## CONTENTS

1 INTRODUCTION 9

1.1 About Cameo Requirements+ and Requirement Engineering 9

1.2 Cameo Requirements+ Features Overview 9

1.2.1 Schema Builder 9

1.2.2 Query Builder 10

1.2.3 Dependency Matrix 10

1.2.4 Glossary 10

1.2.5 Report Generation 10

1.2.6 Storyboard 10

1.2.7 Data Repository Merge 11

1.2.8 Multimedia Data Support 11

1.2.9 Requirement Editor 11

1.3 Cameo Requirements+ Documentation and Support 11

1.4 FAQ (Frequently Asked Question) 12

1.5 Email Support 12

2 GETTING STARTED 13

2.1 System Requirements 13

2.2 Java Virtual Machine (JVM) 13

2.2.1 Changing JVM to Run Cameo Requirements+ 14

2.2.2 JVM for Operating System Dependent Issues 15

2.3 Installation Procedure 15

2.3.1 Windows 2000/2003/XP/Vista Installation 16

2.3.2 Mac OS X Installation 19

2.3.3 Licensing Information 19

2.4 Running Cameo Requirements+ 23

2.4.1 Cameo Requirements+ Configuration File Location 24

2.4.1.1 Windows 24

2.4.1.2 Mac OS X 24

2.4.2 Cameo Requirements+ User Interface 26

2.4.3 Cameo Requirements+ Perspectives Toolbar 27

2.4.4 Default Workspace Perspective 28

2.4.4.1 Data Containment Tree 30

2.4.4.2 Editor Pane 31

2.4.4.3 Main Toolbar 33
CONTENTS

2.4.4.4 Main Menu 35
2.4.4.5 Storyboard Creation Perspective 37
2.4.4.6 Storyboard Workspace Perspective 44
2.4.4.7 Storyboard Workspace (Extended) Perspective 49
2.4.4.8 Requirement Editor Perspective 50

3 CAMEO REQUIREMENTS+ 51

3.1 Working with Cameo Requirements+ 51
  3.1.1 Using Default Schema 51
    3.1.1.1 Creating Projects 52
    3.1.1.2 Working with Sections and Requirements 55
    3.1.1.3 Working with System Boundary and Use Cases 61
    3.1.1.4 Using Rich Text Editor 65
    3.1.1.5 Using Test Suite and Test Case Nodes 67
    3.1.1.6 Creating Script 73
    3.1.1.7 Working with Revision and Version 76
    3.1.1.8 Linking Related Data through Relationship Nodes 80

3.2 Working with Data Repository 83
  3.2.1 Creating New Data Repository 84
  3.2.2 Opening Data Repository 87
    3.2.2.1 Opening Data Repository from the History Record 89
  3.2.3 Searching for Data Repository in the Open Repository Dialog 91
  3.2.4 Saving Data Repository 93
    3.2.4.1 Saving Data Repository with Different Name and in Different Location 94
  3.2.5 Closing Data Repository 95

3.3 Working with Data Containment Tree 95
  3.3.1 Using the Drag-and-Drop Feature in the Data Containment Tree 96
  3.3.2 Creating Nodes in the Data Containment Tree 96
  3.3.3 Deleting Nodes in the Data Containment Tree 97
  3.3.4 Copy, Cut, and Paste Nodes in the Data Containment Tree 98

3.4 Working with Visual Sort 98
  3.4.1 Enabling Visual Sort 98
  3.4.2 Disabling Visual Sort 99

3.5 Renumbering Nodes 99
  3.5.1 Renumbering Nodes in the Data Containment Tree 99
  3.5.2 Deleting Node Numbers in the Data Containment Tree 101
CONTENTS

3.6 Working with Export/Import 101
   3.6.1 Importing Data Repository 101
   3.6.2 Exporting Data Repository 103

3.7 Printing Active Nodes in Different Views 108
   3.7.1 Printing Node Summary View 108
   3.7.2 Printing an Entire Project Documentation 112

3.8 Working with Requirement Editor 114
   3.8.1 Using Requirement Editor Toolbar 114
      3.8.1.1 Adding a New Node 116
      3.8.1.2 Deleting a Node 117

4 CORE FEATURES 119

4.1 Cameo Requirements+ Core Features 119
   4.1.1 Working with Storyboards 119
      4.1.1.1 Storyboard Concepts 120
      4.1.1.2 Understanding Story 120
      4.1.1.3 Storyboard Container Nodes 122
      4.1.1.4 Storyboard Workspace Nodes 123
      4.1.1.5 Storyboard Libraries Nodes 124
      4.1.1.6 Scene, Action, Actor, and State Variable Collections, and Storyboard Nodes 124
      4.1.1.7 Story Nodes 124
      4.1.1.8 Storyboard Nodes 125
      4.1.1.9 Narrative Scene Nodes 126
      4.1.1.10 Decision Scene Nodes 127
      4.1.1.11 Action Template Nodes 128
      4.1.1.12 Actor Template Node 129
      4.1.1.13 State Variable Template 129
      4.1.1.14 Story Step Nodes 130
      4.1.1.15 Action Nodes 131
      4.1.1.16 Actor Nodes 132
      4.1.1.17 State Variable Nodes 132
      4.1.1.18 Region Nodes 133
   4.1.2 Creating Storyboards with Storyboard Creation Wizard 133
      4.1.2.1 Working with the Storyboard Creation Perspective 143
      4.1.2.2 Discovering Business Process Using Stories 173
      4.1.2.3 Managing Storyboard Resources with Storyboard Libraries Perspective 176
CONTENTS

4.1.2.4 Storyboard Report Templates 183

4.1.3 Working with Query Builder 189
   4.1.3.1 Creating Queries with Query Builder Basic Mode 190
   4.1.3.2 Creating Type Constraint Query in Query Builder Basic Mode 191
   4.1.3.3 Creating Attribute Constraint Queries in Query Builder Basic Mode 198
   4.1.3.4 Creating Relation Constraint Queries in Query Builder Basic Mode 205
   4.1.3.5 Editing Existing Queries in Query Builder Basic Mode 206
   4.1.3.6 Creating Queries with Query Builder Advanced Mode 207
   4.1.3.7 Creating Queries with Logical Operator in Advanced Mode 213
   4.1.3.8 Editing Existing Queries in Advanced Mode 216
   4.1.3.9 Executing Queries 217

4.1.4 Working with Dependency Matrix 217
   4.1.4.1 Creating Dependency Matrix 217
   4.1.4.2 Generating Dependency Matrix Report in CSV Format 225
   4.1.4.3 Generating Dependency Matrix Report in HTML Format 226

4.1.5 Working with Glossary 226
   4.1.5.1 Creating Glossary Terms 227
   4.1.5.2 Creating Glossary Domains 229
   4.1.5.3 Adding Definitions to Glossary Terms 230

4.1.6 Working with Report Wizard 231
   4.1.6.1 Built-In Templates 238
   4.1.6.2 Report Templates 240
   4.1.6.3 Creating and Editing Report Templates 240
   4.1.6.4 Creating and Editing Report Profile and Its Value 242

5 OTHER FEATURES 246

5.1 Cameo Requirements+ Other Features 246
   5.1.1 Working with Schema Builder 246
      5.1.1.1 Creating and Editing Node Types 249
      5.1.1.2 Schema Display Name 261
   5.1.2 Working with Schema Library Management 264
      5.1.2.1 Opening Schema Library Management 264
      5.1.2.2 Importing Schemas 264
      5.1.2.3 Exporting Schemas 265
   5.1.3 Working with Data Repository Merge 266
      5.1.3.1 Conflicting Data 273
   5.1.4 Working with Data Repository Migration Wizard 275
CONTENTS

5.1.5 Working with Restore Backup File Wizard  280
  5.1.5.1 Restoring Data Repository 280
  5.1.5.2 Restoring Schema 283
5.1.6 Working with Spelling Checker  285
  5.1.6.1 Checking Spelling Using Spelling Checker 286
  5.1.6.2 Changing Spelling Checker Preferences 287

6  CLIENT-SERVER FEATURES  290

  6.1 Cameo Requirements+ Features on Cameo Team Server  290
    6.1.1 Logging into Cameo Team Server  290
    6.1.2 Schema Builder Wizard: The Server’s Schema  292
      6.1.2.1 Creating New Schemas 292
      6.1.2.2 Editing Existing Schema 294
      6.1.2.3 Migrating Data Repository to Server’s Schema 296
    6.1.3 Schema Library Management for Remote Schema  303
      6.1.3.1 Importing Schema from Local Library 303
      6.1.3.2 Adding Schema 306
      6.1.3.3 Exporting Schema from Cameo Team Server Schema Library 308
      6.1.3.4 Deleting Schema 309
    6.1.4 Data Repository on Cameo Team Server  312
      6.1.4.1 Creating New Data Repository 312
      6.1.4.2 Opening Existing Data Repository 315
      6.1.4.3 Closing Data Repository 320
      6.1.4.4 Adding Local Repository to Cameo Team Server 321
      6.1.4.5 Converting Cameo Team Server Repository to Local Repository 324
      6.1.4.6 Managing Remote Repositories 325
      6.1.4.7 Locking Elements 338
      6.1.4.8 Editing Locked Elements 339
      6.1.4.9 Resource Lock 340
      6.1.4.10 Viewing Locked Elements in Data Repository 340
      6.1.4.11 Unlocking Elements 342
      6.1.4.12 Updating, Committing, and Discarding Changes 344
    6.1.5 Viewing Element History  346
    6.1.6 Logging out of Cameo Team Server  348
CONTENTS

