

10.3 Change Impact Analysis

OpenCert platform facilitates a prototype of Change Impact Analysis algorithm implementation.
It is triggered when any artefact stored in OpenCert database is modified by the user. The algorithm traverses all the related artefacts in order to check if they should be marked as affected by the change, depending on the artefact relation type.

10.3.1 Change Impact Analysis in OpenCert Client

OpenCert Eclipse client editor is the main OpenCert tool where user adds, modifies, removes artefacts and relations between them. Similarly it is the place where Impact Analysis algorithm is triggered when any artefact is changed. Similarly the IA results are presented there.
This has been described in Impact analysis section.

10.3.2 Change Impact Analysis algorithm

This chapter presents technical details describing how IA algorithm traverses relations between artefacts.

The main pieces of information used by the IA engine are relations between Artefacts objects stored in *ArtefactRel* CCL entity.

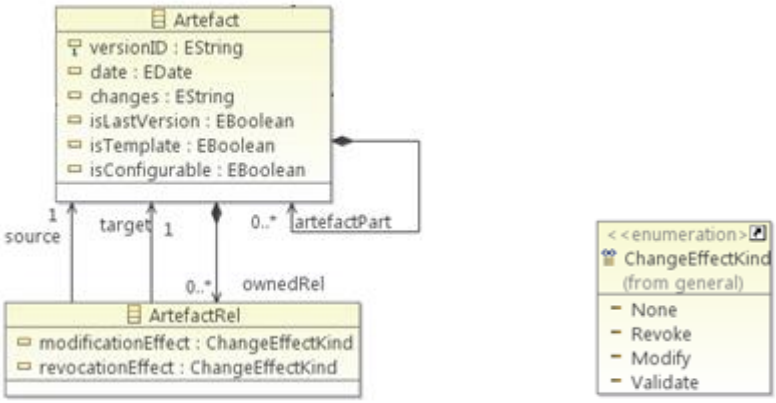


Figure 288 - Artefact Model

Two Artefacts are considered related when there is an *ArtefactRel* instance pointing to one of them as a source and another of them as a target. Please note that *ArtefactRel* has *modificationEffect* and *revocationEffect* attributes.

Note: An ArtefactRel object for specific two artefacts can be added in the following ways:

- A user can add this entity manually in the Evidence Editor of OpenCert platform client
- ArtefactRel entity is added automatically when a parent-child relation is established between two artefacts. When adding artefactPart to parentArtefact, a new ArtefactRel object is created, with modificationEffect=MODIFY and revocationEffect=MODIFY, *source* pointing to parentArtefact and *target* to artefactPart.

It has been arranged that a direction of analysis flow is the following: *ArtefactRel* "*target* affects the *source*". When impact analysis is started:

- It starts from *artefactCDOId* for the specific *EventKind* (either *Modify* or *Revoke*),
- It looks into the related *ArtefactRel* (for which the *artefactCDOId* is a *target*) object
- It traverses to the artefact pointed by *ArtefactRel source*
- Depending on the initial *EventKind* (either *Modify* or *Revoke*), it takes the value of *modificationEffect* or *revocationEffect* from the *ArtefactRel* and assumes the appropriate *AssuranceAssetEvent* on the reached source artefact.

For example, let's assume that there are the following *Artefact* and *ArtefactRel* dependencies:
ArtefactA ---- ArtefactRelA(ModificationEffect:MODIFY, RevocationEffect:REVOKE) ---> ArtefactB
ArtefactB ---- ArtefactRelB(ModificationEffect:REVOKE, RevocationEffect:REVOKE) ---> ArtefactC
ArtefactC ---- ArtefactRelC(ModificationEffect:MODIFY, RevocationEffect:VALIDATE) ---> ArtefactD
ArtefactD ---- ArtefactRelD(ModificationEffect:MODIFY, RevocationEffect:REVOKE) ---> ArtefactE

- The engine starts with *EventKind.MODIFICATION* for *ArtefactA* and navigates via *ArtefactRelA* to *ArtefactB* and because *ArtefactRelA:ModificationEffect* equals *MODIFY*, it reaches *ArtefactB* with *EventKind.MODIFICATION* change effect.

Note: this change effect event is not saved in storage yet. Now it is only used for further traversal, and will be returned as part of the result of *listArtefactsRelationImpacts()* method.

- The engine continues from ArtefactB with EventKind.MODIFICATION and navigates to ArtefactC and because *ArtefactRelB:ModificationEffect* equals REVOKE, it reaches *ArtefactC* with *EventKind.REVOKE* change effect.
- The engine continues from *ArtefactC* with *EventKind.REVOCATION* and traversal path ends here because of *ArtefactRelC:RevocationEffect* equals *VALIDATE*.
- Thus the result is:

- *ArtefactRelA - ModificationEffect.MODIFY*
- *ArtefactRelB - ModificationEffect.REVOKE*

Above algorithm affects the Artefact lifecycle. States of this lifecycle are presented on the figure below.

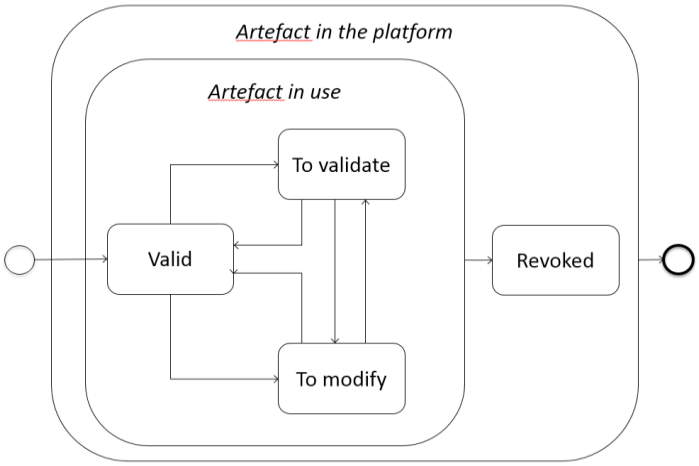


Figure 289 - Artefact lifecycle from the IA point of view

Some of these states require action from user – like "To validate" and "To modify". To address some restrictions of CCL these two states are recognized by the presence of given event date or lack of it. This signals that action from the user is required and after this action date of the event is set.

10.3.3 Impact Analysis result presentation on OpenCert server reports

IA-induced user actions that need to be performed in the assurance project are presented on **Compliance Estimation Report** and **Compliance Report**.
These reports are described in **¡Error! No se encuentra el origen de la referencia.** and Compliance report in general.
Hereafter is a description of **Impact Analysis** results being presented on these reports.

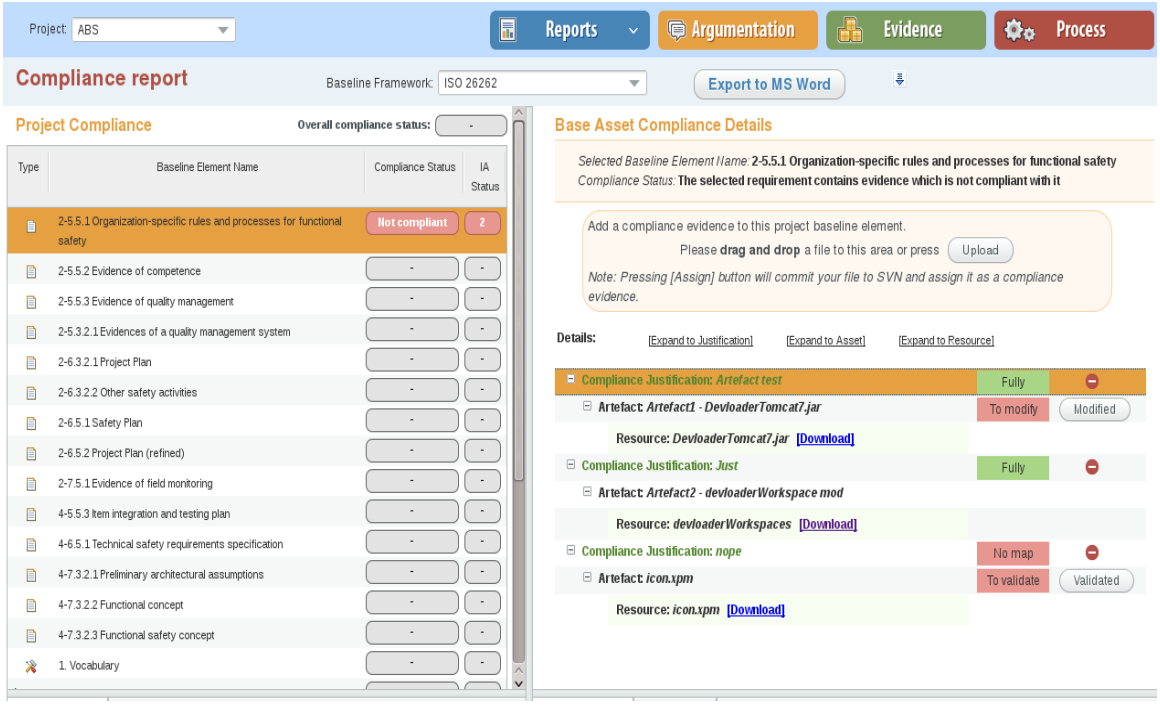


Figure 290 - Web interface showing two IA-induced actions required to be taken by user

In "**Project Compliance**" table on the left panel, "**IA Status**" column presents the status of the specific baseline element from **Impact Analysis** point of view. The following information is presented:

- Grey color means that there is no artefact compliant to this baseline element thus there is no entity on which AI can work.
- Red color means that some of the artefacts compliant with this baseline element where affected by AI and they require attention from the user. The displayed number represents the amount of such affected artefacts.
- Green color means that there is no action required by IA from user after IA traversed the associated artefacts.

On "**Base Asset Compliance Details**" panel the compliant artefacts and their details are presented.
The information also includes **IA results** in case when AI algorithm detected that some action need to be performed by a user.
As it regards the above screenshot, IA execution resulted in detection of two actions required by the user: "**To validate**" and "**To modify**". This information is presented next to the respective artefacts.
When the user takes the according measures (i.e. validates or modifies the respective artefacts) he can simply clicks "**Modified**" and "**Validated**" action buttons to report that the requested activity has been performed.

10.4 Gap Analysis report - Compliance Assessment and Evidence Evaluation

Gap Analysis report facilitates the following pieces of functionality:

- Compliance Assessment by viewing a Gap Analysis
- Viewing Evidence Evaluation results

In order to view Gap Analysis report, please go to OpenCert platform web server page in your web browser at http://<AMASS-SERVER-HOST-NAME>:8080/**.** and select Reports > Gab Analysis report from menu.

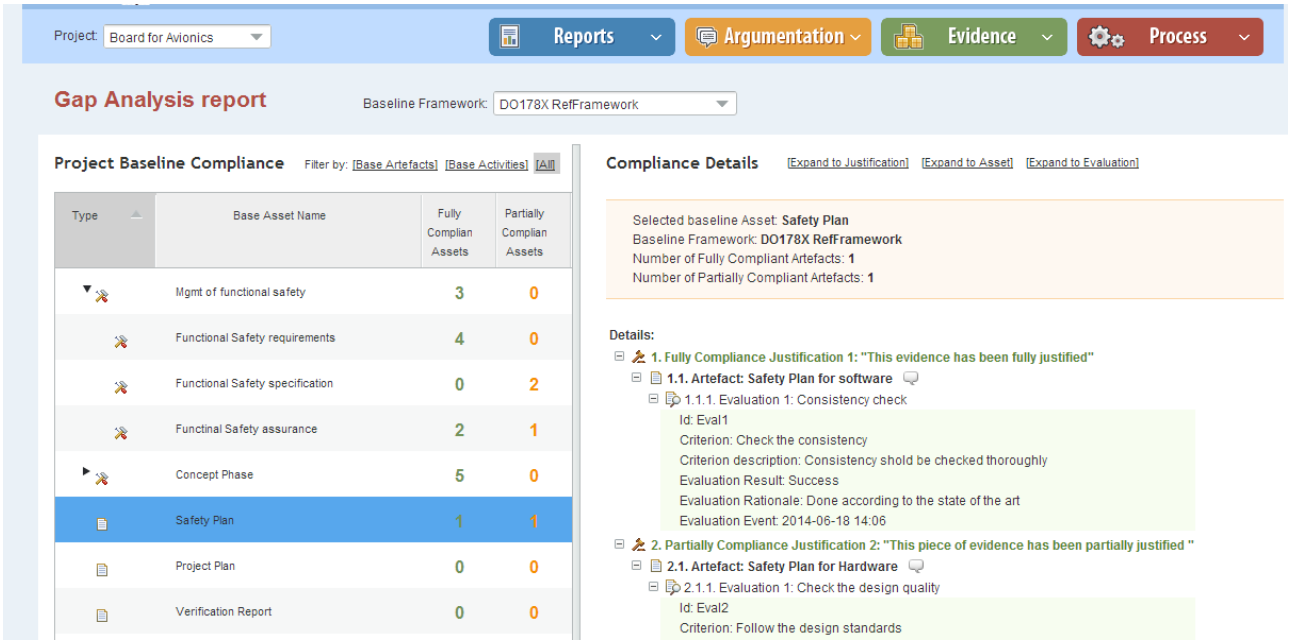


Figure 291 - Gap Analysis report

10.4.1 Gap Analysis report core functionality

Gap Analysis report presents summary and details regarding specific assurance project base artefacts and base activities and their compliance mapping to the actual evidence and activities.
For the selected assurance project its baseline frameworks are presented in a select box.

Figure 292 - Baseline frameworks for the specific assurance project

After choosing the specific baseline framework, the following gap analysis data is presented:

- In a left pane, called “Project Baseline Compliance”, base artefacts and base activities of the selected project baseline framework are shown. They are displayed in a tree structure to express parent-child hierarchy of these items. For each base artefact or base activity the total numbers of fully- and partially- compliant assets are presented.
  The “Project Baseline Compliance” table can be filtered to show only base artefacts or base activities and can be sorted by any column.

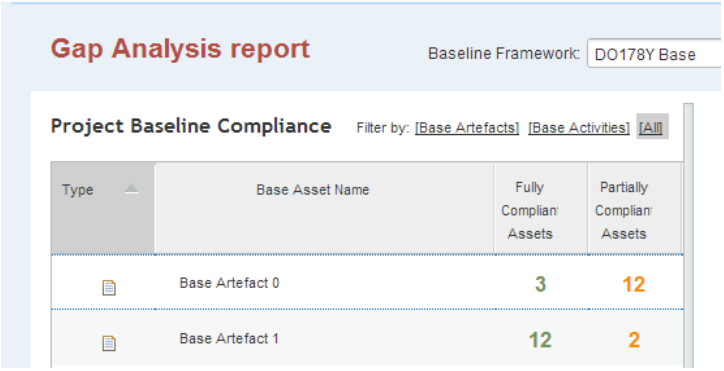


Figure 293 - Project baseline compliance table

- When user selects the specific cell in the left pane table (e.g. specific base artefact or the number of fully- or partially-compliant assets), the right panel is refreshed with the details regarding the selected item.
- For specific base artefact or base activity selected in the left panel, the right details panel presents the following information:
  - Summary of the specific base artefact or base activity compliance mapping
  - For the given base artefact and base activity: Compliance Justification elements from its compliance mapping
  - For each Compliance Justification: the actual assets i.e. artefacts or activities
  - For each artefact or activity: its description and properties are presented on the tooltip
  - For each artefact or activity: its evaluation is shown - in case the asset has been evaluated.