7 CREATING AND IMPORTING REPORT TEMPLATES 350
7.1 Report Template Overview 350
7.2 Working with Templates 350
  7.2.1 Creating a Template 351
  7.2.2 Importing a Template to Cameo Requirements+ 352
7.3 Velocity Template Language 354
  7.3.1 Velocity Variable 355
  7.3.2 Velocity Directives 355
7.4 Report Wizard Directives 362
  7.4.1 #forrow Directive 362
  7.4.2 #forcol Directive 365
  7.4.3 #forpage Directive 367

8 APPENDICES 370
8.1 Appendix A: Cameo Requirements+ Terminology 370
8.2 Appendix B: Cameo Requirements+ Special Node Type 374
  8.2.1 Working with Image Nodes 374
    8.2.1.1 Creating Image Nodes 374
    8.2.1.2 Managing Images Using Image Viewer 376
  8.2.2 Working with Multimedia Nodes 378
    8.2.2.1 Creating Multimedia Nodes 378
1 INTRODUCTION

1.1 About Cameo Requirements+ and Requirement Engineering

Cameo Requirements+ is the 4th generation of requirement management systems. What differentiates it from the existing 3rd generation is its meta-model-based schema. Unlike other requirement management systems that collect documents and display them in a fixed structured order, Cameo Requirements+ collects data and organizes them according to the meta-model structure or schema that represent a real business model.

Cameo Requirements+ makes it possible to customize your schema, which will then enable you to correctly create requirements with respect to the business model of an organization.

Cameo Requirements+ also provides all of the necessary tools of a requirement engineering application, such as a glossary, dependency matrix, report publisher, requirements input and query, storyboard, etc. Cameo Requirements+ will provide domain experts, developers, managers, end-users, and any other stakeholders with a common platform on which to collaborate and participate in requirements planning, capturing, and managing. More importantly, it will offer them a data repository based on a completely open and scalable architecture.

1.2 Cameo Requirements+ Features Overview

You can see a detailed list of Cameo Requirements+ features at: http://www.magicdraw.com/cameo-req.

1.2.1 Schema Builder

You can use Schema Builder to build a new schema or extend an existing one by defining node types, attributes, and links. Define relationships in the schema to increase flexibility. Use any custom schema to match any real world data association - supporting linkages and relationships of any kind.
1.2.2 Query Builder

You can use Query Builder to create queries to search across the requirement repository. Use queries as a custom report generator by searching across standard criteria and isolate subsets of data based on any query criteria. Query Builder will also allow you to view relationships between subsets of the data repository and create multiple views of requirements for different needs.

1.2.3 Dependency Matrix

Cameo Requirements+ Dependency Matrix helps you analyze impacts, relationships, dependencies, gaps, test case coverage, and more. No matter what relationship you have, Dependency Matrix can show traceability, coverage, and impact analysis.

1.2.4 Glossary

Cameo Requirements+ Glossary clears uncertainty that you may have about terms for any specific domain. It supports multiple domains to allow you to add a different definition for a term in multiple domains.

1.2.5 Report Generation

The Cameo Requirements+ Report Engine provides reports on documents and summary views any time you need them. It can generate report in rtf, html, csv, and open document formats. You can also create presentation slides (Open office format) from your requirements or web-centric documentation (with navigation) to share with others.

1.2.6 Storyboard

Cameo Requirements+ Storyboard provides a sequential step-by-step illustration of a requirement scenario. It helps you get an overview of the business scenario from the high level of abstraction and capture the software logic that you are going to create.
1.2.7 Data Repository Merge

When two or more people work on the same repository, they often have to merge data. Data Repository Merge makes it possible for them to merge the work they have done into the original data repository (each of them actually works on a copy of the file). The merging process also enables them to choose which changes to be included in the original data repository.

1.2.8 Multimedia Data Support

Use Multimedia Data Support to add multimedia files and images to a requirement node. This is especially useful when a requirement is too complex to explain by text alone. Add an audio, video, or image file to compliment the requirement.

1.2.9 Requirement Editor

The Requirement editor is free workspace area in Cameo Requirements+ in which the user can review paragraphs of text and create Requirement nodes or other nodes as per the schema and preference configuration. User can import HTML documents into Cameo Requirements+ using this editor. They can further analyze the customer business requirements documents to refine to requirement nodes in Cameo Requirements+, easily by just selecting the text and marking it as requirements.

1.3 Cameo Requirements+ Documentation and Support


You can also find the pdf file of Cameo Requirements+ User Guide from the Cameo Requirements+ home folder or in the system help menu.
1.4 FAQ (Frequently Asked Question)

If you have any question, you may find the answer to your question in the FAQ section (http://www.magicdraw.com/show/faq/faq). It is constantly updated.

1.5 Email Support

You can also send us an e-mail. We provide professional e-mail support for registered Software assurance customers:

- support@magicdraw.com for product related questions (installation, usage, bugs, etc.)
- sales@magicdraw.com for academic, site discounts, sales, and marketing related questions
- contact@magicdraw.com for other issues
2 GETTING STARTED

2.1 System Requirements

To run Cameo Requirements+, your system must meet the following requirements:

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Minimum</th>
<th>Recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel® Duo Core, 2.0 GHz</td>
<td>Intel® Core i3™ or higher</td>
</tr>
<tr>
<td>Memory</td>
<td>2 GB</td>
<td>4 GB (more memory generally improves responsiveness)</td>
</tr>
<tr>
<td>Disk Space</td>
<td>400 MB</td>
<td>&gt; 400 MB</td>
</tr>
<tr>
<td>Video Mode</td>
<td>1280 * 1024 @ 16bit colors</td>
<td>1280 * 1024 @ 32bit colors</td>
</tr>
<tr>
<td>Operating Systems</td>
<td>All support Java compatible JVM 1.6 or higher. Windows XP, Vista, Windows7, and Mac OS X Snow Leopard or higher (most testing procedures and debugging were performed on these platforms).</td>
<td></td>
</tr>
<tr>
<td>Java Virtual Machine (JVM)</td>
<td>JDK 1.6</td>
<td>JDK 1.6.0_18 for Windows JDK 1.6.0_20 for Mac OS X</td>
</tr>
</tbody>
</table>

2.2 Java Virtual Machine (JVM)

JVM is an application that provides an interpretation of the Java bytecode. Different operating systems may have different JVM implementations; thus some bugs may be specific to the operating system or JVM.

JVM might have been installed on your computer. If this is not the case, you can install JVM together with Cameo Requirements+ withVM installer specifically for the Cameo Requirements+ application.
GETTING STARTED
Java Virtual Machine (JVM)

Cameo Requirements+ is a stable environment if it is configured properly and a supported, stable JVM is used. USE THE RECOMMENDED JVM TO AVOID MOST PROBLEMS. All recommendations are written in 2.2.1 and 2.2.2. Because Cameo Requirements+ is an Eclipse-based Java application, its stability and performance mostly depend on the JVM implementation. Refer to the JVM specification and problems description if you have problems with a specific OS.

JVM Version 1.6.0. or later is recommended. You can review your JVM version from the command line by typing: C:\>java -version

2.2.1 Changing JVM to Run Cameo Requirements+

You need to run Cameo Requirements+ by typing a command line in order to run Cameo Requirements+ by other JVM.

The following is a sample of a command line to run Cameo Requirements+:

<installed folder>\cameorp -vm "C:\Program Files\Java\jre1.6.0_18\bin\java"

TIPS
It is generally a good idea to explicitly specify which Java VM to use when running Cameo Requirements+. This is achieved through the "-vm" command line argument as illustrated above.

To create a Windows shortcut menu to an installed Cameo Requirements+:

1. Go to cameorp.exe in the installation folder and right-click to create a shortcut menu (if you used the installer to install Cameo Requirements+, the shortcut menu is already available through the Start menu > All Programs > Cameo Requirements+ 4.1).

2. Select the shortcut and edit its Properties. In the Target: field append the command line arguments such as <installed folder>\cameorp.exe -vm "C:\Program Files\Java\jre1.6.0_18\bin\java".
2.2.2 JVM for Operating System Dependent Issues

Cameo Requirements+ is an Eclipse-based Java application. It is platform independent and runs on a variety of operating systems. However, for all Java-based applications to interact properly with an operating system, the Java Virtual Machine (JVM) software is required. JVM varies depending on the operating system and Cameo Requirements+ may perform inadequately with the unsupported one.

The performance of Java-based applications depends on the performance of JVM. Using a not supported JVM may degrade the performance of Cameo Requirements+. It could even cause Cameo Requirements+ to fail or crash. To avoid these problems, it is recommended that you use the following JVMs:

- Sun (JDK standard) for Windows (2000, 2003, XP, and Vista) JDK 1.6.0_18
- JDK 1.6.0_18 for Mac OS X.

2.3 Installation Procedure

First, obtain the Cameo Requirements+ installation files. You can download the latest version from the Cameo Suite website (http://www.cameosuite.com). You will need a supported JVM to run the tool successfully. You can also install JVM together with the Cameo Requirements+ installer. Information about the latest Java ports is available at http://www.magicdraw.com/jvm_list.htm

Cameo Requirements+ is available in two distributions: (i) installer and (ii) no-installer.

(i) Installer Distribution

The installer distribution comes in two versions: (i) VM and (ii) NoVM. The VM version is bundled with a supported Java Virtual Machine. The NoVM is not.

(ii) No-Installer Distribution

The no-installer distribution does not come with a Java Virtual Machine and is a compressed file, the contents of which, you need to extract.

After downloading it, double-click the installer. The setup wizard will add a Cameo Requirements+ shortcut to the start menu and the desktop. You may also execute the shortcuts from the installation directory.
2.3.1 Windows 2000/2003/XP/Vista Installation

To install Cameo Requirements+ using the installer distribution with Java Virtual Machine (VM):

1. Double-click **CameoReqPlusInstall_4.1SP1_win.exe**. The **Installer Wizard** dialog will open. Click **Next** (Figure 1).

![Figure 1 -- Cameo Requirements+ Installer Wizard](image)

2. Read the license agreement and select the **I Accept the terms of the License Agreement** button. Click **Next** to proceed to the next step (Figure 2).
3. Select an installation folder and click **Next**. (The default installation folder is `c:\program files\Cameo Requirements+ 4.1 SP1`) (Figure 3).
4. Start Cameo Requirements+ by (i) choosing Start > Cameo Requirements+ 4.1, (ii) double-clicking the Cameo Requirements+ 4.1 shortcut on the desktop, or (iii) double-clicking cameorp.exe in the installation folder.

To install Cameo Requirements+ using the installer distribution without Java Virtual Machine (NoVM):

1. Double-click CameoReqPlusInstall_4_1SP1_win_NoVM.exe to run the installer.
2. Read and accept the license agreement.
3. Select an installation folder.
4. Specify the Java Virtual Machine.
5. Start Cameo Requirements+ by (i) choosing Start > Cameo Requirements+ 4.1, (ii) double-clicking the Cameo Requirements+ 4.1 shortcut on the desktop, or (iii) double-clicking cameorp.exe in the installation folder.
GETTING STARTED
Installation Procedure

To install Cameo Requirements+ using the no-installer distribution:

1. Install the Java Virtual Machine if it has not yet been installed.
2. Unzip CameoReqPlus_4_1SP1_win_no_install.zip into a directory.
3. Double-click cameorp.exe to run the application.

2.3.2 Mac OS X Installation

All Mac OS X come with a Java Virtual Machine by default.

To install Cameo Requirements+ using the installer distribution:

1. Double-click CameoReqPlusInstall_4_1SP1_mac_cocoa.zip to run the installer.
2. Read and accept the license agreement.
3. Select an installation directory.
5. Double-click the Cameo icon to start the Cameo Requirements+ application.

To install Cameo Requirements+ using the no-installer distribution:

1. Double-click CameoReqPlus_4_1SP1_mac_cocoa_no_install.tar.
2. Unzip it into a directory.
3. Double-click the Cameo icon to start the Cameo Requirements+ application.

2.3.3 Licensing Information

A license is required to install Cameo Requirements+. License keys can be of demo, evaluation, or commercial type. A demo license can be obtained from the product download section on www.cameosuite.com or by writing to sales@magicdraw.com to get an evaluation license.

When Cameo Requirements+ starts for the first time, the License Agreement Terms dialog will be displayed. Click Agree and the License Information dialog will open.
To add a license:

1. Click **Help > License > License Upgrade** (Figure 4).

![Figure 4 -- License Upgrade Menu](image)

2. Click **Agree** to accept the license agreement (Figure 5).

![Figure 5 -- Cameo Requirements+ End-User License Agreement](image)

3. Click **Browse...** in the **Product Registration** dialog to locate a license file (Figure 6).
4. Select the license key file and click Open to add the license key (Figure 7).
GETTING STARTED
Installation Procedure

Figure 7 -- Selecting the License Key File

To see the license information:

1. Click Help > License > License Information (Figure 8).
2.4 Running Cameo Requirements+

After installing Cameo Requirements+ in a directory, run the Cameo Requirements+ executable.

To run Cameo Requirements+ executable:

- On Windows – Run the executable file called cameorp.exe that is located in the installation folder or double-click the Cameo Requirements+ shortcut on the desktop if you installed it there.

- On Mac OS X - Start Cameo Requirements+ by double-clicking the shortcut or the Cameo Requirements+ application in the installation folder.
2.4.1 Cameo Requirements+ Configuration File Location

Cameo Requirements+ configuration files `camer.ini` and `camerpc.ini` are stored in the user’s home directory and Cameo Requirements+ installation home directory by default.

2.4.1.1 Windows

`camer.ini` is located in `<installation folder>/configuration`.

`camerpc.ini` is located in `<installation folder>/tools`.

2.4.1.2 Mac OS X

`camer.ini` is located in `<installation folder>/Cameo Requirements+.app/Contents/MacOS`.

`config.ini` is located in `<installation folder>/configuration`.

<table>
<thead>
<tr>
<th>NOTE</th>
<th><code>camerpc.ini</code> is not available in Mac OS X.</th>
</tr>
</thead>
</table>
### Table 2 -- cameorp.ini Command List

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-clean</code></td>
<td>To clean cached data used by the Cameo Requirements+/Cameo Requirements+ command line. Try to run once with this option if you observe startup errors after installation or update.</td>
</tr>
<tr>
<td><code>-data workspacePath</code></td>
<td>The path of the workspace on which the Cameo Requirements+/Cameo Requirements+ command line runs.</td>
</tr>
<tr>
<td><code>-configuration configurationFileURL</code></td>
<td>The location for the Cameo Requirements+/Cameo Requirements+ command line configuration file, expressed as a URL. The configuration file determines the location of the application, the set of available plug-ins, and the primary feature. Note that relative URLs are not allowed. The configuration file is written to this location when the application is installed or updated.</td>
</tr>
<tr>
<td><code>-initialize</code></td>
<td>To initialize the configuration being run. All runtime related data structures and caches are refreshed. Handy with shared installs: running the Cameo Requirements+/Cameo Requirements+ command line once with this option from an account with write privileges will improve startup performance.</td>
</tr>
<tr>
<td><code>-vmargs args</code></td>
<td>When passed to the Cameo Requirements+/Cameo Requirements+ command line, this option is used to customize the operation that Java VM used to run. The given arguments are dependent on the VM that is being run.</td>
</tr>
</tbody>
</table>

The example of args:
- `-Xms128m`
- `-Xmx512m`
- `-Dcom.sun.management.HotSpotDiagnostic`

All arguments following (but not including) the `-vmargs` entry are passed directly through to the indicated Java VM as virtual machine arguments (that is, before the class to run).

**NOTE**

If a startup argument, such as `-data`, is provided after the Java VM arguments (-vmargs), the Cameo Requirements+/Cameo Requirements+ command line will not start and you will receive a "**JVM terminated. Exit code=1**" error.
2.4.2 Cameo Requirements+ User Interface

In this chapter, you will find information about how to define Cameo Requirements+ according to each perspective, an introduction to the Cameo Requirements+ User Interface, and how to work with other features.

Cameo Requirements+ provides five perspectives to help you create requirements and storyboards, and manage storyboard resources. They are: (i) Default Workspace, (ii) Storyboard Creation, (iii) Storyboard Workspace, (iv) Storyboard Workspace (extended), and (v) Requirement Editor perspectives.

(i) Default Workspace Perspective

It is the Cameo Requirements+ default perspective. It is useful for entering and managing the requirements and their relationships.

(ii) Storyboard Creation Perspective

Use this perspective to create a storyboard.

(iii) Storyboard Workspace Perspective

Use this perspective to manage all storyboard collections/resources. You can create, remove, or modify a scene (story image), actor, action, and state variable.

(iv) Storyboard Workspace (Extended) Perspective

Use this perspective to manage all storyboard collections/resources and display all views related to the storyboard creation.

(v) Requirement Editor Perspective

This perspective is designed to work with Requirement Editor. You can easily gather and manage requirements with this perspective.
2.4.3 Cameo Requirements+ Perspectives Toolbar

To change the perspectives:

1. Click (the Open Perspective button located on the top right hand corner of the Cameo Requirements+ perspective) (Figure 9).

   ![Open Perspective Button](image)

   *Figure 9 -- Open Perspective Button*

2. Click Other … (Figure 10).

   ![Other Button](image)

   *Figure 10 -- Other Button*

3. Then select a perspective in the Open Perspective dialog (Figure 11).
2.4.4 Default Workspace Perspective

When the Default Workspace perspective is selected, a screen will be displayed (Figure 12). This default perspective screen is the main window where most actions will take place. The screen consists of five sections (Figure 12): (i) Data Containment Tree, (ii) Editor pane, (iii) Main toolbar, (iv) Main Menu, and (v) Perspective toolbar.
(i) **Data Containment Tree**

It allows you to manage a node within the data root node of Cameo Requirements+ includes viewing, deleting, and adding different types of nodes.

(ii) **Editor Pane**

It allows you to manage variables inside the selected node in the tree view.

(iii) **Main Toolbar**
GETTING STARTED
Running Cameo Requirements+

It allows you to easily perform some of the common commands in Cameo Requirements+ such as creating a new repository, loading a repository, saving a repository, etc.

(iv) Main Menu

It allows you to access all of the commands within Cameo Requirements+.

(v) Perspective Toolbar

It allows you to change to a different perspective.

2.4.4.1 Data Containment Tree

The purpose of the Containment tree is to arrange nodes (requirements, test cases, etc.) in Cameo Requirements+ into a tree structure and provide a way to manage them with respect to the schema. A typical tree in Cameo Requirements+ is shown in Figure 13. The Containment tree will allow you to modify a node on the tree only if the modification complies with the given schema. For instance, it is not allowed to add a “Report” node to the “Requirement” node in the default schema, in this case the option will be disabled.