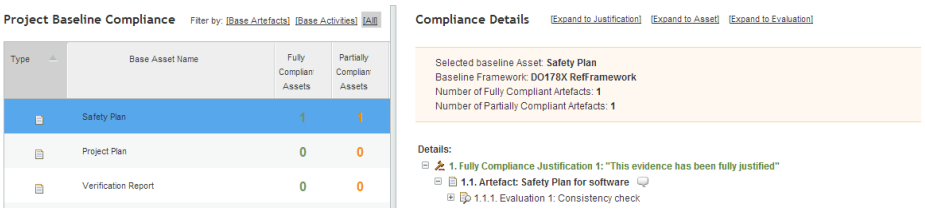


Figure 294 - Compliance details for the selected baseline element

The “Compliance Details” tree can be expanded to the specific level by pressing links at the top of the panel.

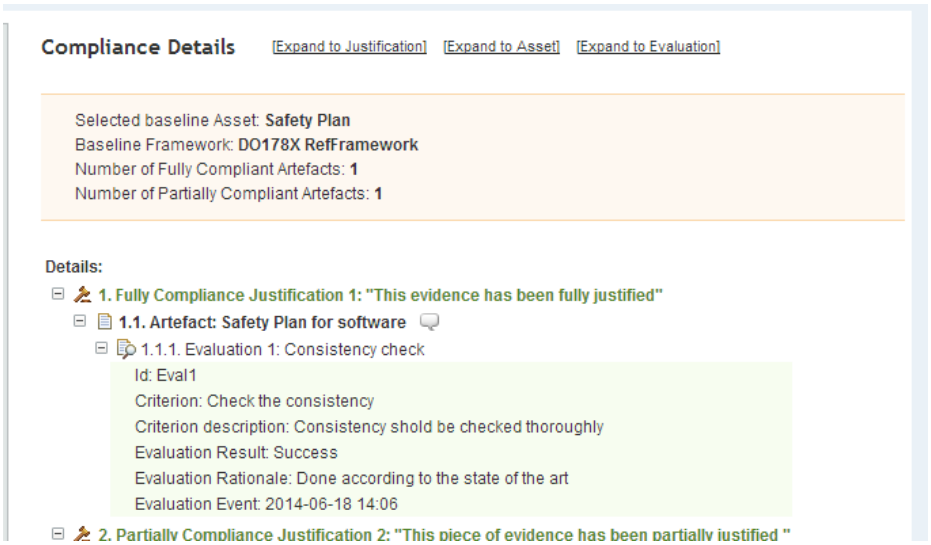


Figure 295 - Compliance details

For example, if user is interested only in justification or assets or their evaluation, he can *press [Expand to Justification]*, *[Expand To Asset]* or *[Expand to Evaluation]* links respectively. The details tree is expanded accordingly.

10.4.2 Viewing Evidence Evaluation in Gap Analysis report

In case a specific evidence item has been evaluated in OpenCert client tool, its evaluation data are shown on the Gap Analysis report.

All the evaluation properties (Id, Criterion, Evaluation result, etc.), and the evaluation date and time are presented.

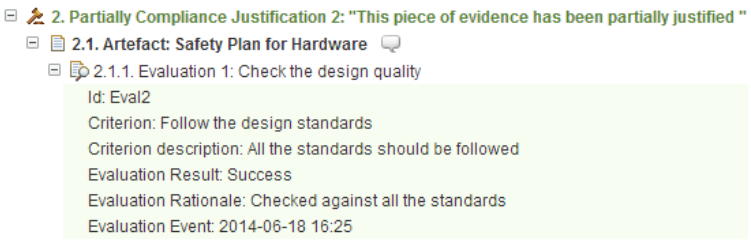


Figure 296 - Evidence evaluation details

10.5 Metrics reports

This chapter presents the implemented metric reports, their functionality and layout.

10.5.1 Metrics Estimation Report

The Metrics Estimation Report can be accessed via the following OpenCert web server menu item:

Figure 297 - Menu item directing to "Metrics Estimation report"

When a specific OpenCert safety project is selected in the top panel, its defined baselines are presented in the middle panel select box, as shown in Figure 211.
The report data is divided into two panels. The first one, on the left, is a static menu panel in which the user can select a type of metrics to analyse.

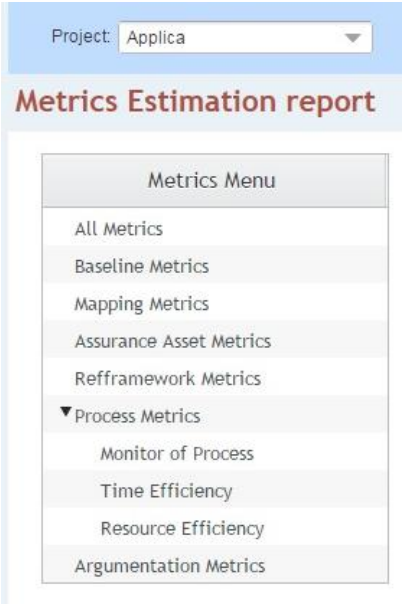


Figure 298 - Metrics Menu in the top-left portion of the report

Once a specific metrics is selected on the Metrics menu, the metrics menu details are presented in the right part of the report with a description of the main goal and all the different types of charts related to that metric.
Also, the user has the option to the selected metric to a Word Document with more detailed information.
As a small example, the following figure:

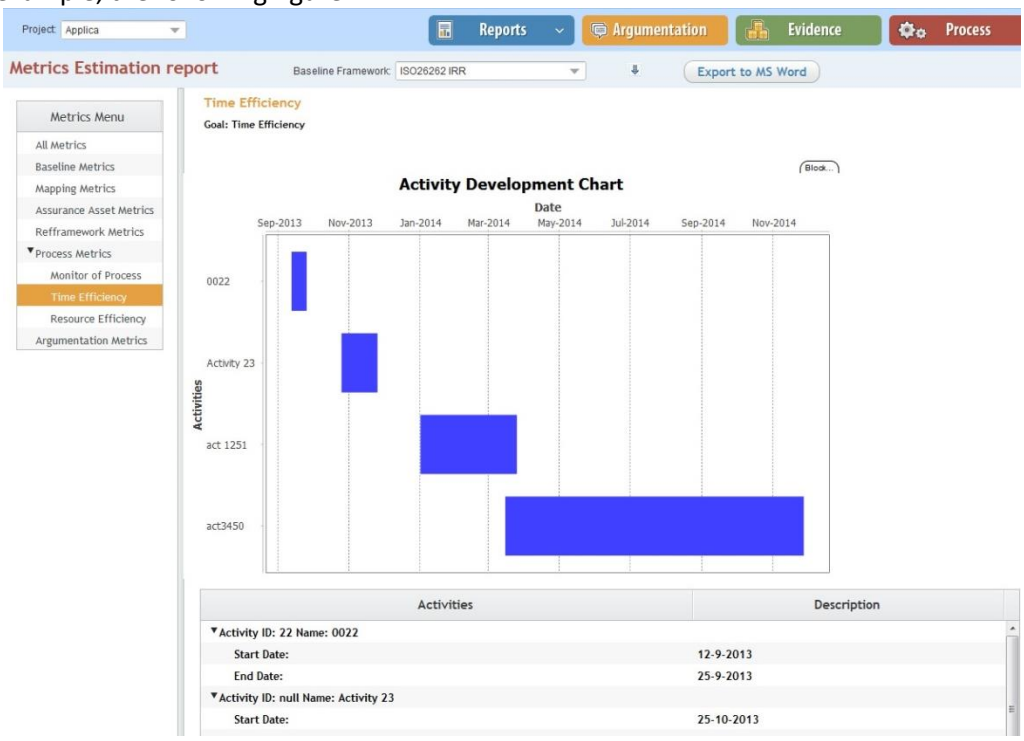


Figure 299 - Description of the selected metric type presented at the left

10.5.2 Equivalence Map Report

The Equivalence Map Report can be accessed via the following OpenCert web server menu item as showed beforehand.

In this case the equivalence metrics are between two Reference frameworks, not specific OpenCert safety project needs to be selected in the top panel. The only possible configurations are between reference frameworks as shown in figure below

Figure 300 - Selection of reference frameworks

After the selection, the metrics of the equivalence maps and a detailed description are showed on the screen. There is also a possibility to export this information to a document.

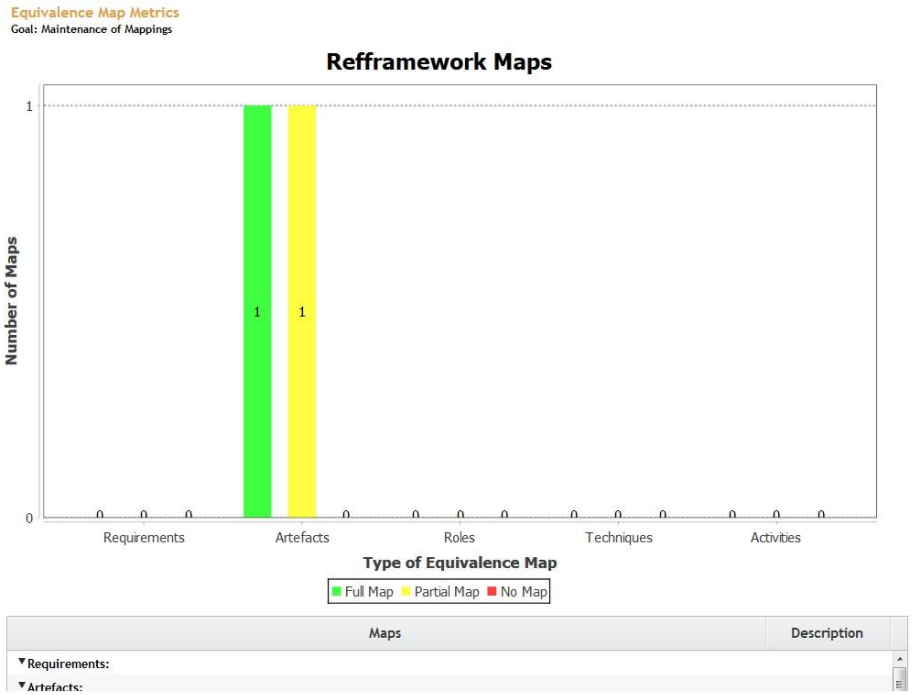


Figure 301 - Equivalence Map Report

10.6 Administration web GUI

OpenCert server web pages provide a few basic pages for server administration.
They are accessible through “Administration” menu.

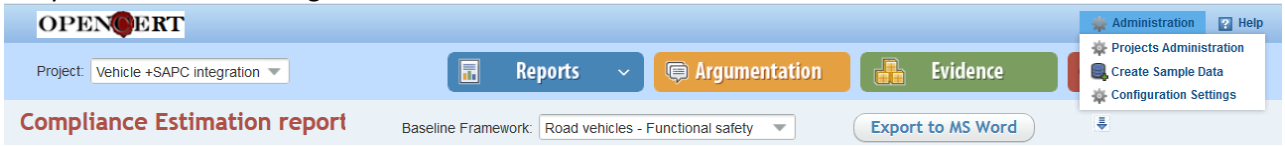


Figure 302 - Administration menu

10.6.1 Projects Administration

This administration page facilitates project editing basic actions.
For more advanced project editing functionality, OpenCert Eclipse Client Editor should be used.
The page allows the following basic actions:

- Project name and description editing.
- Project baseline name editing.
  Note that only one baseline of the project is presented.
- Project baseline element editing.
  User can add, remove and modify baseline elements.
  For the specific baseline element, its name and description can be specified.

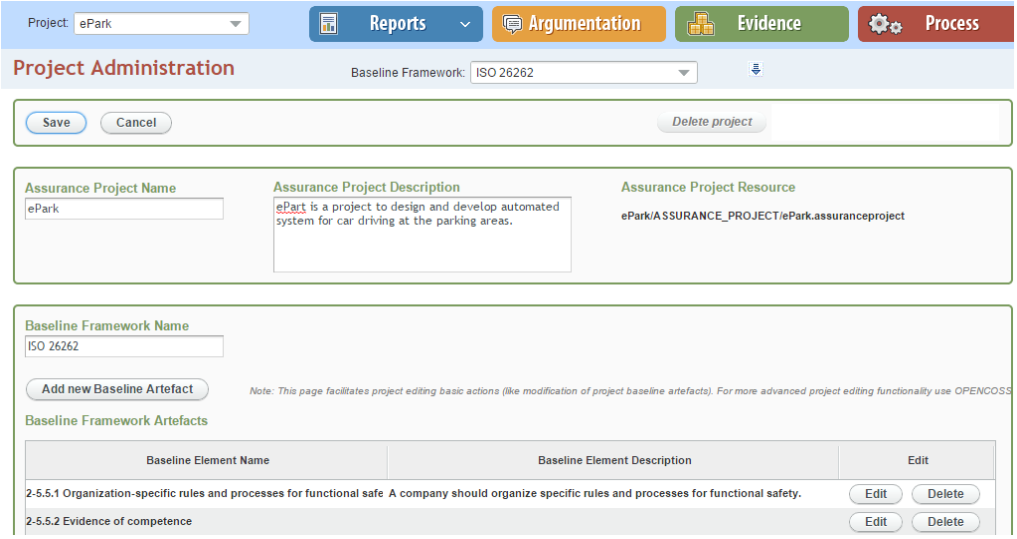


Figure 303 - Project Administration web page on OpenCert server

10.6.2 Create Sample Data

This OpenCert server administration web page provides a functionality, which allows user to generate sample data in the database.
The sample data can be generated for example in order to demonstrate Gap Analysis report.

OpenCert user can generate the following sample data for the selected assurance project:

- baseline framework
- base artefacts
- base activities
- artefacts and activities being mapped with compliance mapping to the base artefacts and base activities.

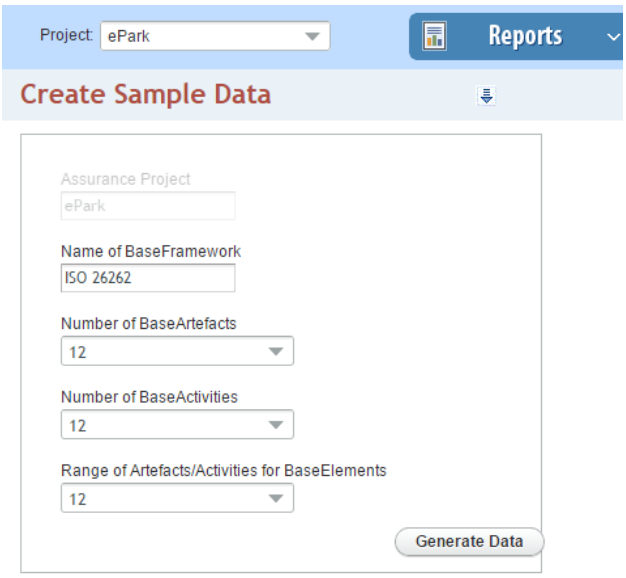


Figure 304 - Create sample data page

After pressing [**Generate Data**], sample data will be created and inserted into the selected assurance project.

10.6.3 Configuration Settings

This page presents the main configuration settings of the OpenCert server.
These settings are stored and can be modified on OpenCert server host, in *AMASS-properties.xml* file which is present in the operating system user home directory.

11 Specification of Families/Lines

11.1 Process Lines

11.1.1Importing Library and Fixing the Problems

The library contents are persisted in their own folders and XMI files, but the XMI files produced by the EPF-C cannot be directly opened, the UMA based exceptions have been raised. The problems with XMI files needed to be resolved for mapping the elements of a target configuration and variability abstractions in BVR. For this reason we have implemented a dialogue wizard, as shown in Figure 285, the recent/default path choice is automatically filled in the path text box otherwise the path containing a specific method library might be browsed. The dialogue wizard performs two tasks: (i) copies the contents of the method library in the target directory; and (ii) resolves problems with the XMI files. The error free models are made available in the project folder. All the model files can be opened, for example, method configurations, method plugins, method content descriptions and processes.

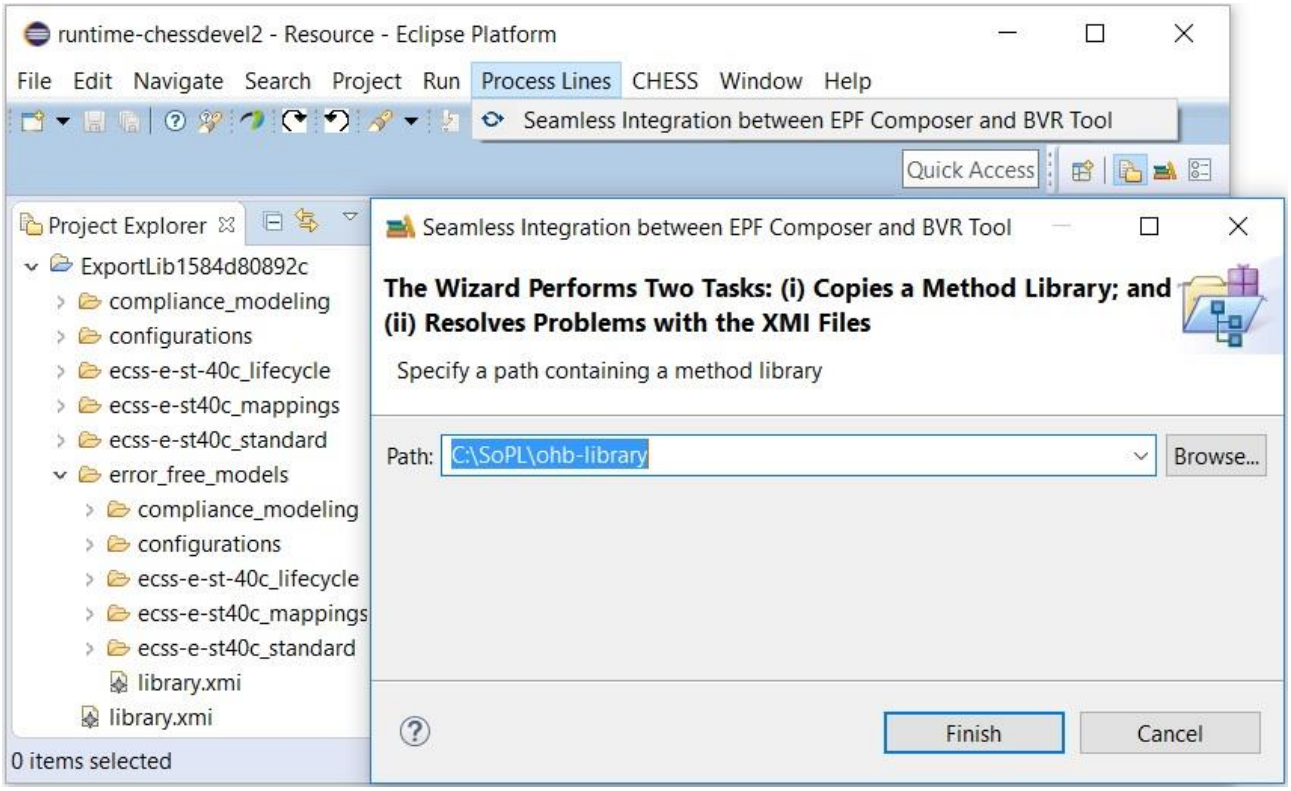


Figure 285: The achievement of error free models

11.1.2BVR VSpec and Resolution Editors

The tailoring rules provided in Annex R of ECSS-E-ST-40C are modeled within the VSpec editor. The resulting VSpec model, represented in Figure 286, shows the tree structure representing logical constraints to be considered during the resolution. As recalled in the background, the solid line indicates that the particular feature applies to all criticality levels, whereas the dashed line represents a variation point. The whole tree cannot be visualized due to space limitations; therefore the minimize option (+) is used for hiding the features. The tasks associated with *software integration test plan development*}, and *software units and software component integration and testing* are marked as optional. As the choices are associated with multiple criticality levels, the constraints have been applied. The multiplicity xor(1..1) is assigned to the criticality; therefore exactly one out of A, B, C and D must be selected for the software product. As per the specification in Annex R, four constraints are applied for the *Test_plan* and *Integration_ and_testing_software_units*.

The constraint *D implies (Test_plan and Integration_and_testing_software_units)* enforces that if criticality D is selected then both specified tasks must be included for tailoring purpose. Otherwise, validation error is produced. The constraint *(A or B) implies (not Test_plan)* indicates that the *Test_plan* must be excluded for processes with criticality A or B. Likewise, the constraint *((A or B) or C) implies (not Integration_and _testing_software_units)* enforces the exclusion of particular task for tailoring of processes with criticality A, B or C. The constraint *(C and Test_plan) implies ((not SUITP_K9) and (not SUITP_K10))* enforces inclusion of *Test_plan*, but also exclusion of *SUITP K.9* and *K10* for criticality C. The possible tailoring scenarios have been covered.

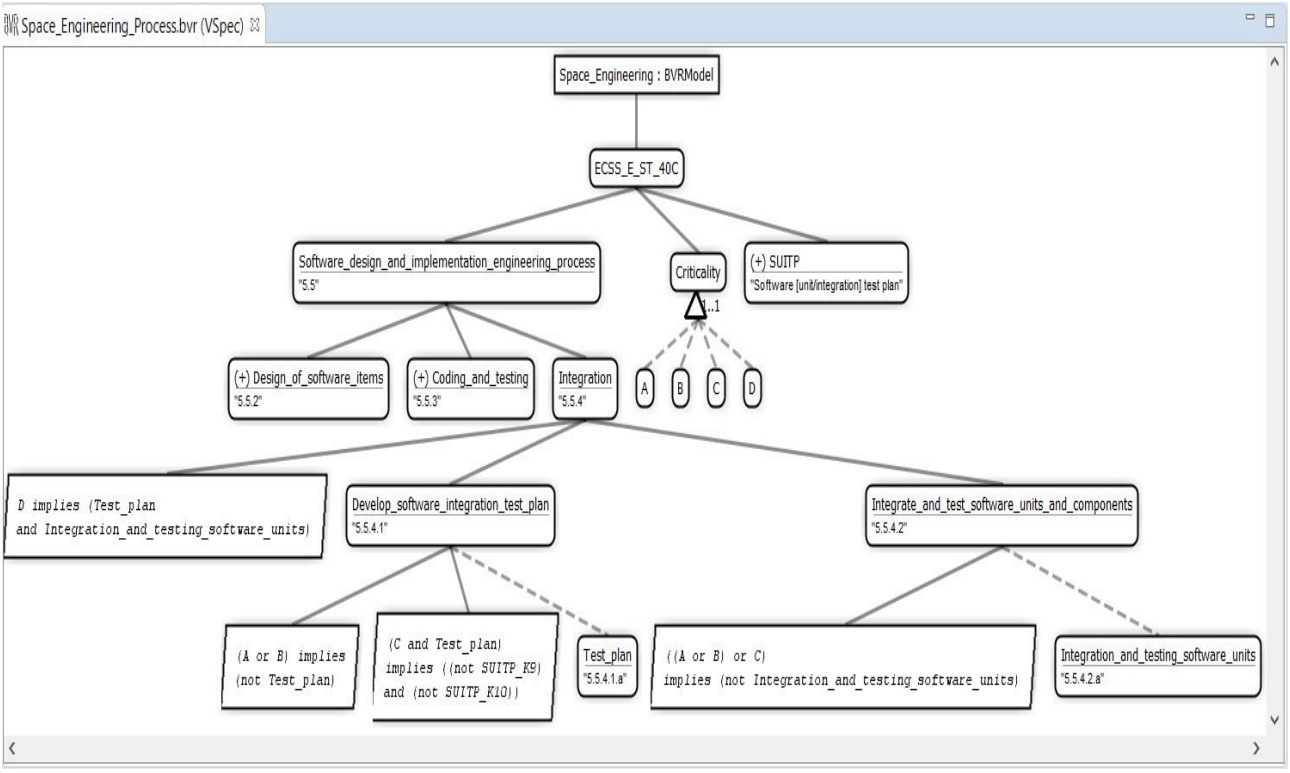


Figure 286: VSpec model

The resolution models are automatically generated from the VSpec model, but the choices needed to be included or excluded for individual processes, as illustrated in Figure 287. In this regard, multiple resolutions might be defined for the process with variability. The implemented editor supports error checking and validation of resolutions. Thus, valid tailoring is guaranteed if the constraints are properly specified.

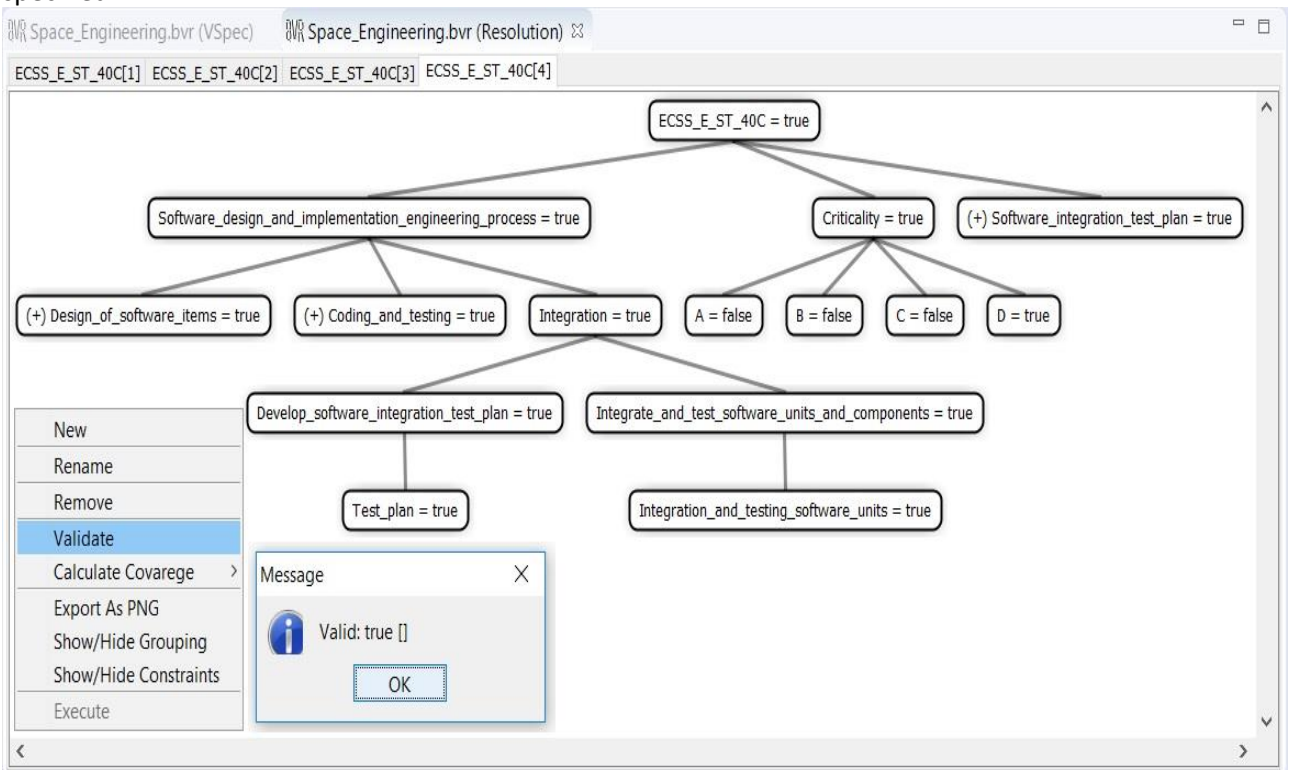


Figure 287: Resolution model

11.1.3Deriving the Processes and Export Back

The derivation process involves the substitutions in which elements of a placement fragment are removed and elements of a replacement are injected. To create the placements and replacements, the elements from the models are dragged and dropped on the realization editor, for which the method library, plugins, configurations, content descriptions and processes can be considered. It is therefore possible to define the placements and replacements between multiple models. The placements are visualized in red, as shown in Figure 288, while the replacements in blue colours.

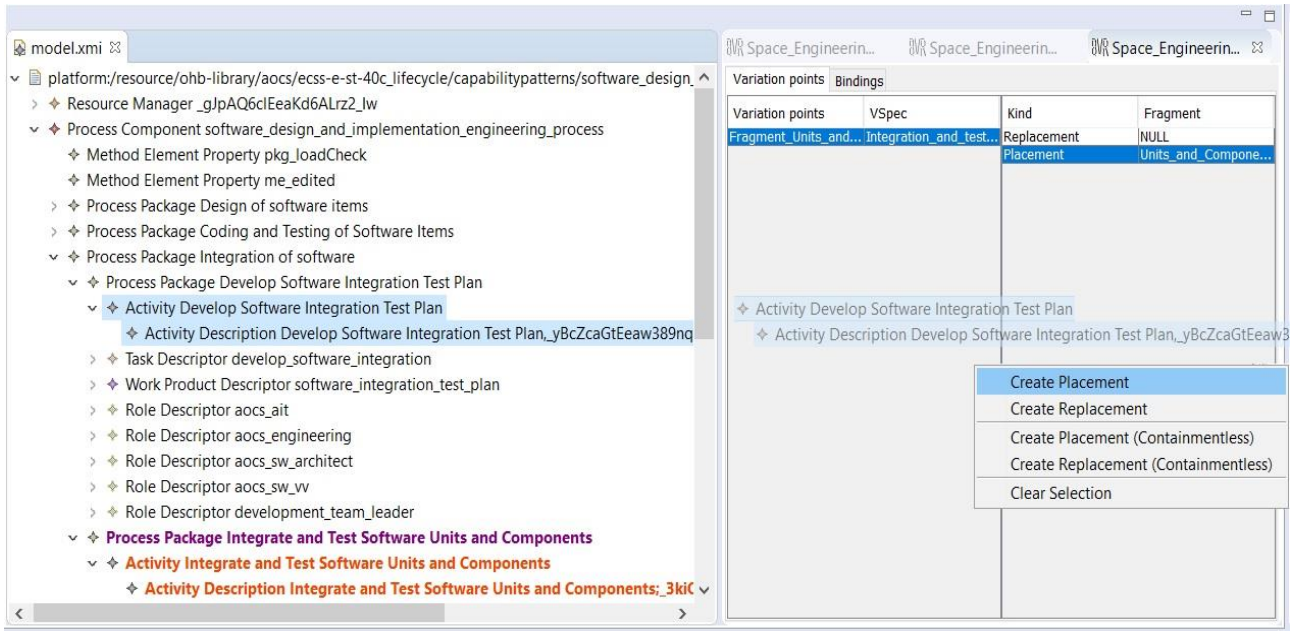


Figure 288: Realization editor

The variation points have the associated bindings, i.e., the mapping of boundary elements between placements and replacements. As the requirements are either included or exuded based on the criticality of the system, the bindings specifies the replacements with NULL elements. Otherwise the specific replacement elements needed to be selected in the bindings tab. The variation points also refer to the VSpecs to define what abstract notion of variability the variation point actually realizes. The inclusion/exclusion of particular choices for the configuration/resolution is therefore considered. In order to derive the configuration, the execute option in particular resolution (editor) is selected. This generates the desired models that are automatically exported back to the EPF-C. Consequently, the pop-up menu is appeared. The yes option loads the changes into EPF-C.

11.2 Product Lines

In the context of AMASS, CHESS-ML has been selected to model the systems. CHESS-ML, however, does not offer any support for managing variability as needed for our purposes. The integration between CHESS and BVR represents a feasible and technically advantageous solution for variability management at product level.