The Data Containment tree shows (Figure 13): (i) Active data repository, (ii) Containment tree toolbar, and (iii) Visual Sort.
(i) Active data repository

The Active data repository displays the filename of the data repository that is currently open.

(ii) Containment tree toolbar

You can expand or collapse the Containment tree by clicking the toolbar buttons.

(iii) Visual Sort

The Visual Sort button allows you to arrange the tree items in either ascending or descending order.

2.4.4.2 Editor Pane

When you double-click a node, Cameo Requirements+ will open the editor pane that allows you to modify or add any variable or node to the schema. Not only does the editor pane allow you to modify or add data to the node, but it also shows in-depth information about the node itself in the summary view.
The editor pane has 4 sections (Figure 14): (i) Active node, (ii) Editor, (iii) Time stamp, and (iv) Other views tab.

(i) **Active node**

An active node is the node that is currently open.

(ii) **Editor**

The editor will update node attributes dynamically according to the ones specified in the schema. For example, if an attribute has a date data type, then the editor will provide the date picker for entering the date.

(iii) **Time stamp**

The time stamp of data entered.
(iv) Other views tab

The Other views tab allows you to review information about an active node in different views. Cameo Requirements+ provides four different views in the editor pane: Property, Table, Summary (Figure 15), and Document.

![Figure 15 -- Editor Pane Summary View in Default Perspective](image1)

2.4.4.3 Main Toolbar

The main toolbar contains buttons that allow you to create a new data repository, open, close, or save a data repository (Figure 16).

![Figure 16 -- Main Toolbar](image2)
### Table 3 -- Main toolbar buttons and Functions

<table>
<thead>
<tr>
<th>Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Icon" /></td>
<td>To create a new data repository.</td>
</tr>
<tr>
<td><img src="image2" alt="Icon" /></td>
<td>To open a data repository.</td>
</tr>
<tr>
<td><img src="image3" alt="Icon" /></td>
<td>To close an active data repository.</td>
</tr>
<tr>
<td><img src="image4" alt="Icon" /></td>
<td>To save an active data repository.</td>
</tr>
<tr>
<td><img src="image5" alt="Icon" /></td>
<td>To save an active data repository in another location.</td>
</tr>
<tr>
<td><img src="image6" alt="Icon" /></td>
<td>To undo the previous action in Cameo Requirements+.</td>
</tr>
<tr>
<td><img src="image7" alt="Icon" /></td>
<td>To redo the previous action in Cameo Requirements+.</td>
</tr>
<tr>
<td><img src="image8" alt="Icon" /></td>
<td>To add data into the Containment Tree. Alternatively, add from the “Data” menu.</td>
</tr>
<tr>
<td><img src="image9" alt="Icon" /></td>
<td>To remove a node from the Containment Tree. Alternatively, select a node in the Containment Tree and remove it from the shortcut menu.</td>
</tr>
<tr>
<td><img src="image10" alt="Icon" /></td>
<td>To cut a node from Cameo Requirements+ (in the Containment Tree, Editor Pane, etc.).</td>
</tr>
</tbody>
</table>
### 2.4.4.4 Main Menu

The main menu bar contains all commands and actions in Cameo Requirements+. Depending on the schema, some of the commands may be inactive. For example, a non-relatable node cannot use the “Add Relation” command in the “Data” menu as opposed to a relatable node. Figure 18 shows the command is disabled when a non-relatable node is selected, while Figure 19 shows the command is enabled when a relatable node is selected (Figure 17).

<table>
<thead>
<tr>
<th>Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Copy Icon]</td>
<td>To copy a node from Cameo Requirements+ (in Containment Tree, Editor Pane, Property View, etc.).</td>
</tr>
<tr>
<td>![Paste Icon]</td>
<td>To paste a node in Cameo Requirements+ (paste data back into the Containment Tree, editor pane, Property view, etc.)</td>
</tr>
<tr>
<td>![Search Icon]</td>
<td>To search into each attribute value of each node and return list of attributes which contains the input text.</td>
</tr>
</tbody>
</table>

**NOTE**

Cut, copy, and paste commands only work with a node.

*Figure 17 -- Main Menu*
Table 4 -- Main menu commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>File</td>
<td>To manage data repository files and Cameo Requirements+ main functions.</td>
</tr>
<tr>
<td>Edit</td>
<td>To copy, cut, paste, or find text.</td>
</tr>
<tr>
<td>Search</td>
<td>To search into each attribute value of each node and return list of attributes which contains the input text.</td>
</tr>
<tr>
<td>Data</td>
<td>This menu contains dynamic commands that change according to the schema on the active data repository (Figure 18).</td>
</tr>
<tr>
<td>View</td>
<td>To open perspective views.</td>
</tr>
<tr>
<td>Server</td>
<td>To open Client-Server menu.</td>
</tr>
<tr>
<td>Help</td>
<td>To open the help file.</td>
</tr>
</tbody>
</table>

Figure 18 -- Data Menu with the "Add Relation" Command Disabled
2.4.4.5 Storyboard Creation Perspective

The Storyboard Creation Perspective has four default views: (i) Data Collection, (ii) Image Editor, (iii) Properties, and (iv) Story views (Figure 20).
Figure 20 -- Storyboard Creation Perspective

(i) Data Collection View

The purpose of the Data Collection view (Figure 21) is to arrange the Storyboard nodes into a tree structure and also provide a way to manage the nodes with respect to the schema. Click a node in the Data Collection view to see its details in the other views.
(ii) Image Editor

Image Editor displays the image of narrative scene associated with the Story Step (Figure 22). You can add and remove image files, and create and edit regions from the image.
Figure 22 -- Image Editor in Storyboard Creation Perspective

Figure 23 -- Image Editor Toolbar
### Table 5 -- Image Editor buttons and functions

<table>
<thead>
<tr>
<th>Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Image Import]</td>
<td>To import a new image to the selected scene.</td>
</tr>
<tr>
<td>![Image Delete]</td>
<td>To delete an image from the selected scene.</td>
</tr>
<tr>
<td>![Image Viewer]</td>
<td>To open selected image in the image viewer.</td>
</tr>
<tr>
<td>![Region Create]</td>
<td>To create region on the selected image.</td>
</tr>
<tr>
<td>![Region Remove]</td>
<td>To remove the selected Region.</td>
</tr>
<tr>
<td>![View Minimize]</td>
<td>To minimize the Story View.</td>
</tr>
<tr>
<td>![View Maximize]</td>
<td>To maximize the view to the full screen.</td>
</tr>
</tbody>
</table>

(iii) **Properties View**

When you select a Storyboard node in the **Data Collection View** (by clicking the node), the Properties pane will allow you to modify or add any variable to the selected node. The Data Collection view shows in-depth information about a node (Figure 24).
(iv) Story View

The Story view displays a sequential story step of the selected story. You can use this view to create, delete a Story Step or play the Story, etc. You can also create a Story Step by dragging the scene from the Scene Library to the Story view (Figure 25).
Figure 25 -- Story View in Storyboard Creation Perspective

Figure 26 -- Story View Toolbar

Table 6 -- The Story view buttons and functions

<table>
<thead>
<tr>
<th>Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Icon]</td>
<td>To add a new story step.</td>
</tr>
<tr>
<td>![Icon]</td>
<td>To delete a selected story step.</td>
</tr>
<tr>
<td>![Icon]</td>
<td>To play a story.</td>
</tr>
</tbody>
</table>
2.4.4.6 Storyboard Workspace Perspective

The Storyboard Workspace perspective is mainly used to work with reusable components of the storyboard. The Storyboard Workspace perspective has six default views: (i) Data Collection view, (ii) Storyboard Libraries view, (iii) Properties tab view, (iv) Image Editor tab view, (v) Story view, and (vi) Selection view (Figure 27).
(i) Data Collection View

The purpose of the Data Collection view is to arrange the Storyboard nodes into a tree structure and also provide a way to manage the nodes with respect to the schema. Click a node in the Data Collection view to see its details in the other views. A typical Data Collection tree view is shown in Figure 21 above.

(ii) Storyboard Libraries View

This view contains four storyboard library collections (Figure 28):
(a) Scene collection: contains scene templates.

(b) Action collection: contains action templates.

(c) Actor collection: contains actor templates.

(d) State Variable collection: contains State Variable templates.

A template will be shown whenever a storyboard selection is chosen from the Libraries box (Figure 29).

---

**Figure 28 -- Storyboard Libraries View**

---

**Figure 29 -- Storyboard Libraries Collection**
GETTING STARTED
Running Cameo Requirements+

Figure 30 -- Storyboard Libraries Toolbar

Table 7 -- The Storyboard Libraries Buttons and Functions

<table>
<thead>
<tr>
<th>Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Plus Sign]</td>
<td>To add a template from the selected collection (Figure 29).</td>
</tr>
<tr>
<td>![Minus Sign]</td>
<td>To remove a selected template.</td>
</tr>
<tr>
<td>![Minus Sign]</td>
<td>To minimize the Story view.</td>
</tr>
<tr>
<td>![Maximize]</td>
<td>To maximize the Story view to full screen.</td>
</tr>
</tbody>
</table>

(iii) Selection View

This view facilitates a quick creation of story by selecting multiple scenes from Scene Library (Figure 31).
To create a new story:

1. Select the scenes from which a new story will be created.

2. Click . The Story and Step Creation Wizard dialog will open allowing you to use the selected scenes as story steps for creating a new story.
GETTING STARTED
Running Cameo Requirements+

To add to an existing story:

1. Select the scenes to add to an existing story.

2. Click . The Story and Step Creation Wizard dialog will open allowing you to add the selected scenes as new story steps to the existing story.

2.4.4.7 Storyboard Workspace (Extended) Perspective

The Storyboard Workspace (Extended) Perspective will show all views that will be use to create storyboard. This perspective has six default views: (i) Story, (ii) Data Container, (iii) Properties View, (iv) Properties Editor, (v) Scenes Flow, and (vi) Storyboard Libraries.

Figure 32 – Storyboard Workspace (Extended) Perspective
2.4.4.8 Requirement Editor Perspective

The Requirement Editor Perspective is designed for requirement gathering from the paragraphs of text and to create requirement nodes or other nodes. This perspective has three default views: (i) Data Container, (ii) Properties View, (iii) Requirement Editor.

Figure 33 -- Requirement Editor Perspective
3 CAMEO REQUIREMENTS+

3.1 Working with Cameo Requirements+

To make the most of Cameo Requirements+, you need to understand how Cameo Requirements+ works. Unlike other requirement management tools, Cameo Requirements+ uses a schema to help arrange and store information in Cameo Requirements+. A schema is like a blueprint of a business model. A node in the schema can be made to represent an actual business model. The advantages of using schemas are as follows:

- Domain experts can add constraints to a requirement type.
- A requirement type can be represented directly.
- Requirements can be arranged and represented in a more logical manner.

Cameo Requirements+ comes with a default schema; however, you still can create a new schema to fit any business model. For more information on schema creation, see Working with Schema Builder section, Chapter 5 below.

3.1.1 Using Default Schema

Cameo Requirements+ comes with a default schema (cameorequirements.schema) that is designed to help you manage your day-to-day requirement engineering activity needs. It is compatible with the SysML Requirements and Use Case diagrams.

In Cameo Requirements+, a Data node is the highest node in the entire data repository. Every node must be created under a Data node. A schema determines the structure and constraint of all allowed nodes in a data repository.

A Data node includes the following nodes from the default schema (Figure 34):

(i) **Project.** A project for your developing system.

(ii) **Storyboard Workspace.** A default workspace for your storyboard collections.

(iii) **Query Collection.** A container for all your queries.
(iv) **Matrix Collection.** A container for all your dependency matrices.

(v) **Glossaries.** A container for all terms and domains.

(vi) **Script Folder.** A container for all script nodes.

![Figure 34 -- Container Nodes under the Data Node](image)

### 3.1.1.1 Creating Projects

Cameo Requirements+ provides a project node for storing the details of your project. A project in Cameo Requirements+ represents your developing system.

To create a project in the data repository:

1. Select a Data node.
2. Either (i) right-click and select **Add Project** from the shortcut menu (Figure 35) or (ii) click **Data > Add Project** on the main menu (Figure 36).
Figure 35 -- Adding Project from Shortcut Menu

Figure 36 -- Adding Project from Main Menu

Table 9 -- Project Node Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Manager</td>
<td>The manager of the developing system.</td>
</tr>
<tr>
<td>Project Owner</td>
<td>The owner of the developing system.</td>
</tr>
<tr>
<td>Start Date</td>
<td>The starting date of the developing system.</td>
</tr>
<tr>
<td>Due Date</td>
<td>The expected due date of the developing system.</td>
</tr>
<tr>
<td>Name</td>
<td>The name of the developing system project.</td>
</tr>
<tr>
<td>Number</td>
<td>The project number if there are multiple ongoing projects.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the developing system.</td>
</tr>
</tbody>
</table>
A project node includes (Figure 38):

(i) **Section.** A container for all your requirements.

(ii) **System Boundary.** A container that represents the system boundary of your use cases.

(iii) **Storyboard Workspace.** A container for your storyboard collections (shared within the project only).

(iv) **Test Suite.** A container for all your test cases.

(v) **Glossaries.** A container for all terms and domains (shared within the project only).
3.1.1.2 Working with Sections and Requirements

The Cameo Requirements+ default schema uses sections to manage different kinds of requirements. A section represents a folder that can be used to contain requirements. The Cameo Requirements+ default schema provides a number of requirements according to SysML Requirement diagrams.

3.1.1.2.1 Section Nodes

A section is a container that allows you to categorize and manage different kinds of requirements.

To add a section in the Data Containment tree:

1. Select a project node from the Data Containment tree.
2. Right-click the project node and select Add Section from the shortcut menu (Figure 39), or click Data > Add Section (Figure 40).
3.1.1.2.2 Default Schema Requirements Nodes

All requirements must be created inside a section node. There are eight default schema requirement nodes (Figure 39):

(i) Requirement
(ii) Business Requirement
(iii) Functional Requirement
(iv) Performance Requirement
(v) Interface Requirement
(vi) Usability Requirement
(vii) Physical Requirement
(viii) Add Design Constraint
There are also a number of requirement types in Cameo Requirements+ that are supported in the SysML requirement diagram.

### Table 10 -- Requirement Types

<table>
<thead>
<tr>
<th>Requirement Types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement</td>
<td>A general requirement.</td>
</tr>
<tr>
<td>Business Requirement</td>
<td>A high-level business requirement.</td>
</tr>
<tr>
<td>Functional Requirement</td>
<td>A system functionality requirement.</td>
</tr>
<tr>
<td>Performance Requirement</td>
<td>A system performance constraint requirement.</td>
</tr>
<tr>
<td>Interface Requirement</td>
<td>A system interface (user and system interface) requirement.</td>
</tr>
<tr>
<td>Usability Requirement</td>
<td>A usability-related requirement.</td>
</tr>
<tr>
<td>Physical Requirement</td>
<td>A system physical requirement, such as the physical environment requirement.</td>
</tr>
<tr>
<td>Design Constraint</td>
<td>A system design constraint.</td>
</tr>
</tbody>
</table>

(i) Requirement Node

A Requirement Node type is the base requirement node for other requirement node types.
Table 11 -- Requirement Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The summary of the requirement.</td>
</tr>
<tr>
<td>Number</td>
<td>The requirement number.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the requirement.</td>
</tr>
<tr>
<td>Risk Kind</td>
<td>Three requirement risk types: High, Medium, and Low.</td>
</tr>
<tr>
<td>Verify Method</td>
<td>Four requirement verification methods: Analysis, Demonstration, Inspection, and Test.</td>
</tr>
<tr>
<td>Priority</td>
<td>The requirement’s priority.</td>
</tr>
<tr>
<td>Author</td>
<td>The author of the requirement.</td>
</tr>
<tr>
<td>To Do</td>
<td>A personal to-do-list for the requirement.</td>
</tr>
<tr>
<td>Note</td>
<td>Important notes for the requirement.</td>
</tr>
<tr>
<td>Relatable Node</td>
<td>The outgoing relationship to other requirements.</td>
</tr>
</tbody>
</table>

(ii) Business Requirement Node

A Business Requirement node is used for a high-level business objective. It includes all attributes from the requirement node plus two additional attributes: cost and business value.

Table 12 -- Business Requirement Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>The cost of the requirement.</td>
</tr>
<tr>
<td>Business Value</td>
<td>The business value.</td>
</tr>
</tbody>
</table>

(iii) Functional Requirement Node

A Functional Requirement node represents the functional requirement for the system functional needs and constraints. It does not contain any additional attributes.

(iv) Performance Requirement Node

A Performance Requirement node is a kind of non-functional requirement that addresses the system performance issue. It has three additional attributes: response time, throughput, and resource usage.
Table 13 -- Performance Requirement Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response Time</td>
<td>The acceptable or expected system response time.</td>
</tr>
<tr>
<td>Throughput</td>
<td>The acceptable or expected amount of materials or items passing through the system or process.</td>
</tr>
<tr>
<td>Resource Usage</td>
<td>The resource usage of the system performance.</td>
</tr>
</tbody>
</table>

(v) Interface Requirement Node

An Interface Requirement node is a kind of non-functional requirement that represents the Application Interface (API) requirement that is related to the functional requirement. It has two additional attributes: (i) protocol and (ii) technology.

Table 14 -- Interface Requirement Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>The protocol used in the system.</td>
</tr>
<tr>
<td>Technology</td>
<td>The technology of choice for the system development.</td>
</tr>
</tbody>
</table>

(vi) Usability Requirement Node

A Usability Requirement node represents the system usability-related issue. It has two additional attributes: (i) click count and (ii) usability evaluation methods.

Table 15 -- Usability Requirement Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click Count</td>
<td>The allowable click count for the user interface.</td>
</tr>
<tr>
<td>Usability Evaluation Methods</td>
<td>The usability evaluation methods.</td>
</tr>
</tbody>
</table>

(vii) Physical Requirement node

A Physical Requirement node is a non-functional requirement that represents the physical environment of the developing system. It does not have any additional attributes.
(viii) Design Constraint Node

A Design Constraint node is the design constraint of the developing system. It has two additional attributes: (i) language and (ii) constraint.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>Three supported constraint languages for you to describe the design constraint: OCL, English, and other languages.</td>
</tr>
<tr>
<td>Constraint</td>
<td>The design constraint body.</td>
</tr>
</tbody>
</table>

To add a requirement in a section:

1. Select a section node.
2. Either (i) right-click and select Add Requirement from the shortcut menu (Figure 42) or (ii) click Data > Add Requirement on the main menu (Figure 43). (You can also choose to add another requirement).
3.1.1.3 Working with System Boundary and Use Cases

A system boundary in Cameo Requirements+ represents the use case system boundary for the developing system’s use case scope. It can contain all use cases.