11.2.1VSpec, Resolution, and Realization

The generation of target product is performed with three editors: VSpec, Resolution, and Realization. To create the BVR model file, right click the project/folder in Eclipse IDE, select New / Other and find BVR model under Example EMF Model Creation Wizards (the easiest way is to search for “BVR” once you have selected New/Other); fill in your own preferred name(s); select BVR Model from dropdown menu “Model object”. The instructions for the VSpec, Resolution, and Realization editors might be located in the Deliverable 4.3 of VARIES (VARiability In safety-critical Embedded Systems) project [10].

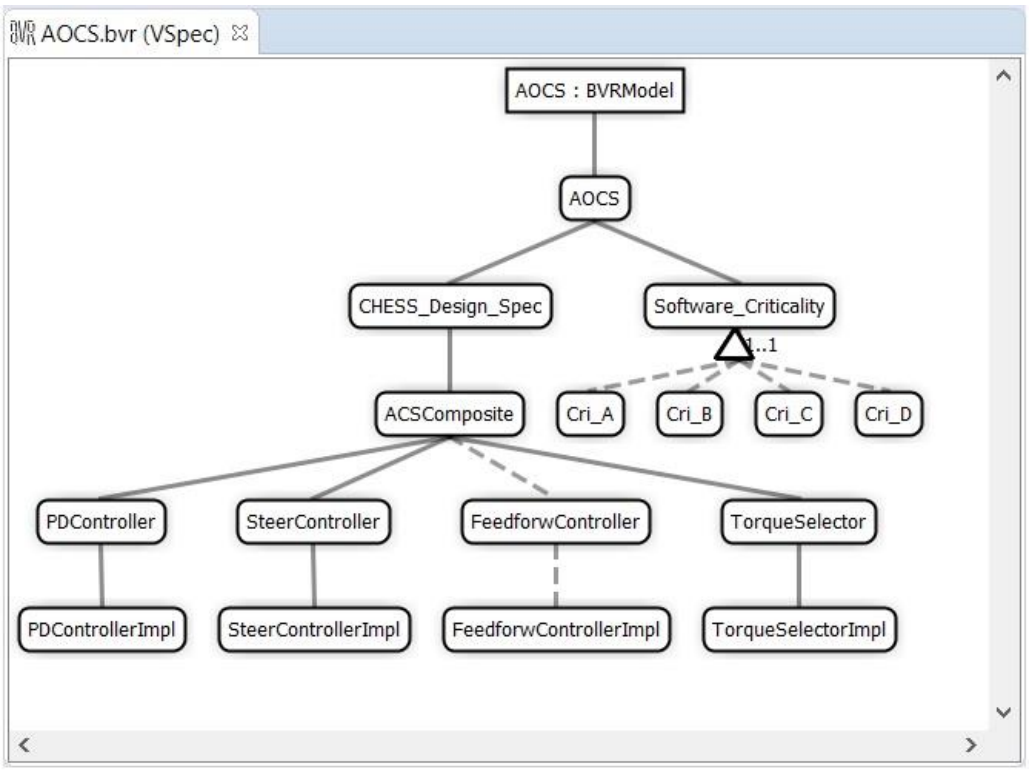


Figure 289: VSpec model regarding Attitude Control System

The resolution models are automatically generated from the VSpec model. To perform the resolution, engineers have to specify the desired inclusion/exclusion choices for the specific configuration/resolution. The validation process is executed to conform whether the resolution corresponds to the VSpec model. It is possible to define multiple resolutions for the processes with variabilities.

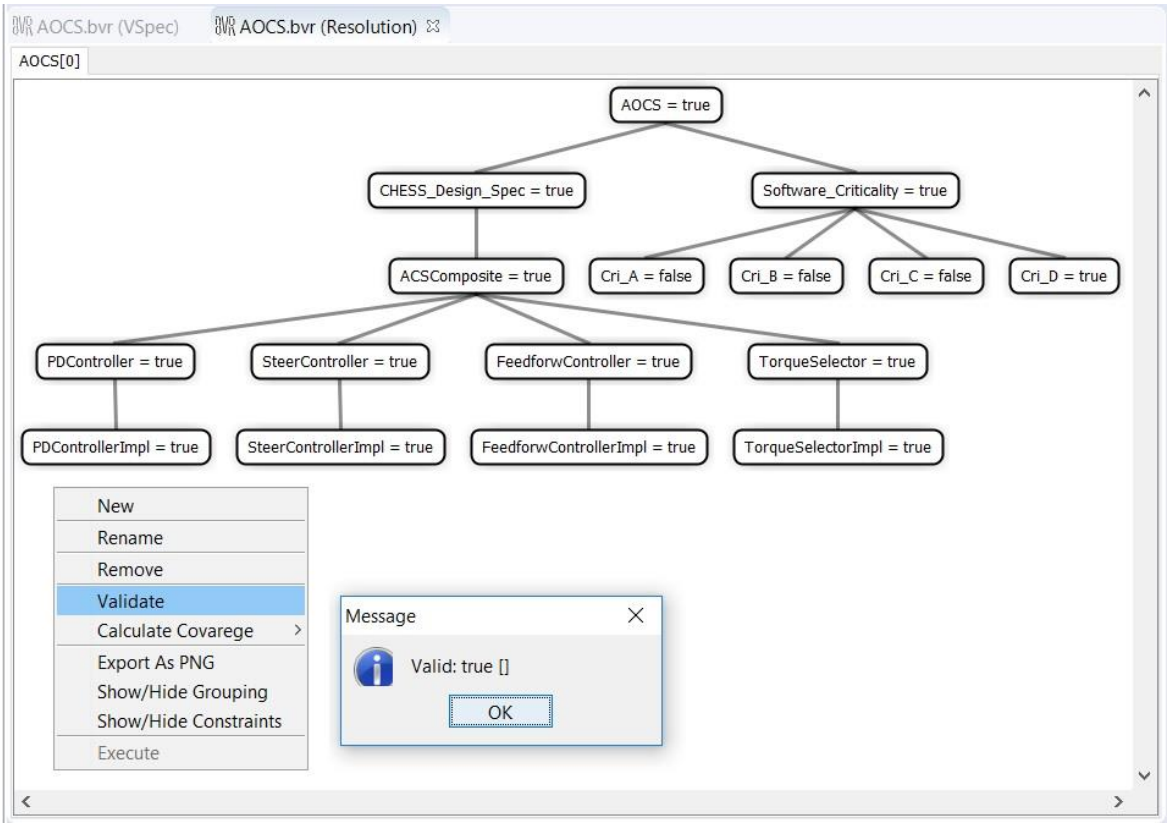


Figure 290: Resolution Model regarding Attitude Control System

Realization is based on the placements and replacements within the fragment substitutions. Fragment substitution removes elements within the placement and substitutes them with the elements in replacement. Therefore, the fragments have been specified for the CHESS model. In order to derive the configuration, the execute option in particular resolution (editor) is selected. This generates the desired product (.uml) model that is exported back to the CHESS project. To see the changes in CHESS editor, the command for removal of orphan entries is implemented.

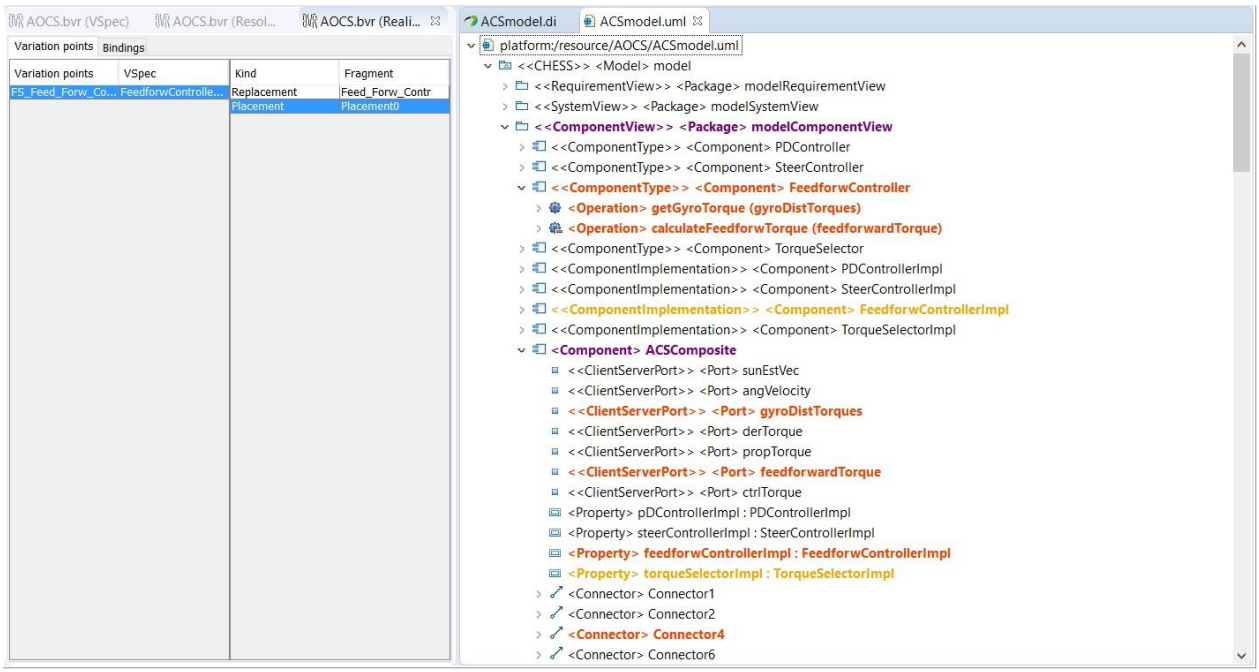


Figure 291: Realization Model regarding Attitude Control System

12 Integration with External Tools

This section explains how a user can connect the AMASSS Tool Platform with third-party tools such as system modelling and V&V tools.

12.1 V&V Tools (FBK)

V&V activities of models can be performed by using some FBK Tools, specifically OCRA for contract based verification and nuXmv for the behaviour verification. The integration with the AMASS platform is done by means of adapters; currently two kind of tool adapters are available: the first one that invokes the FBK tools locally by passing the artefacts and the command via files. The second that does the same functionalities via the OSLC-Automation adapter.

Integration to FBK Tool via files

As mentioned before, the tools have to be installed locally to the AMASS platform; once downloaded the tools from the FBK site (http://es.fbk.eu/technologies), we need to configure the adapter in the way explained in the paragraph 7.11

Integration to FBK Tool via OSLC

FBK exposes the OSLC service provider that allows the tool integration remotely. At the address http://docker-es.fbk.eu:8080 responds both the registry server and the automation server. Even in this case we need to configure the adapter in the way explained in the paragraph 7.11

12.2 Requirements Quality Analyzer (RQA)/OSLC-KM

OSLC-KM standard allows exchanging artefact contents from different and heterogeneous sources. In this way, TRC has integrated some of the activities and functionalities of its RQA application within the AMASS platform. In the scope of this prototype P1, the integration has been done in the following way.

The idea is to retrieve meaningful information from any Papyrus model in the scope of the project to be assured, scan it with TRC technology and populate the evidences for this project with the output information.

Technically, it works in the following way: TRC has exposed a web service (available at http://authoring.reusecompany.com:9999/OslcKmService.svc/GetSrlFromContent) which receives the content of the Papyrus model, scans it and returns an instance of the OSLC-KM standard to the AMASS platform. Then, the platform receives and transforms it to an instance of an evidence. Finally, this evidence is saved in the current project.

From the user point of view inside the AMASS platform application, you can find a new menu option whose name is “OSLC-KM” with several submenus, the relevant one for this integration is “Import evidence model from file”.

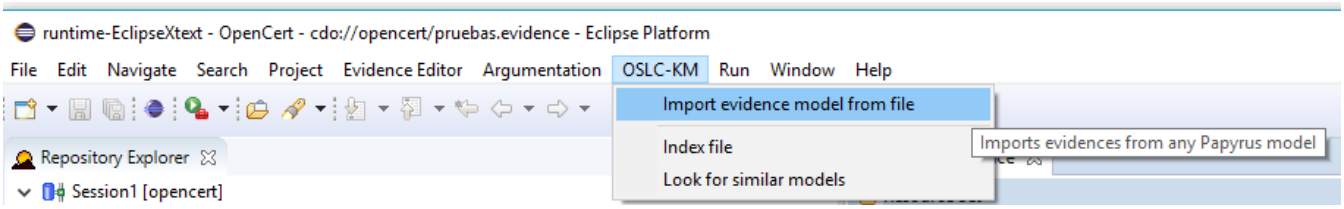


Figure 305 - New OSLC-KM menu option

If it's clicked then a wizard is started, it has two steps:

1) Select the Papyrus file to be added as evidence:

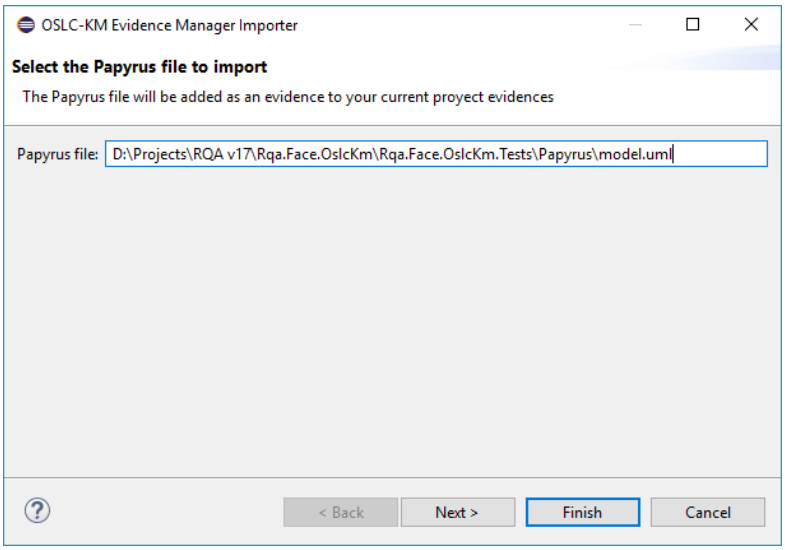


Figure 306 - Import Papyrus file wizard page #1

2) Select the URI of the OSLC-KM Web Service: by default, it's filled up with the public OSLC-KM Web Service, but the user can change it.

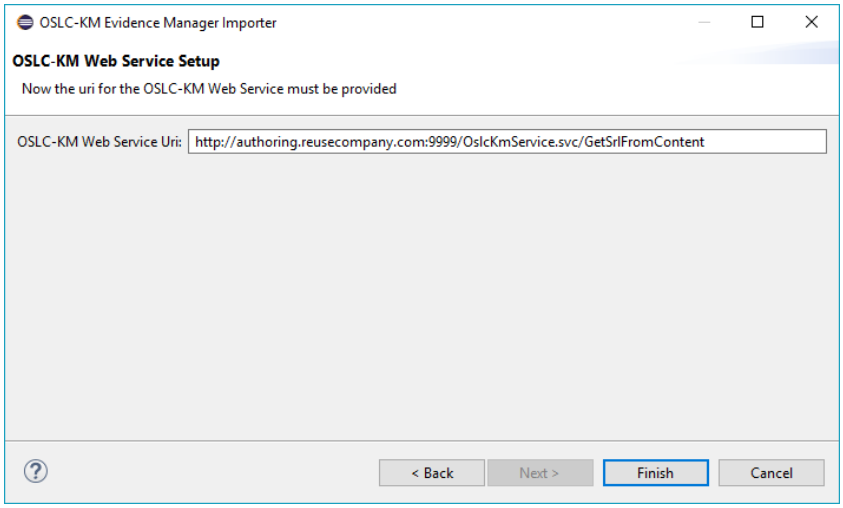


Figure 307 - Import Papyrus file wizard page #2

After finishing the wizard, if the model has been imported successfully as an evidence an information message box is shown as in the following *Figure 308*

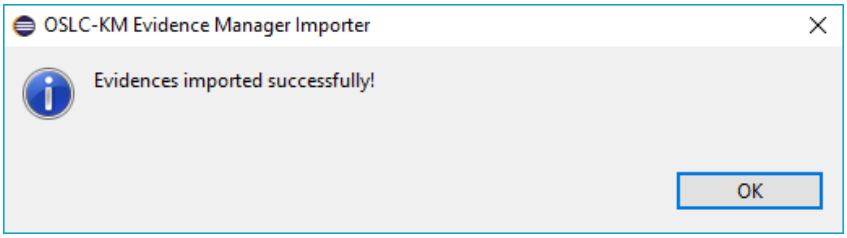


Figure 308 - Import Papyrus file success message

The output is stored in a file whose name is “pruebas.evidence” stored in the current project, as shown in the following *Figure 309*

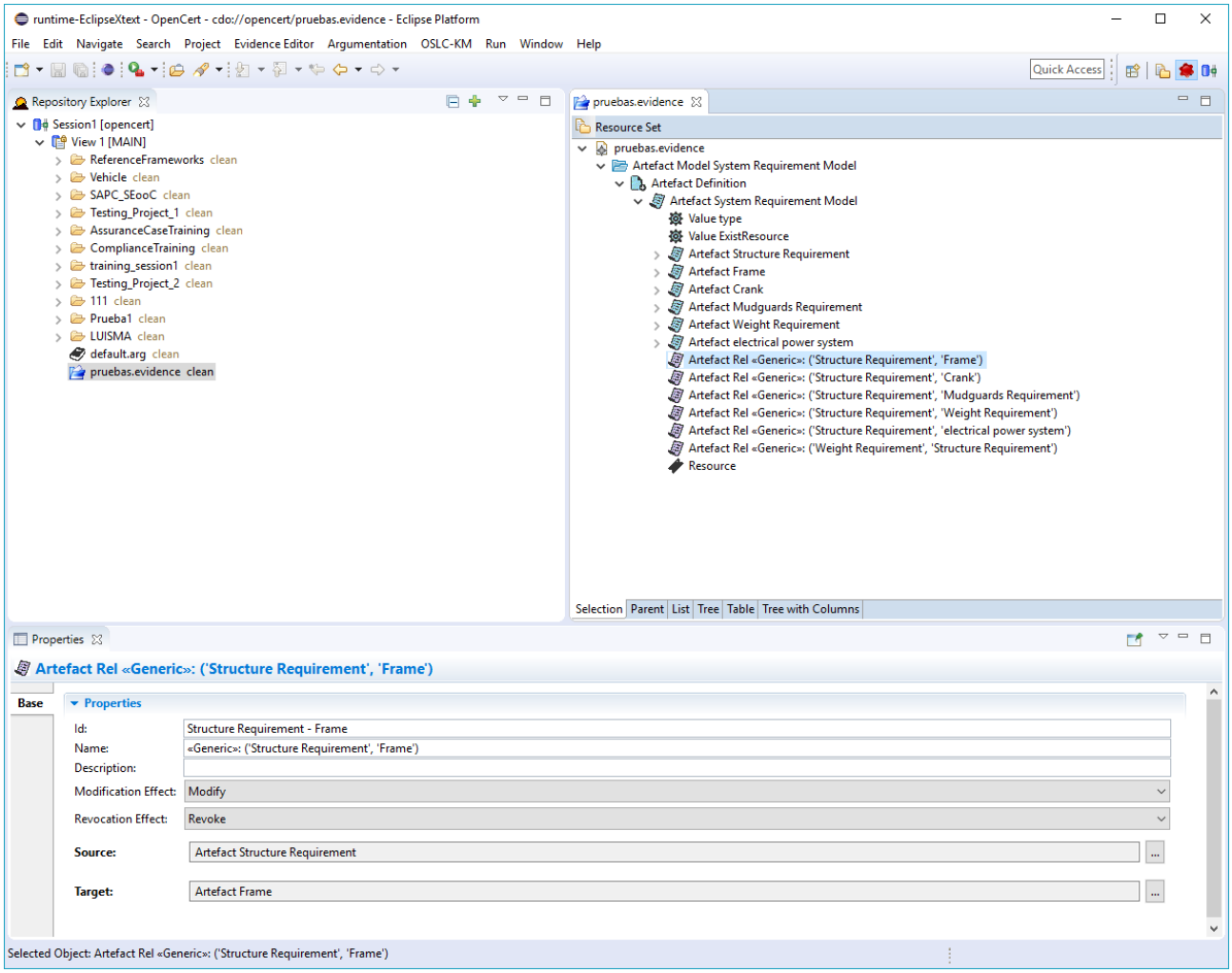


Figure 309 - Evidence imported in AMASS database

This is a basic demonstration of the power of OSLC-KM technology for integration of multiple sources of structured information. In the next prototype (Prototype P2), several improvements will be performed in this integration:

1. Selection of the project to insert the evidence.
2. Selection of the evidence name to be inserted in the project.
3. Moving the configuration of the OSLC-KM web service URI to a configuration plugin.
4. Widening the scope of the models to be scanned, such as Rhapsody or Magic Draw models.
5. Allow selecting those types of models in the wizard selection page.

On the other side, from RQA side, the same integration shall be created. From the quality analysis result window, there will be a possibility to store those results in the AMASS database. It will be performed with the same idea, mapping the results to an instance of the OSLC-KM standard, then loading it in AMASS platform and finally storing them as a new evidence for the selected project.

12.3 Reuse Discovery using OSLC-KM (TRC)

As described in 12.2 (*Requirements Quality Analyzer / OSLC-KM*), OSLC-KM standard allows exchanging artefact contents from different and heterogeneous sources. In this way, both operations (Index and Search for similar models) have been integrated within the AMASS platform. In the scope of this prototype P1, the integration has been done in the following way.

The main idea is to retrieve relevant models for a given Papyrus model, based on the tools provided by TRC.

The implementation for this prototype P1 is only based on local files.

Technically, it works in the following way: a REST web service has been published by TRC with bot operations:

- Indexing a model, available at http://authoring.reusecompany.com:9999/OslcKmService.svc/IndexContent
- Retrieving a model, available at http://authoring.reusecompany.com:9999/OslcKmService.svc/RetrieveSimilarContent

Both functions receive the content of a Papyrus model in order to index the content, and returns result:

- **IndexContent**: returns true/false, depending on whether the indexing process was successful or not.
- **RetrieveSimilarContent**: returns a list of physical URIs as a result of the query to get the relevant models detected for the given model.

From the user point of view inside the AMASS platform application, you can find a new menu option whose name is "OSLC-KM" with several submenus, the relevant ones for this integration are "Index file" and "Look for similar models".

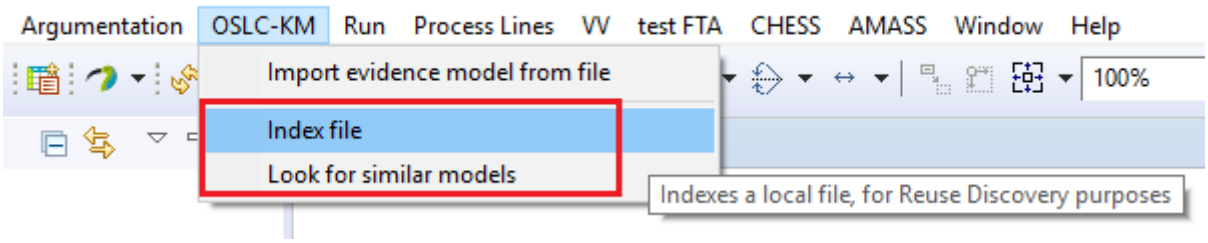


Figure 310 - New OSLC-KM menu option for Reuse Discovery

When any of them is selected, then a wizard is started, it has two steps:

1) Select the Papyrus file to be indexed into the database (if the selected option was "*Index file*"), or Select the Papyrus file to discover similar models (if the selected option was "*Look for similar models*"):

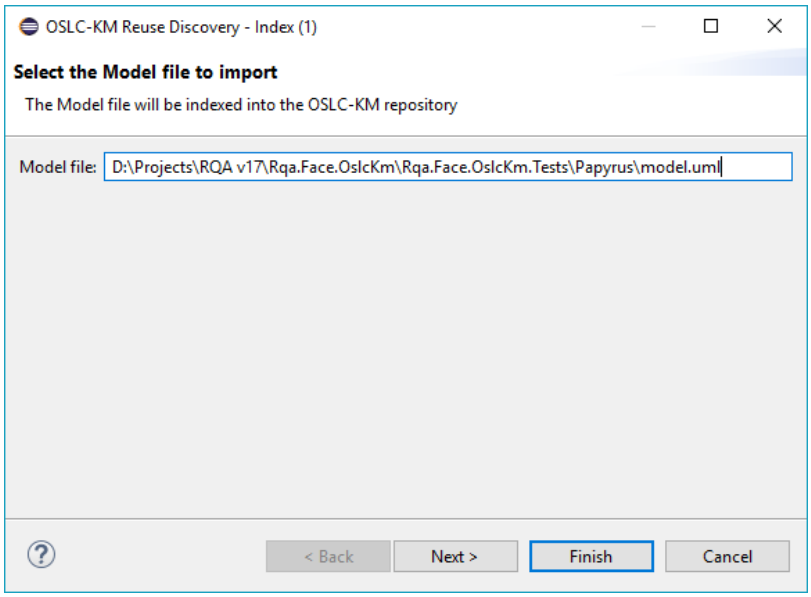


Figure 311 - Wizard page #1 for file selection

2) Select the URI of the OSLC-KM Web Service: by default, it's filled up with the public OSLC-KM Web Service, but the user can change it.

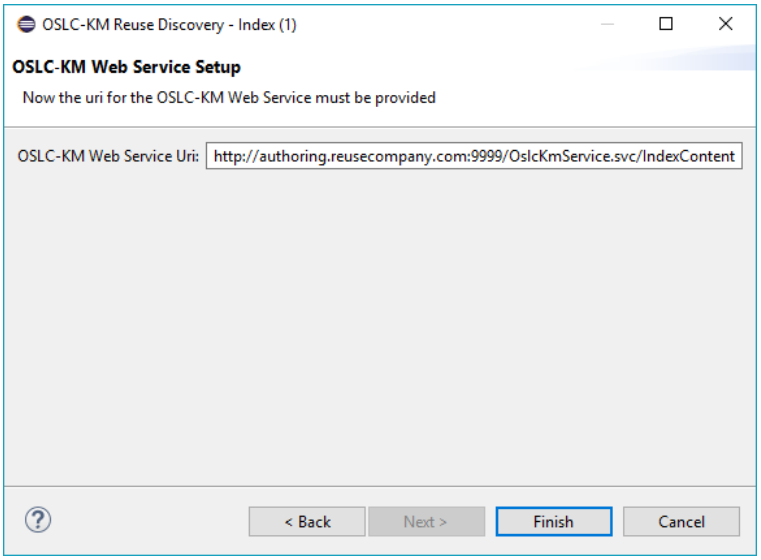


Figure 312 - Wizard page #2 for "Index file" operation

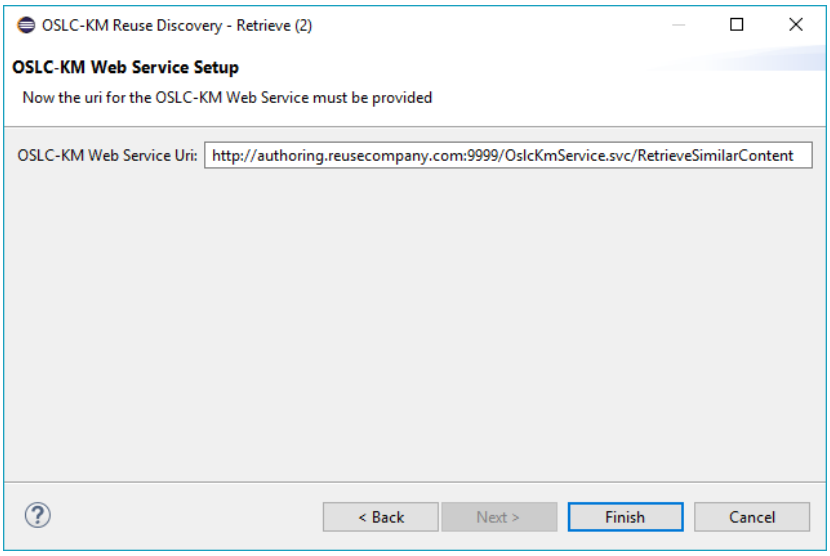


Figure 313 - Wizard page #2 for "Look for similar models" operation

After finishing the wizard, we get either the confirmation or failure of the performed operation:

- If it's an indexing operation, we must get a message box as follows:

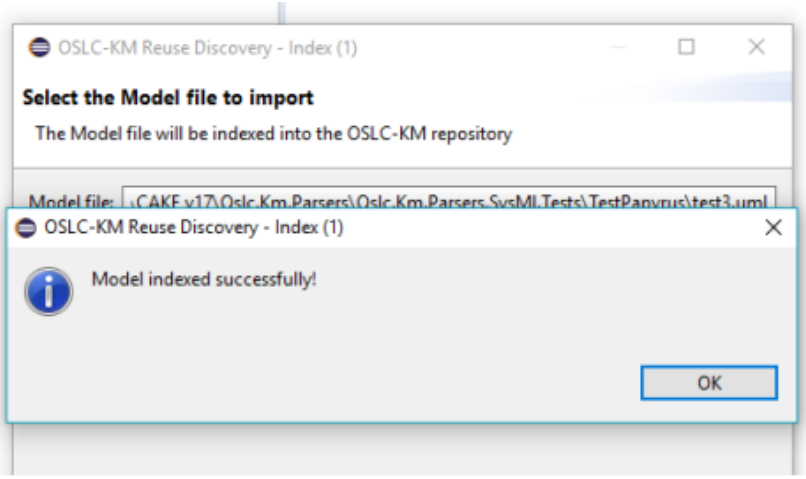


Figure 314 - Confirmation dialog for "Index file" operation

- If it's a retrieval operation, we must get the URIs of the relevant models recovered out of the selected one:

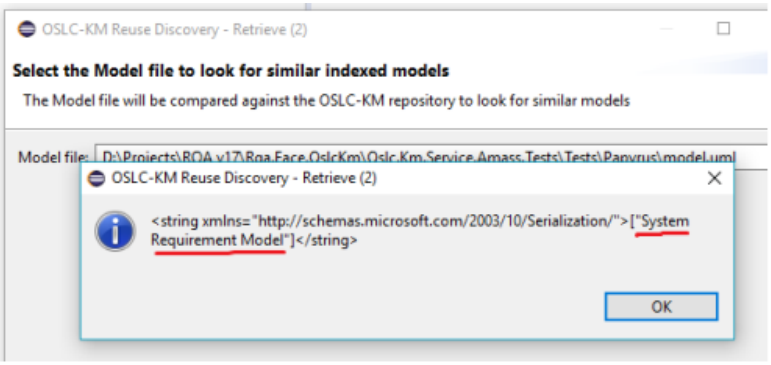


Figure 315 - Result dialog for "Look for similar models" operation

This is a basic demonstration of the technical integration of OSLC-KM within the AMASS platform in the context of Reuse Discovery. In the next prototype (Prototype P2), several improvements will be performed in this integration:

1) Moving the configuration of the OSLC-KM web service URI to a configuration plugin.
2) Widening the scope of the models to be indexed (not only Papyrus models, but also Rhapsody, Magic Draw, etc).
3) Allow browsing models in the wizard selection page (instead of typing the whole file path).
4) Integration with CDO.
5) Improving the way search results are displayed in the user interface

12.4 SVN

The current support for integration with SVN is part of the functionality for Evidence Management. It is presented in Section 9.1.