(i) System Boundary Nodes

Use System Boundary nodes to contain use cases that are internal to the system.

To create a system boundary node in the Data Containment tree:

1. Select a project node.
2. Either (i) right-click and select Add System Boundary from the shortcut menu (Figure 44) or (ii) click Data > Add System Boundary on the main menu (Figure 45).

**NOTE**
- You can create a requirement within another requirement.
- You can only create a requirement node inside a section node.
(ii) Use Case Nodes

A Use Case node describes what the system can do from a user's point of view.

To create a Use Case in the Data Containment tree:

1. Select a system boundary node.
2. Either (i) right-click and select Add Use Case from the shortcut menu (Figure 46) or (ii) click Data > Add Use Case on the main menu (Figure 47).

Table 17 -- System Boundary Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the system.</td>
</tr>
<tr>
<td>Number</td>
<td>The system number if there are multiple systems in the project.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the system.</td>
</tr>
</tbody>
</table>
A Use Case node has all of the requirement node attributes and 17 other additional attributes (Table 18).
Table 18 -- Use Case Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>The goal of a use case.</td>
</tr>
<tr>
<td>Subject</td>
<td>The subject of a use case.</td>
</tr>
<tr>
<td>Use Case Complexity</td>
<td>Three use case complexity types: Simple, Average, and Complex.</td>
</tr>
<tr>
<td>Assumption</td>
<td>Assumptions that apply to a use case.</td>
</tr>
<tr>
<td>Implementation Issue</td>
<td>Implementation-related issues of a use case.</td>
</tr>
<tr>
<td>Outstanding Issue</td>
<td>An outstanding issue.</td>
</tr>
<tr>
<td>Component Issue</td>
<td>Component issues, for example, to include a use case function in a component.</td>
</tr>
<tr>
<td>Extension Point</td>
<td>To describe a point in a use case where an extending use case may provide additional behavior</td>
</tr>
<tr>
<td>Pre-condition</td>
<td>An environment or condition that must be fulfilled before a use case is executed.</td>
</tr>
<tr>
<td>Post-condition</td>
<td>The result after a use case has been executed.</td>
</tr>
<tr>
<td>Basic flow description</td>
<td>A description about the shortest and simplest path to achieve the goal.</td>
</tr>
<tr>
<td>Alternative flow description</td>
<td>A description about the alternative path for a use case to achieve its goal.</td>
</tr>
<tr>
<td>Exceptional flow description</td>
<td>A description about the exceptional flow when a use case encounters an unexpected incident.</td>
</tr>
<tr>
<td>Basic flow</td>
<td>A story node that represents the shortest and simplest path for a use case to achieve its goal. (It can accept only a story node.)</td>
</tr>
<tr>
<td>Alternative flow</td>
<td>A story node’s alternative path for a use case to achieve its goal. (It can only accept a story node.)</td>
</tr>
<tr>
<td>Exceptional flow</td>
<td>A story node’s alternative path for a use case to handle an unexpected incident. (It can only accept a story node.)</td>
</tr>
<tr>
<td>Storyboard link</td>
<td>A link to the storyboard that represents a use case.</td>
</tr>
</tbody>
</table>

**NOTE** A Use Case node can only be created inside a system boundary node.
3.1.1.4 Using Rich Text Editor

You can decorate the Description attribute of each node by using a rich text editor, the new schema available in Cameo Requirements+ 4.1. It provides a formatting tool that you can use to edit the Description field in the Properties Editor and apply a style to your text (Figure 48).

The rich text editor provide 4 lines as the default editor size. You can increase the size of the rich text editor by using the increase/decrease button on the right of the editor.

Figure 48 -- Rich Text Editor

Figure 49 -- Rich Text Editor Sizing button
As default schema that bundle with Cameo Requirements+ 4.1, Rich Text field will available only for Description attribute. More text fields can be customize to enable Rich Text capability.

To turn normal Text field to Rich Text field:

1. extract the repository file to edit the schema file (.ecore) inside schema folder.
2. find the attribute that you want to customize, the attribute format should be like this

```
<eStructuralFeatures xsi:type="ecore:EAttribute" name="description">
    
</eStructuralFeatures>
```

3. add the Rich Text specific node under your selected node to define it as Rich Text field.

```
<eStructuralFeatures xsi:type="ecore:EAttribute" name="description">
    <eAnnotations source="http://www.nomagic.com/ns/cameo/UI_PROPERTY">
        <details key="richText" value="true"/>
    </eAnnotations>
</eStructuralFeatures>
```

4. add the schema file back into the repository file and reopen the repository.

To edit the inserted Image or Multimedia:

1. select the image/multimedia content in the editor.
2. click image/multimedia button. (the same button when you add this content)
3.1.1.5 Using Test Suite and Test Case Nodes

A Test Suite node in Cameo Requirements+ represents a set of test cases. For example, you may use one test suite for a functional test and the other suite for a performance test. A Test Case is a set of conditions or variables under which a tester will determine whether the developing system meets the requirements.
(i) Test Suite Nodes

A Test Suite node is a collection of test cases by categories.

To add a Test Suite node in the data Containment tree:

1. Select a project node.
2. Either (i) right-click and select Add Test Suite from the shortcut menu (Figure 50) or (ii) click Data > Add Test Suite on the main menu (Figure 51).
Table 19 -- Test Suite Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A test suite summary, for example, the Functional test suite.</td>
</tr>
<tr>
<td>Number</td>
<td>A test suite number if there are multiple test suites in a project.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of a test suite.</td>
</tr>
</tbody>
</table>

(ii) Test Case Nodes

A Test Case node contains a set of test steps and variables that the tester will use to determine whether the implementation meets the requirements.

To create a Test Case node:

1. Select a test suite node.
2. Either (i) right-click and select **Add Test Case** from the shortcut menu (Figure 52)
   or (ii) click **Data > Add Test Case** on the main menu (Figure 53).
Figure 53 -- Adding Test Case Node from Main Menu

CAMEO REQUIREMENTS+
Working with Cameo Requirements+
Table 20 -- Test Case Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of a test case.</td>
</tr>
<tr>
<td>Number</td>
<td>A test case ID.</td>
</tr>
<tr>
<td>Description</td>
<td>A detail description of a test case. For example, what the test case is trying to verify.</td>
</tr>
<tr>
<td>Objective</td>
<td>The objective of a test case.</td>
</tr>
<tr>
<td>Precondition</td>
<td>The precondition that must be fulfilled before a test case can start.</td>
</tr>
<tr>
<td>Post condition</td>
<td>The result of a test case.</td>
</tr>
<tr>
<td>Verify status</td>
<td>Four verification status types: Pass, Fail, Inconclusive, and Error.</td>
</tr>
<tr>
<td>Author</td>
<td>The author of a test case.</td>
</tr>
<tr>
<td>To Do</td>
<td>The author’s personal to-do-note.</td>
</tr>
<tr>
<td>Note</td>
<td>Notes for a test case.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing verification and trace to relationship to the requirement or implementation.</td>
</tr>
</tbody>
</table>

(iii) Test Step Nodes

A Test Step node is a node that represents a step in a test case.

To add a Test Step node:

1. Select a Test Case node.
2. Either (i) right-click and select Add Test Step from the shortcut menu (Figure 54) or (ii) click Data > Add Test Step on the main menu.
A test variable represents a variable for the test data that will be used in a test step or test case.

To add a test variable node:

1. Select a test case node.
2. Either (i) right-click and select Add Test Variable from the shortcut menu (Figure 55) or (ii) click Data > Add Test Variable on the main menu.
3.1.1.6 Creating Script

With Cameo Requirements+ you can now create a script and keep it in the script node for later execution.

(i) Script Folder Nodes

Cameo Requirements+ uses a script folder to keep all script nodes.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>A default value for a test variable.</td>
</tr>
<tr>
<td>Value Type</td>
<td>A data type of a test variable.</td>
</tr>
<tr>
<td>Value</td>
<td>The actual value of a variable.</td>
</tr>
<tr>
<td>Name</td>
<td>A variable name.</td>
</tr>
<tr>
<td>Number</td>
<td>A variable number (optional).</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of a variable.</td>
</tr>
</tbody>
</table>
To create a script node in the Data Containment tree:

1. Select a project node in the Data Containment tree.
2. Either (i) right-click and select **Add Script Folder** from the shortcut menu (Figure 56) or (ii) click **Data > Script Folder** on the main menu (Figure 57).

![Figure 56 -- Adding Script Folder Node from Shortcut Menu](image1)

![Figure 57 -- Adding Script Folder Node from Main Menu](image2)
(ii) Script Nodes

Cameo Requirements+ uses a script node to keep all executable scripts. You can create a script node under the data and test case nodes.

To create a script node in Data Containment tree:

1. Select a Script Folder node.
2. Either (i) right-click and select Add Script from the shortcut menu (Figure 58) or (ii) click Data > Add Script on the main menu (Figure 59).