References

[1] Tuft, B.: Eclipse Process Framework (EPF) Composer Installation, Introduction, Tutorial and Manual (2010), https://eclipse.org/epf/general/EPF_Installation_Tutorial_User_Manual.pdf
[2] Object Management Group: Software & systems process engineering meta-model specification. Tech. rep. (2008), http://www.omg.org/spec/SPEM/2.0/
[3] McIsaac, B.: Ibm rational method composer: Standards mapping. Tech. rep., IBM Developer Works (2015)
[4] Origin Consulting, GSN Community Standard Version 1. 2011.
[5] Richard Hawkins, Software Contribution Safety Argument Pattern (2009) http://www.goalstructuringnotation.info/archives/234
[6] OpenUP: Key capabilities of the unified method architecture (uma). http://epf.eclipse.org/wikis/openupsp/base_concepts/guidances/concepts/introduction_to_uma,_94_eoO8LEdmKSqa_gSYthg.html, accessed: 2017-01-20
[7] FBK. OCRA: A tool for Contract-Based Analysis. Available at https://es.fbk.eu/tools/ocra/.
[8] A. Cimatti, M. Roveri, A. Susi, and S. Tonetta. Validation of Requirements for Hybrid Systems: a Formal Approach. ACM Trans. Softw. Eng. Methodol., 21(4):22, 2012.
[9] FBK. nuXmv: a new eXtended model verifier. Available at http://nuxmv.fbk.eu.
[10] Deliverable 4.3 VARIES VARiability In safety-critical Embedded Systems, https://github.com/SINTEF-9012/bvr/raw/master/docs/VARIES_D4.3_v01_PU_FINAL.pdf
[11] https://www.youtube.com/watch?v=edHAxb8-1Io, https://www.eclipsecon.org/france2016/sites/default/files/slides/EclipseConf2016%20sysml%20and%20requirements.pdf
[12] Stefano Tonetta: Linear-time Temporal Logic with Event Freezing Functions. GandALF 2017: 195-209

Appendix A. Standard Modeling and compliance in EPF

Standard modeling and compliance has been approached in the commercial version of the EPF composer, the IBM Rational Method Composer (RMC) [3]. This feature of the tool is based on the capacity of RMC to define new process elements. Specifically, in this solution, a new type of process element named Requirement is modeled and instantiated to define standard requirements. However, this functionality is not at disposal in the EPF composer. In the following subsections, we describe how this solution can be adopted in EPF.

A.1 Standard modeling

So, in order to define requirements in EPF, we exploit the Variability Mechanisms of SPEM 2.0 that are at disposal in this tool. Our idea is to define a customized Practice to represent requirements that will be extended and customized to define requirements contained in standards.

The definition of the customized practice to represent the base requirement will be as follows. We define a new Practice with name “requirement” in our plug-in (Section 7.2.2 of the EPF manual [3]). Then, in the Icon section of the Description tab, we include the icon of a target by clicking “Select…” in sections “Shape Icon Preview:” and “Node Icon Preview”. This step is optional but it helps to distinguish better Requirements from regular Practices. In order to facilitate this initial step, in the context of the AMASS project we have defined a plug-in named compliance_modeling with the Requirement practice defined and customized, which can be imported (see Section 10.8 of EPF manual) in the method library that requires Standard Modeling.

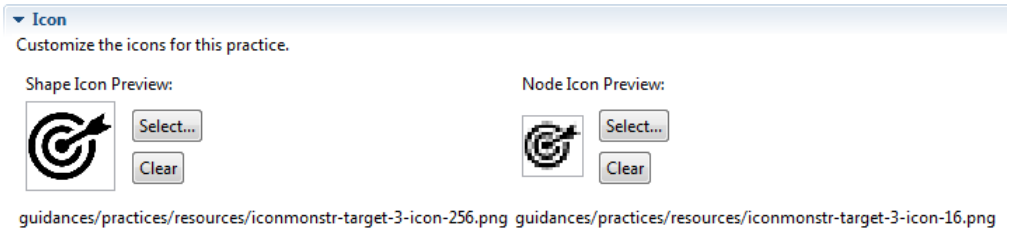


Figure 316 - Icon customization of Requirement practice.

In order to define requirements of a specific standard, we create a new plug-in and a content package. In the Guidance folder of this plug-in, we create a regular Practice but in the “Content Variability” (see Figure 317) section of the Description tab, we select Extends in the Variability Type field and click “Select…”. In the new dialog that appears, we select the requirement practice that we have defined (or it is in the compliance_modeling plug-in). Then, we give a name an appropriate description for the requirement. Optionally, we can customize this new requirement using an Icon. We repeat the same method, to define new requirements.

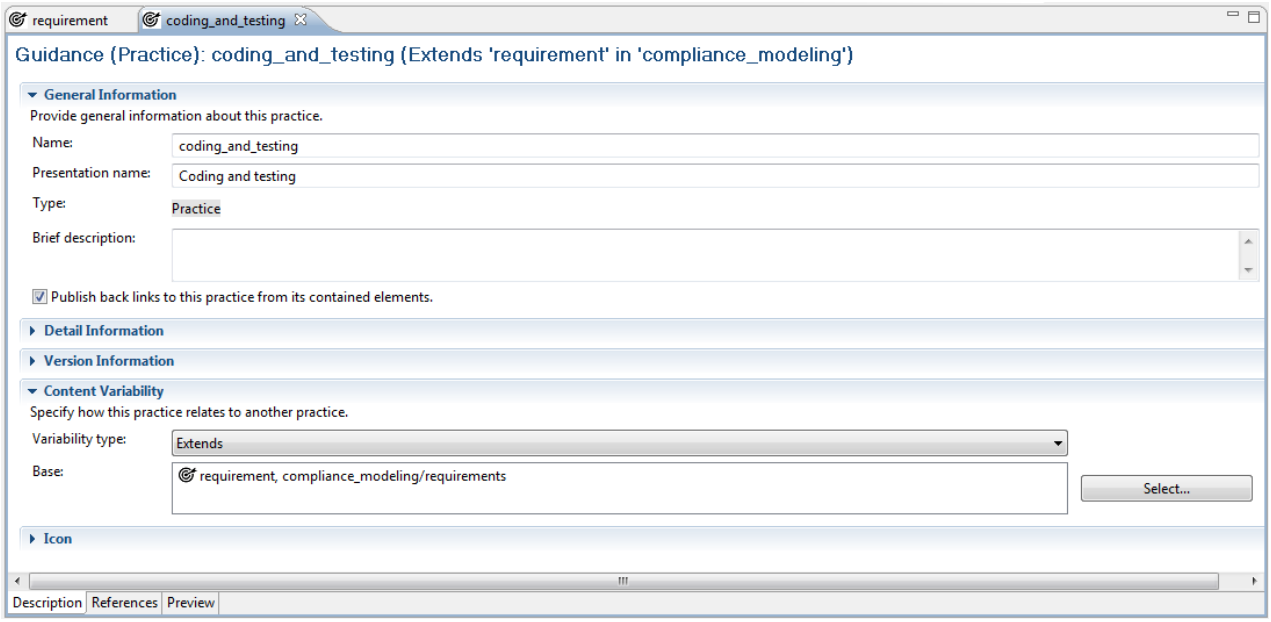


Figure 317 - Definition of a new standard requirement.

Requirements can be nested and composed to emulate the structure of the standard or define multi-part requirements. To include a requirement inside another requirement, we right click and select "New -> Practice" and follow the method described behind to define a new standard requirement. Using this procedure, we can obtain requirements structure like the one depicted in Figure 318.

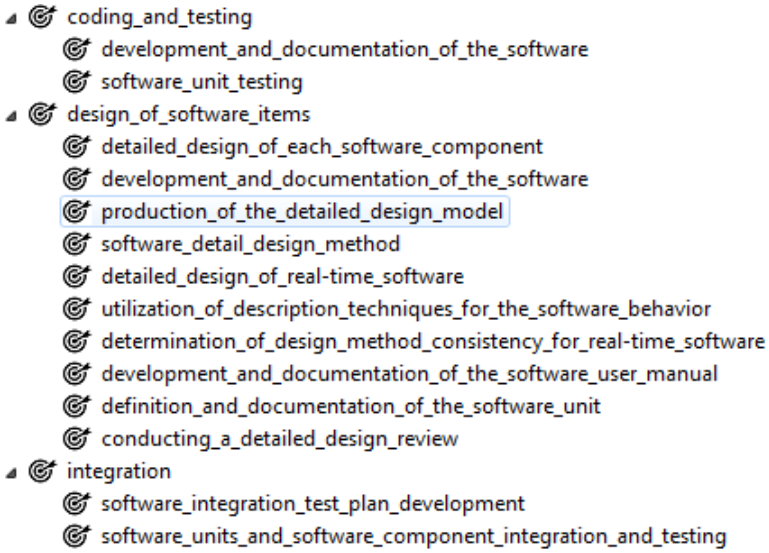


Figure 318 - Standard requirements modeled in the EPF composer.

A.2 Process compliance

Process compliance can be modeled in EPF using a similar procedure to those presented in [3]. In order to apply this step, we must model first the lifecycle contained in the standard as a process in the EPF composer (see Section 4 of this manual) in a separate plug-in. Then, we define a new plug-in that will be used just to map the compliance of the process with the standard. So, in this point, we have three plug-ins (at least) in our Method Library, one to model the standard, another to model the lifecycle and a third one, for the mapping (see Figure 319). This procedure makes possible to re-use processes and standards in different mappings for compliance.

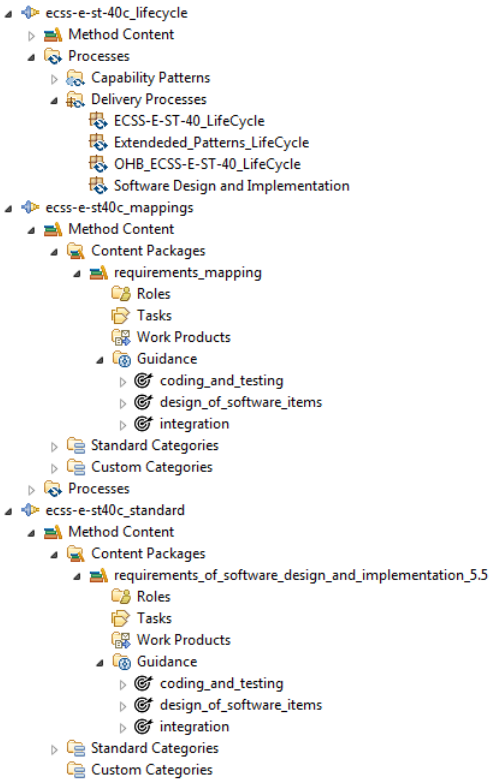


Figure 319 - Method library organization for standard mapping.

In the plug-in for the mapping, we make a copy of all the requirements that we have defined in the plug-in for the standard. To do so, we select all the requirements, right click and select Copy. Then, in the Guidance folder of the plug-in for mapping, we right click and select Paste. Then, we modified each copied requirement in the plug-in for compliance in the following way. In the Content Variabilty section of the Description tab, we replace “Extends” by “Contributes” and click “Select…” to select the original requirement which is in the plug-in for the standard. We can see an example of the modification of the copied requirement in Figure 320.

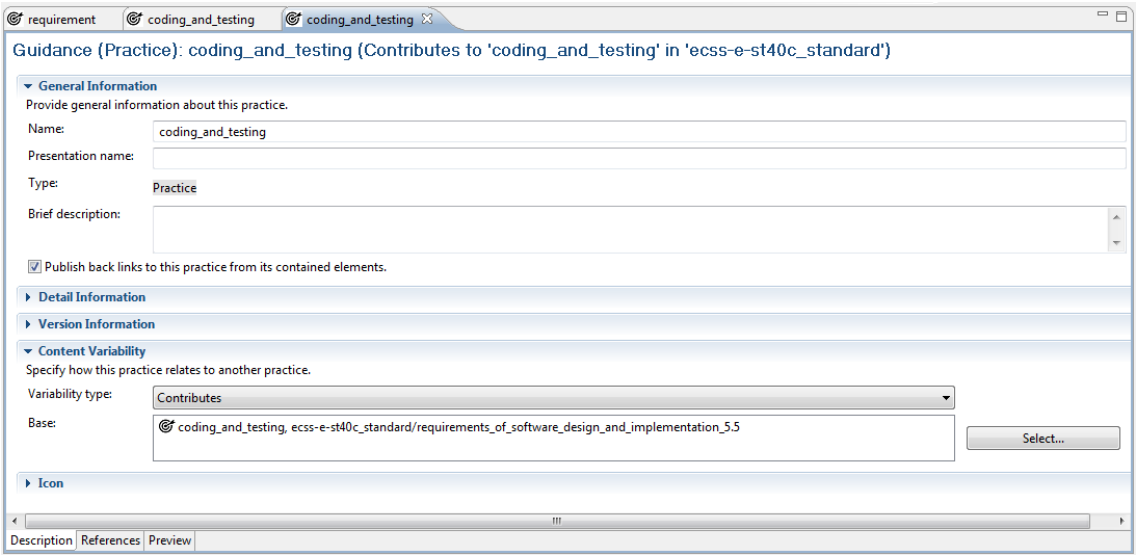


Figure 320 - Mapped Requirements in the EPF composer.

The next step to model the compliance is to link this requirement with the evidence of the accomplishment of this requirement. To do so, we select the References tab in the copied requirement and click “Add…”. In the dialog that appears, we select the process element that demonstrates the satisfaction of the

requirement. In our solution, we can provide the following types of process elements: Activity, Capability Pattern, Delivery Process and Guidance. The EPF composer allows to add other kind of process elements like work products, roles and tasks but the mentioned elements are the only ones that can be taken directly from the lifecycle of the standard to demonstrate its compliance. By instance, for the requirement "Development of the Software" we provide as evidence the activity "Develop software" (see Figure 321).

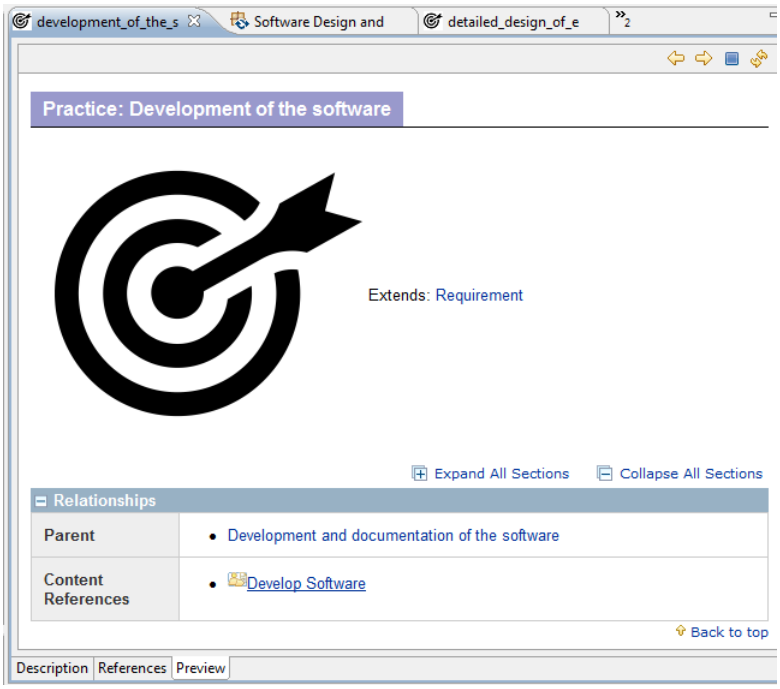


Figure 321 - Preview tab of the mapped requirement "Development of the Software".