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of a script folder.</td>
</tr>
<tr>
<td>Number</td>
<td>A script folder ID.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of a script folder.</td>
</tr>
</tbody>
</table>

**NOTE** You can create only one script folder for one project.
3.1.1.7 Working with Revision and Version

You can use (i) Revision and (ii) Version nodes to annotate a project version whenever it changes. Since version control is not available in Cameo Requirements+ Standard Edition, you can use these nodes to update a project version manually.

**NOTE** Version control is available in Cameo Requirements+ Server Edition.
(i) Revision History Node

The revision history node keeps all version nodes used in a project.

To create a revision from the Data Containment tree:

1. Select a project node from the Data Containment tree.
2. Either (i) right-click and select **Add Revision** from the shortcut menu (Figure 60) or (ii) click **Data > Add Revision** on the main menu (Figure 61).
A version node is used to record changes made to a project.

To create a revision for a project:

1. Select a project node from the Data Containment tree.
2. Either (i) right-click and select **Add Version** from the shortcut menu (Figure 62) or (ii) click **Data > Add Version** on the main menu (Figure 63).

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of a project revision history.</td>
</tr>
<tr>
<td>Number</td>
<td>A revision ID.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of a revision.</td>
</tr>
</tbody>
</table>

NOTE Only one revision node can be created for one project.

(ii) **Version Node**

A version node is used to record changes made to a project.
Figure 62 -- Adding Version Node from Shortcut Menu

Figure 63 -- Adding Version Node from Main Menu
3.1.1.8 Linking Related Data through Relationship Nodes

Cameo Requirements+ provides a relationship node for connecting two nodes that are related. The relationship node contains references of the source and target nodes. A Cameo Requirements+ default schema provides Relationship Editor for creating relationships the Requirement, Use Case, or Test Case nodes. Therefore, you can only create a relationship node from the Requirement, Use Case, or Test Case nodes.

3.1.1.8.1 Creating Relationships Using Relationship Editor

To create a relationship node type:

1. Double-click a Requirement or Test Case node in the tree to show its details in the Editor pane.

2. Click the Add button in Relatable Node Attributes section (Figure 64). The True Relationship Creation dialog will open.
3. Select a relationship type from the **Relations** list (Figure 65).
4. Select a target node from the **Target** list.
5. Click **Finish**.
3.1.1.8.2 Relationship Node Type

Once a relationship has been created, the relationship node will be created in the tree with the source and target nodes (Figure 66).

![Image of relationship nodes in a data containment tree]

*Figure 66 -- Relationship Nodes in the Data Containment Tree*

To view full details of a relationship node:

- Double-click a Relationship node to open its details in the Editor pane.

**NOTE**

- You can use Relationship Editor to edit the name of a relationship node.
- You cannot use Relationship Editor to edit the names of the source and target nodes.

You can only edit the name of a relationship node, but not the source and target nodes. You can edit them through Properties Editor of the Relation node.
The Cameo Requirements+ default schema provides specific fields for storing the node’s relationship details (Figure 67).

![Figure 67 -- Relationship Node in Editor Pane](image)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A relationship name.</td>
</tr>
<tr>
<td>Source</td>
<td>The available source node list that a relationship node supports.</td>
</tr>
<tr>
<td>target</td>
<td>The available target node list that a relationship node supports.</td>
</tr>
</tbody>
</table>

### 3.2 Working with Data Repository

In Cameo Requirements+, all data entered into the program are stored in a Data Repository. A Data Repository is responsible for keeping all nodes data and files imported into Cameo Requirements+. A data repository maintains consistency of the working environment even though it is open on a different machine.
3.2.1 Creating New Data Repository

To create a new data repository:

1. To create a new data repository, (i) click **File > New** (Figure 68), (ii) press **Ctrl + N**, or (iii) click on the main toolbar (Figure 69).

2. The **New Data Repository** dialog will open. Click **Browse** to select a location to save the new data repository (Figure 70).
3. A dialog will open for you to enter the **filename** and **type**. Click **Save** when done (Figure 71).
4. Select a schema to be used with the new data repository from the Schema Library table and click Finish (Figure 72).
3.2.2 Opening Data Repository

To open a data repository:

1. You can (i) click **File > Open** (Figure 69), (ii) click on the main toolbar (Figure 73), (iii) press **Ctrl + O**, or (iv) drag a **CMZIP** file to the tree and a dialog will appear, click **Open**.
2. The Open Data Repository dialog will open. Select a previous data repository from the Recently Open File table or click Browse... to select a data repository file (Figure 74).

3. Click Finish.
3.2.2.1 Opening Data Repository from the History Record

Cameo Requirement+ Data Repository History records the last 20 projects that you have opened.

**NOTE**

If a data repository in the Recently Open File table is written in red, it means that the file is no longer available or has been moved to another path (Figure 75).
To open a previous data repository file from the main menu:

1. Click **File > Recently Open**.
2. Select a data repository (Figure 76).

![Figure 76 -- Using the Recently Open Menu to Open a Data Repository](image)

To remove a data repository from history:

1. In the **Open Data Repository** dialog, select a data repository from the **Recently Open File** table.
2. Click **Remove from History** (Figure 77).

![Figure 77 -- Removing a Data Repository from History](image)

To clear the entire history record:

- Open the **Open Data Repository** dialog and click **Clear History** (Figure 78).
3.2.3 Searching for Data Repository in the Open Repository Dialog

To search for a data repository in the Open Repository dialog:

1. Open the Open Data Repository dialog and select either File Name or Location from the Search box in the Recently Open File section to search for a data repository by filename or location (Figure 79).
2. Type a keyword in the **Search** box (Figure 80).
3. Select a data repository from the Recently Open File table.

### 3.2.4 Saving Data Repository

To save a data repository:

1. To save a data repository, (i) click **File > Save** (Figure 81), (ii) click **on the main toolbar** (Figure 82), or (iii) press **Ctrl + S**.

![Figure 80 -- Searching for Data Repository by Filename](image)

![Figure 81 -- Saving Data Repository Using the Main Menu](image)
3.2.4.1 Saving Data Repository with Different Name and in Different Location

To save a data repository with a different name:

1. Either (i) click **File > Save As...** on the main menu (Figure 83) or (ii) click on the main toolbar (Figure 84).
### 3.2.5 Closing Data Repository

To close a data repository:

- You can (i) click **File > Close Data Repository** on the main menu (Figure 85), (ii) click on the main toolbar (Figure 86), or (iii) press **Ctrl + W**.

![Using the Main Menu to Close a Data Repository](image)

![Using the Main Toolbar to Close a Data Repository](image)

### 3.3 Working with Data Containment Tree

In Cameo Requirements+, all information is stored in a tree structure. The Cameo Requirements+ Data Containment tree view will help you visualize data and manipulate it. You can create, edit, delete, or complete a specific action by using the tree view dynamic menu that corresponds to a node type in the schema.
To open the tree menu:

1. Select a node in the tree.
2. Either right-click to open the shortcut menu or click Data to open the menu.

**NOTE**

You can use the tree menu to add certain types of nodes, based on a schema, under a selected node.

### 3.3.1 Using the Drag-and-Drop Feature in the Data Containment Tree

To drag a node to the target node in the Data Containment tree:

1. Click a node in the Containment tree.
2. Drag it to the target node. Depending on the schema, some target nodes will reject the selected node and no action will be committed.

### 3.3.2 Creating Nodes in the Data Containment Tree

To create a new node in the Data Containment tree:

1. Select a node in the Data Containment tree.
2. Either (i) right-click and select Add <node name> or (ii) click Data > Add <node name> on the main menu to create a node.

**NOTE**

The schema design will determine whether you can create an allowed node in any selected node. The schema node hierarchy will be used by Cameo Requirements+ as the dynamic menu and behavior constraint.
3.3.3 Deleting Nodes in the Data Containment Tree

To delete a node in the Containment tree:

1. Select a node in the tree.
2. Either right-click or click **Data** on the main menu.
3. Select **Remove <Node Name>** from the menu (Figure 87).

*Figure 87 -- Deleting the Business Requirement Node from Data Containment Tree Using the Shortcut Menu*
### 3.3.4 Copy, Cut, and Paste Nodes in the Data Containment Tree

To copy, cut, or paste a node in the Data Containment tree:

1. Select a node in the Data Containment tree.
2. Either (i) right-click and select **Cut** or **Copy**, or (ii) click **Edit > Cut** or **Copy** on the menu.
3. Select the target node.
4. Either (ii) right-click and select **Paste** or (ii) click **Edit > Paste** on the menu.

| NOTE | The Paste button may be disabled due to the constraints defined in the schema structure. |

### 3.4 Working with Visual Sort

To arrange nodes in the tree view alphabetically, you can use the visual sorting mode provided by Cameo Requirements+. The visual sorting mode can arrange the nodes alphabetically in ascending or descending order.

#### 3.4.1 Enabling Visual Sort

To enable the visual sorting mode:

1. Click the **Visual Sort** button on the top of the Data Containment tree.
2. Select a sorting type: **Ascending by label** or **Descending by label** (Figure 88).

*Figure 88 -- Visual Sort Button is Enabled*
3.4.2 Disabling Visual Sort

To disable the visual sorting mode:

- Click the depressed **Visual Sort** button on top of the tree (Figure 89).

![Figure 89 -- Visual Sort Button is Disabled](image)

3.5 Renumbering Nodes

The Cameo Requirements+ renumbering feature allows you to automatically number each data node based on the tree structure (applies to any node type with a "number" field in it).