We can model three different situations of compliance using this solution, full, no-compliance or partial compliance. The full compliance of a requirement is modeled providing evidence for the requirement or each sub-requirement in case of multi-part requirements as "coding_and_testing" in Figure 318. The no compliance is modeled by not providing any kind of evidence. Finally, the partial compliance is modeled by decomposing a requirement in sub-requirements. Then, for each part of the requirement which is accomplished, we provide evidence and for those sub-requirements not satisfied we provide nothing. This is the situation of the requirement Document sys requirements in Figure 322. The sub-requirement "Detail a use case" is accomplished, while the sub-requirement "Identify and outline requirements" is not accomplished.

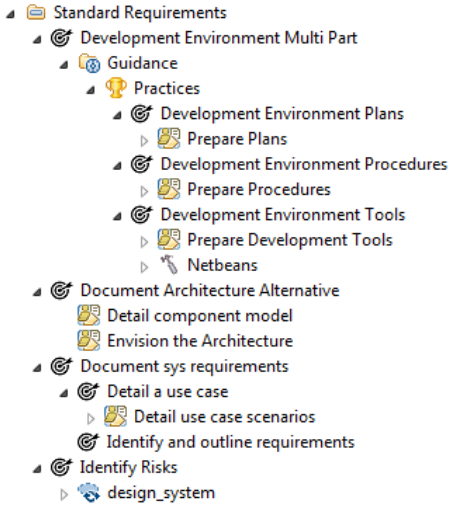


Figure 322 - Compliance situations in the EPF composer.

We can group mapped requirements in Custom Categories to facilitate their visualization in the Browsing perspective. To do so, we create a new Custom Category (Section 4.5.3 of EPF manual) in the plug-in for the mapped requirements named Mapped Requirements. In the Assign tab of this Custom Category, we assign all requirements of this plug-in.

A.3 Recommendation Tables Modeling

Using the same principles of the customization and extension of practices used for requirements, we have developed a method to model recommendation tables in the EPF composer. These new concepts can be found in the compliance_modeling plugin in the content package named recommendation_tables (see Figure 323). In order to illustrate the modeling of the recommendation tables, we use portions of ISO 26262.

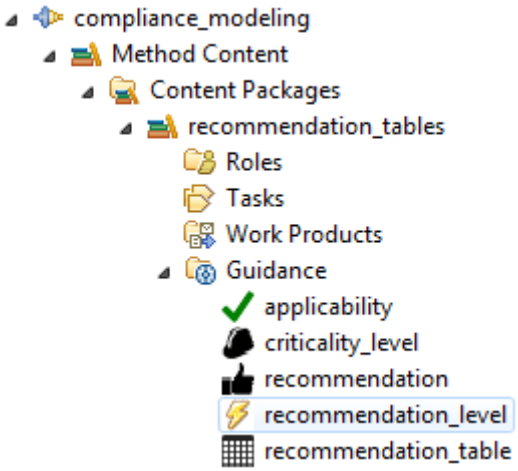


Figure 323 - Customized practices to model recommendation tables

The first step to model the recommendation tables is to extend practices of the compliance_modeling plug-in to use the same notation for criticality and recommendation levels used in the standard. For instance, in the case of ISO 26262, we need to create five specializations of criticality_level, ASIL A, ASIL B, ASIL C, ASIL D and ASIL QM, and three specializations of recommendation_level, highly recommended, not recommended and recommended. In order to accomplish this task, we create a new practice (see Section 7.2.2 of EPF Manual) for each specialization that we want to model in the Guidance folder of our plug-in. We give to these practices names that correspond to the criticality and recommendation levels considered in the standard. Additionally, these practices will be linked to the original criticality_level defined in compliance_modeling plug-in using variability relationships. So, in the Content Variability section of the Description tab of the Practice, we select Variability type Extends. We push the **Select..** button and in the dialog that appears, we select criticality_level. The final result is depicted in Figure 324. Optionally, we can customize this practice with an icon, in this case we have used a green helmet. We repeat the same operation for the other criticality and recommendation levels. The case of the recommendation levels is the same but the base element in the Content Variability section of the practice is recommendation_level.

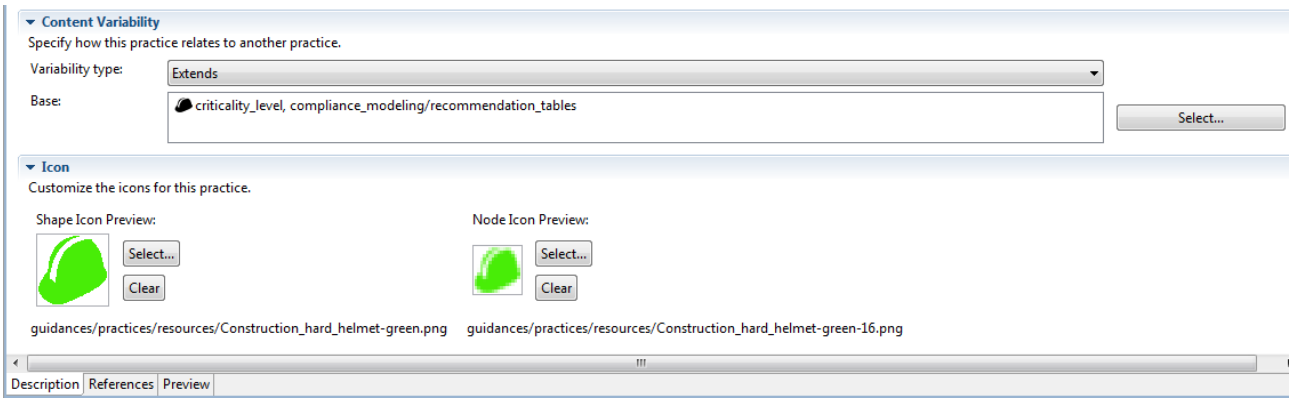


Figure 324 - Modeling of criticality level for ISO26262

Once, we have the specializations of the criticality and recommendation levels, we can start the modeling of the recommendation tables. We recommend performing this task in a separate Content Package, to create a new Content Package follow the instructions depicted in Section 4.2.3 of the EPF Manual. The plug-in will have a structure similar to the plug-in depicted in Figure 325. In this case, we have a separate content package for criticality and recommendation levels too.

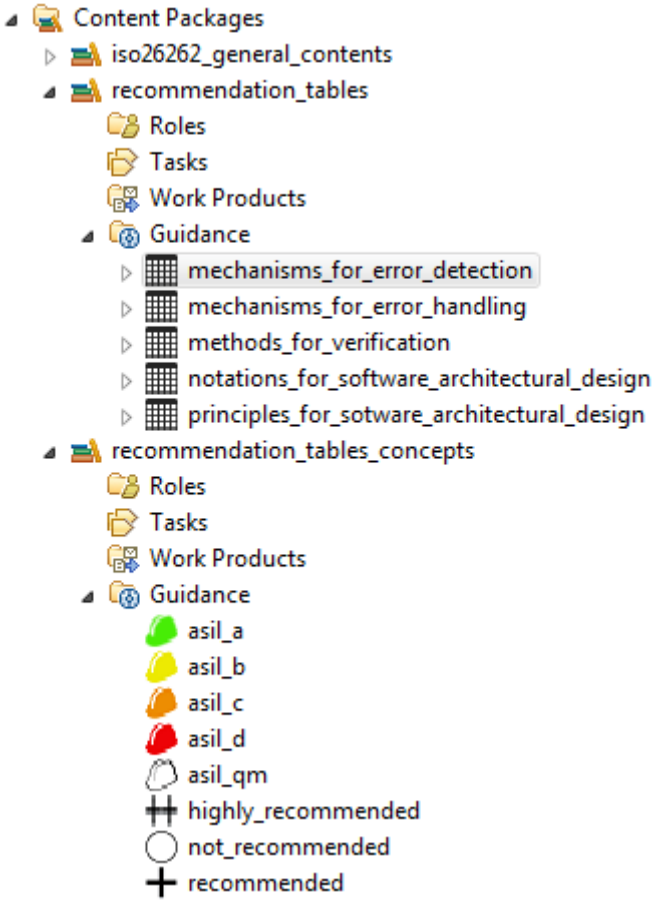


Figure 325 - Recommended plug-in structure for modeling recommendation tables

We illustrate the modeling of the recommendation table using Figure 326 - Mechanisms for error detection at the software architectural level of ISO 26262., which is part of ISO 26262. The starting point for the modeling of a recommendation table is the specialization of the practice recommendation_table (see Figure 323). To do so, we create a new practice with the name of the table that we want to model. In this case, we create the practice with the name mechanisms_for_error_detection. Then, in the Content

Variability section, we select Extends and recommendation_table as base varability element. Optionally, we can use an icon to make easier to distinguish between recommendation tables and standard practices.

	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Range checks of input and output data	++	++	++	++
1b	Plausibility check[a]	+	+	+	++
1c	Detection of data errors[b]	+	+	+	+
1d	External monitoring facility[c]	o	+	+	++
1e	Control flow monitoring	o	+	++	++
1f	Diverse software design	o	o	+	++

[a] Plausibility checks can include using a reference model of the desired behaviour, assertion checks, or comparing signals from different sources.

[b] Types of methods that may be used to detect data errors include error detecting codes and multiple data storage.

[c] An external monitoring facility can be, for example, an ASIC or another software element performing a watchdog function.

Figure 326 - Mechanisms for error detection at the software architectural level of ISO 26262.

Each row of the table is modeled as a recommendation practice (see Figure 323). In order to attach recommendations, we right click the table and select **New -> Practice**. In this way, recommendations will be nested inside the recommendation table. Then we customize this practice by linking it with recommendation in the Content Variability section of the practice, selecting Extends in the variability type and optionally, adding a new icon.

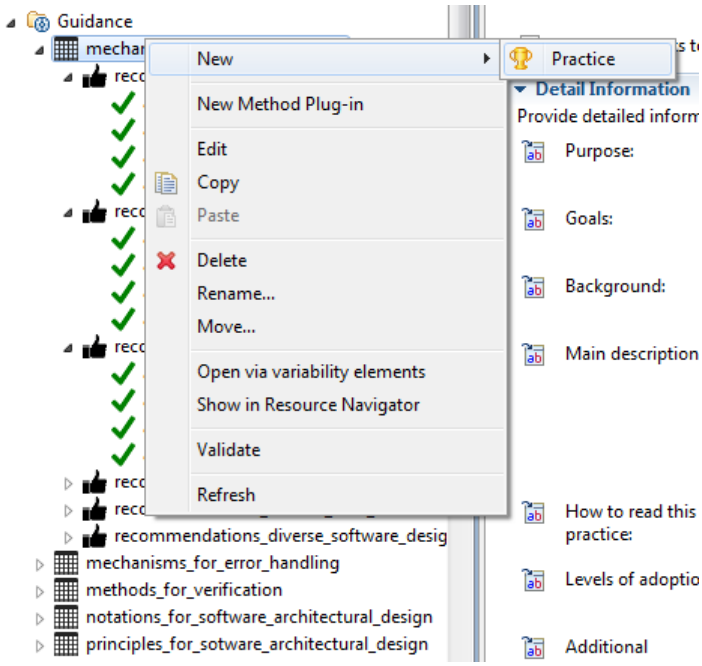


Figure 327 - Adding recommendations to the table

Recommendations must be linked with those process elements for which the recommendation is produced. For example, the first row of is recommendations on the use of the practice Range checks of input and output data. In order to link both elements, the recommendation and the recommended process element, we go to the references tab of the recommendation and select **Add...** In the dialog box that appears, we select the practice range_checks_of_input_and_output_data. The final result is depicted in Figure 328.

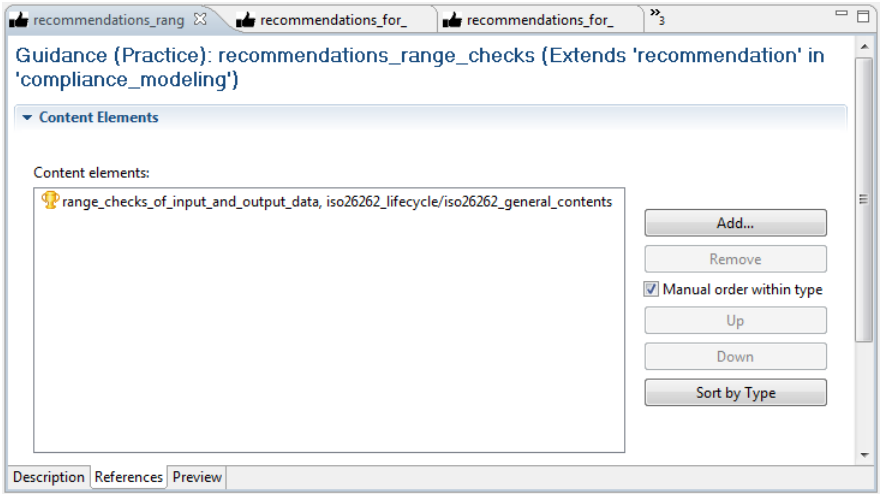


Figure 328 - Link between recommendation and recommended element.

In order to model the level of recommendation for each criticality level, we use applicability practice (see Figure 323). We right click the recommendation and select **New -> Practice**. We give an appropriate name to this practice, by instance for the first row of and ASIL A, we can use the name applicability_range_checks_asil_a. The content variability section of this practice is modified as follows, Variability Type is set to Extends and the Base (push **Select...** button) is applicability from the plug-in compliance_modeling. Optionally, we can customize using an icon such as a green tick. The last step is to link the applicability to the corresponding criticality and recommendation levels. To do so, we use the references tab of the applicability practice. In this tab, we push **Add..** and in the dialog that appears we select the corresponding criticality and recommendation levels. The final result should be similar to Figure 329, which models the recommendation for ASIL A of the first row of . We repeat the same operation for each level of ASIL.

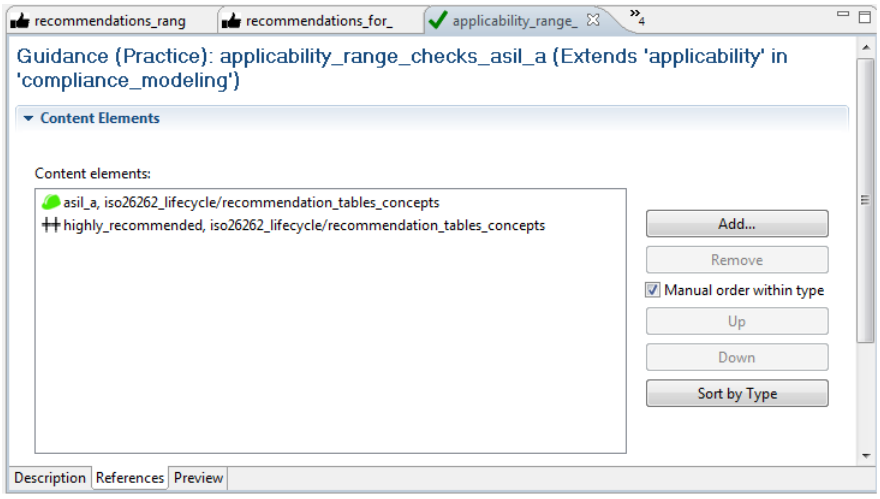


Figure 329 - Reference tab for applicability practice.

The best way to visualize the final result is to create a new custom category and to include practice corresponding to recommendation tables (Section 4.5.3 of EPF manual). The final result in the browsing perspective can be seen in Figure 330.

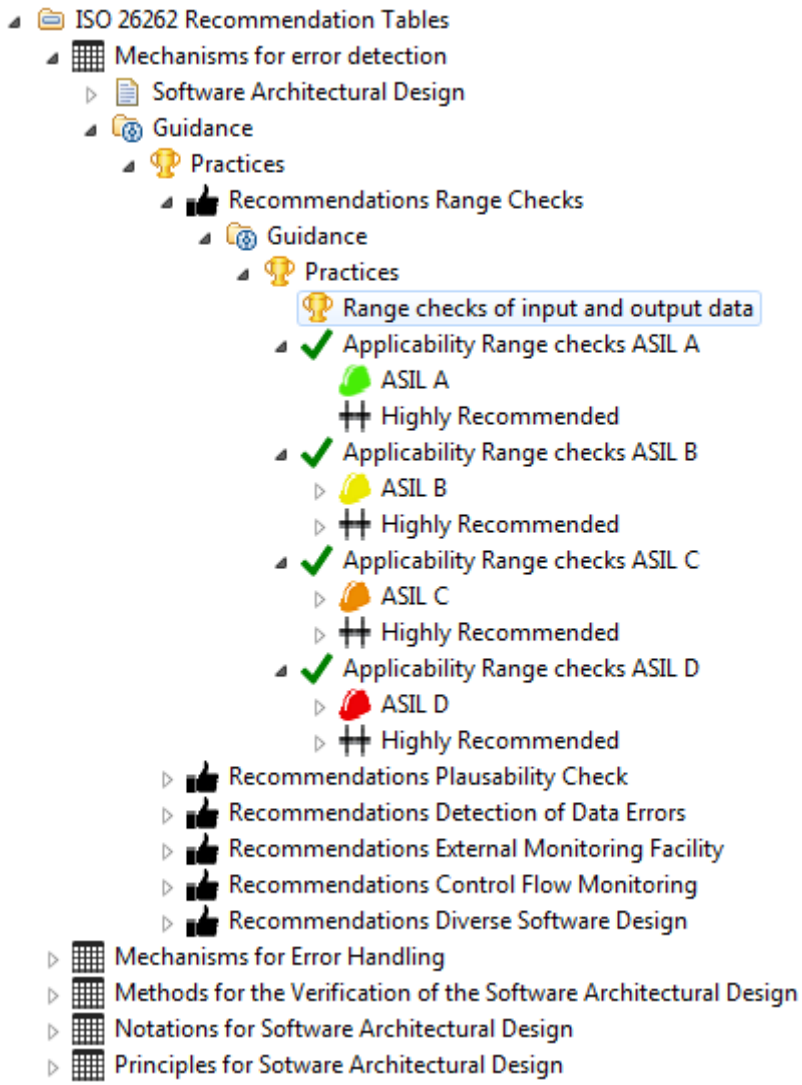


Figure 330 - Recommendation tables for ISO 26262 in the Browsing perspective

A.4 Web-based monitoring of Compliance status

EPF supports the web-based monitoring of the compliance status by means of its generated website. In order to generate a website that contains our mapped requirements, we define a new Method Configuration (Section 4.5.4 of EPF manual) with a name of our election. In the Views of the Method Configuration form, we click “Add view…” and in the dialog that appears, we select Mapped Requirements that will be in the folder “Custom Categories” (see Figure 331).

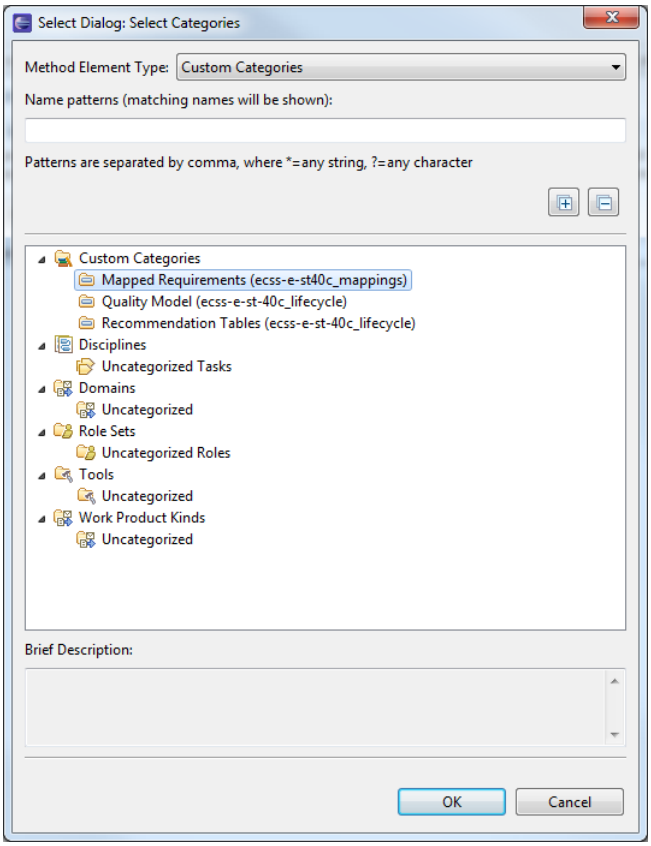


Figure 331 - Dialog to select Mapped Requirements.

In order to generate website (see Section 10.1 of the EPF manual), we click in the menu toolbar Configuration and select Publish. In the wizard that appears, we select the Method Configuration that we have defined to include the Mapped Requirements and click next. In the next screen, we select “Publish the entire configuration”. In the next screen (see Figure 332), we select different publishing options of the website like to give a name or to add a banner. In the last screen, we select the folder in which the generated website will be stored and “Static website”. Finally, we click finish.

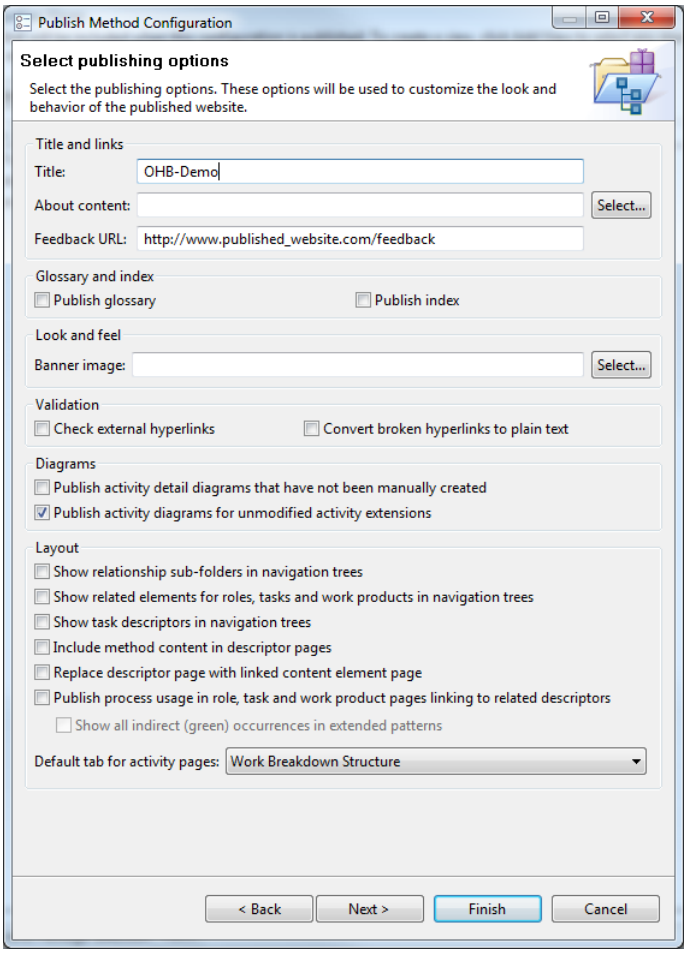


Figure 332 - Dialog for publishing options of the generated website.

The generated website will open automatically in our browser, an example is depicted in Figure 333. In order to check which requirements have been accomplished, we click “Mapped requirements” and expand the tree. Additionally, if we click one of the requirements, we get additional information about the requirement in the right side of the generated website.

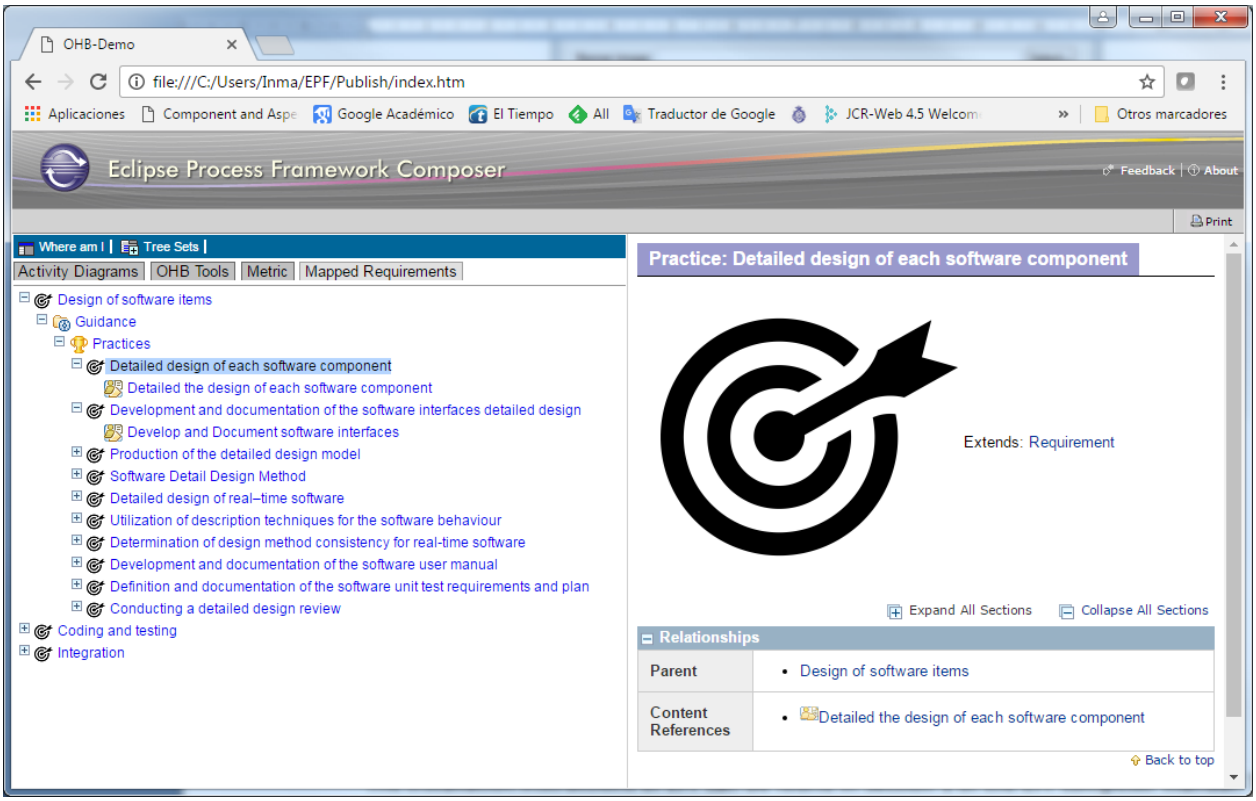


Figure 333 - EPF composer generate website.

Appendix B. OCRA Language to define Formal Properties and Contracts

OCRA constraint language can be interpreted over discrete traces (in this case, it coincides with LTL) or over hybrid traces as described [8]. The relevant syntax of the language has been summarized in Figure 334 - The OCRA language grammar. together with the corresponding mathematical formulation in HRELTL.

constraint	:=	*atom* |	ϕ	:=	a |
		not *constraint* |			$\neg\phi$ |
		constraint **and** *constraint* |			$\phi \wedge \phi$ |
		constraint **or** *constraint* |			$\phi \vee \phi$ |
		constraint **implies** *constraint* |			$\phi \rightarrow \phi$ |
		always *constraint* |			$G\phi$ |
		never *constraint* |			$G\neg\phi$ |
		in the future *constraint* |			$F\phi$ |
		constraint **until** *constraint*;			$\phi\, U\phi$;
		then *constraint* |			$X\phi$ |
		historically *constraint* |			$H\phi$ |
		in the past *constraint* |			$P\phi$ |
		constraint **since** *constraint*;			$\phi\, S\phi$;
		term **at next** *constraint*;			$t\ @F\phi$;
		term **at last** *constraint*;			$t\ @P\phi$;
atom	:=	**true** |	a	:=	$\top$ |
		false |			$\bot$ |
		term relation term |			$t \bowtie t$ |
		time_until(*term* **)** *relation term* |			$\rhd_{\bowtie t} t$ |
		time_since(*term* **)** *relation term* |			$\lhd_{\bowtie t} t$ |
		change(*term* **)** |			$v' \neq v$ |
		fall(*boolean_term***)** |			$\phi \wedge \neg X\phi$ |
		rise(*boolean_term* **)** |			$\neg\phi \wedge X\phi$ |
		boolean_term ;			t;
term	:=	*port* |	t	:=	v |
		constant |			c |
		term function term |			$t \star t$ |
		der(*port* **)** |			$\dot{v}$ |
		next(*port* **)** ;			v';

Figure 334 - The OCRA language grammar.

In case of discrete time, *der*, *time_until*, and *time_since* are not allowed.

Basic formulas are defined with linear arithmetic predicates over the variables or their derivatives. For examples, *x-e<limit* and *der(x)<0* are well-defined formulas. Predicates can be combined with Boolean and temporal operators. For example, *x-e<limit* and *der(x)<0* and *always x-e<limit* are well-defined formulas.

In temporal logic, a formula without temporal operators is interpreted in the initial state. Thus, *x=0* characterizes all traces that start with a state evaluating x to 0, and then x can evolve arbitrarily. Instead, to express that a predicate holds along the whole evolution, one may use the *always* operator as in *always x=0*.

Another classical example of properties is the response to a certain event. The formula *always (p implies in the future q)* defines the set of traces where every occurrence of *p* is followed by an occurrence of *q*. Note that *q* may happen with a certain delay (although there is no bound on such delay). The formula always (p implies *q*) instead forces *q* to happen at the instant of *p*.

The above formulas do not constrain the time model of the traces. Therefore, they can be interpreted either as discrete traces or as hybrid traces. However, the logic is suitable to characterize specific sets of

hybrid traces, constraining when there should be discrete events and how the continuous variables should evolve along continuous evolutions.
The *der(.)* operator is used to specify constraints on the derivative of the continuous evolution of continuous variables. For example, the following OCRA constraint:
always (train.location<=target implies der(train.location)>=0)
characterizes the set of hybrid traces where in all states, if the train has not yet reached the target location, its speed (expressed as the derivative of the location) is greater than or equal to zero.
The *next(.)* operator is used to specify functional properties requiring discrete changes to variables. For example, we can express the property that the warning variable will change value after the train's speed passes the limit with the following constraint:
10 always (speed>limit implies in the future next(warning)!=warning)
The expression *change(x)* can be used instead of *next(x)!=x*.
The expression *fall(x)* is an abbreviation for *x and then(not x)*, i.e. a Boolean term *x* becomes false.
The expression *rise(x)* is an abbreviation for *(not x) and then x*, i.e. a Boolean term *x* becomes true.
In order to constrain the delay between two events, we use the *time_until(.)* and *time_since(.)* operators, which denote respectively the time that will elapse until the next occurrence of an event and the time that elapsed since the last occurrence of an event. For example, the formula *always (p implies time_until(q)<max_delay)* defines the set of hybrid traces where *p* is always followed by *q in less than max_delay time units*.
The operator *. at next .* has a similar but more general purpose. It denotes the value of the left expression, known as the sample, at the next step in which the right expression, known as the trigger, will be true. For example, the formula *always (p implies (((time at next q)- time)< max_delay))* is the discrete time equivalent of the previous example, using an explicit user defined time variable. There is also an *. at last .* operator, which denotes the value of the sample at the last step in which the trigger was true.
For further information see [7] and [12].