3.5.1 Renumbering Nodes in the Data Containment Tree

To enable nodes numbering in the Data Containment tree:

1. Right-click a node in the tree.
2. Select **Renumber Subtree** from the shortcut menu.
3. Select a renumbering type (Figure 90).
Figure 90 -- Selecting Renumbering Type from the Shortcut Menu
3.5.2 Deleting Node Numbers in the Data Containment Tree

To clear nodes numbering in the Data Containment tree:

1. Right-click a node in the tree.
2. Select **Renumber Subtree > Clear numbering**.

<table>
<thead>
<tr>
<th>Renumbering type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numbers</td>
<td>Integer numbers. For example, 1, 2, 3, 4…</td>
</tr>
<tr>
<td>Lowercase letters</td>
<td>Lowercase letters. For example, a, b, c, d…</td>
</tr>
<tr>
<td>Capital letters</td>
<td>Uppercase letters. For example, A, B, C, D…</td>
</tr>
<tr>
<td>Roman numerals</td>
<td>Roman numerals. For example: I, II, III, IV…</td>
</tr>
<tr>
<td>Roman numerals lowercase</td>
<td>Lowercase Roman numerals. For example: i, ii, iii, iv…</td>
</tr>
</tbody>
</table>

**NOTE**
You can only renumber a node that has an attribute name number.

3.6 Working with Export/Import

Cameo Requirements+ can import and export both data repository and schema. The schema is a `.schema` file, the standard file format for Cameo Requirements+ schema files. The data repository is a `.cmzip` file, the standard file format for Cameo Requirements+ archives.

3.6.1 Importing Data Repository

Cameo Requirements+ can import and export data repositories. It can import a data repository from a `.cmzip` file to the current data repository or export all or part of the data repository to a `.cmzip` file.
To import a data repository:

1. Click **File > Import Data Repository...** (Figure 91). The **Import Data Repository** dialog will open.

   ![Figure 91 -- Importing a Data Repository](Image)

2. Select the file to be imported from the **Import Data Repository File** dialog and click **Open** (Figure 92).
3. Drag the `.cmzip` file from computer file system to the Data Containment tree.

| NOTE | When importing a data repository to an active data repository in Cameo Requirements+, both `.cmzip` files must use the same schema, otherwise only nodes that match the active data repository schema will be imported. |

### 3.6.2 Exporting Data Repository

Cameo Requirements+ supports two types of export:
(i) Export as Baseline (this will export the nodes with the Baseline status).

(ii) Export normally (in this case, the nodes will have the same change status as before the export).

**NOTE**
See Working with Data Repository Merge section below for more information on Export as Baseline.

(ii) To export a data repository normally (Export normally):

1. Select a node in from the tree or press **Ctrl** to select several nodes.
2. Either (i) right-click and select **Export...** (Figure 93) or (ii) click **File > Export Data Repository...** (Figure 94).
Figure 93 -- Exporting Data Repository Using the Shortcut Menu
3. The **Export Data Repository** dialog will open. Clear the **Export as Baseline** check box and click **Browse** to select a location to export and save the file (Figure 95).
4. The Export Data Repository File dialog will open (Figure 96). Select a location, type a filename, and select a file type.

5. Click Save.
3.7 Printing Active Nodes in Different Views

Cameo Requirements+ allows you to print an active node in four different views: (i) properties, (ii) table, (iv) summary, and (v) document views.

3.7.1 Printing Node Summary View

The Summary view in the editor pane shows a detailed description of a node.
To print a summary view:

1. Double-click a node.
2. Click the **Summary** view tab in the lower part of the editor pane.
3. Either (i) click the **Print** button on the toolbar (Figure 97) or (ii) click **File > Print** (Figure 98).
The system should provide security to monitor the user activity.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>Sam Rice</td>
</tr>
<tr>
<td>To Do</td>
<td>Need to define more information about the user profile.</td>
</tr>
<tr>
<td>Note</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>The system should provide security to monitor the user activity.</td>
</tr>
<tr>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>Risk Kind</td>
<td>Low</td>
</tr>
<tr>
<td>Verify Method</td>
<td>Test</td>
</tr>
<tr>
<td>Priority</td>
<td>Highest</td>
</tr>
<tr>
<td>Cost</td>
<td></td>
</tr>
<tr>
<td>Business Value</td>
<td>Ensure the security of the system.</td>
</tr>
</tbody>
</table>

Figure 97 -- Printing Node Summary View from the Editor Pane Toolbar
4. The **Print** dialog will open. Select a printer and click **Print** (Figure 99).
3.7.2 Printing an Entire Project Documentation

The Cameo Requirements+ Document view provides a complete documentation of all the requirements in a project. The requirements coverage in the document view depends on the selected node, for example, if you select a business requirement node, you will then have the selected business requirement information presented in the Document view. However, if you select a containment, section, or project node, you will then get all the sub nodes of the selected containment node shown in the Document view.

To print a node in the Document view:

1. Double-click the project node in the Data Containment tree.
2. Click the Document view tab in the editor pane.
3. The entire project with all its containment nodes and sub nodes will open in the Document view (Figure 100).

4. Either (i) click the **Print** button on the editor pane toolbar or (ii) click **File > Print**. The **Print** dialog will open.

5. Select a printer and click **Print**.
3.8 Working with Requirement Editor

Cameo Requirements+ provides a text editor that allows you to type in or paste some text from other sources. Requirement Editor has the capability to capture some specific text and turn it into a requirement. You can also use Requirement Editor to edit an existing requirement.

To open Requirement Editor:

- Right-click a node in the Containment tree and select **Open with > Requirement Editor**. Requirement Editor will open (Figure 101).

3.8.1 Using Requirement Editor Toolbar

The Requirement Editor toolbar (Figure 102) contains buttons that allow you to select a specific font style for a node, create position hierarchy for a new node, add external resources, and convert some text into Rich Text Format.
Figure 102 -- Requirement Editor Toolbar
Table 28 – Requirement Editor Toolbar Functionalities

<table>
<thead>
<tr>
<th>Drop-down list/Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Node Type</strong>: to change some text into a node according to the selected type.</td>
<td></td>
</tr>
<tr>
<td><strong>Field Type</strong>: to create position hierarchy for a new node, for example, if you select (i) <em>Heading 1</em>, the new node will be created at the same level as the selected node in the Containment tree or (ii) <em>Heading 2</em>, the new node will be created as the child of the selected node in the Containment tree.</td>
<td></td>
</tr>
<tr>
<td>To apply font styles on the text that represents a node.</td>
<td></td>
</tr>
<tr>
<td>To insert a table, image, media file, or hyperlink, and to remove a hyperlink. To edit the inserted image or media file, select at the image or the media in the editor, then click the image/media button.</td>
<td></td>
</tr>
<tr>
<td>To import an HTML document into the editor.</td>
<td></td>
</tr>
<tr>
<td>To delete a node from the editor.</td>
<td></td>
</tr>
<tr>
<td>To toggle the view that will show Field Type in each section.</td>
<td></td>
</tr>
</tbody>
</table>

3.8.1.1 Adding a New Node
To add a new node using Requirement Editor:

1. Open Requirement Editor.
2. Select the text to be created as the requirement name.
3. Select **Node Type**.
4. Select **Field Type**.
5. Click **Save**.

### 3.8.1.2 Deleting a Node

To delete node using Requirement Editor:

1. Right-click a node in the Containment tree and select **Open with > Requirement Editor**.
2. Select the text that will represent the node.
3. Right-click the selected text and select **Remove Node Recursively**. The node and all its children will be deleted.
**NOTE**

- Keyboard Short-cuts for Cut, Copy and Paste do not support yet on Rich Text editor and Requirement Editor. You can perform Cut, Copy and Paste operation via key combination Ctrl+X, Ctrl+C and Ctrl+V.

- Heading style will not apply to the selecting text if this text have set font size. Font size setting have to removed first, the heading style will be applied. You can remove font size setting by clicking on Remove Format button on Rich Text toolbar.

- Font size/type will not apply on Rich Text editor and Requirement Editor if after selecting the style, you clicking on the text field. You can apply the style to text by

  1. After selecting the style, you start entering the text without clicking on the text field.

  2. Highlighted text on the text field and apply the style to it, from this point the style will be applied follow the new style.

- Keyboard Home / End on Mac are working like Page-up / Page-down on Rich Text editor and Requirement Editor.

- Multimedia on Rich Text editor and Requirement Editor will display, if Spell Checker on Rich Text editor and Requirement Editor is active. You can activate Spell Checker by clicking on Spell Checker button on Rich Text toolbar.
4 CORE FEATURES

4.1 Cameo Requirements+ Core Features

Cameo Requirements+ core features provide major functionalities to help you with your requirements engineering and data, and Cameo Requirements+ data management. Since the features work with a specific schema structure, it is necessary that you understand the schema or the sequence of the schema structure to avoid problems concerning dysfunctional features. This section will guide you through all the features and related schemas.

4.1.1 Working with Storyboards

Cameo Requirements+ Storyboard provides a sequential step-by-step illustration of a requirement scenario. The purpose is to give you an overview of the system that you are going to create.

A storyboard is a way in XP programming of eliciting needs or requirements from stakeholders at the requirements gathering phase. It helps business or system analysts gather and refine requirements, and make dynamic prototypes in a user-friendly way. A storyboard will improve communication among customers, managers, end-users, and developers, and help them develop a common understanding in terms of requirements by defining requirements clearly, unambiguously, and thoroughly. Thanks to visual scenarios, now every stakeholder can easily take part in the requirement validation process.

A storyboard can also help system analysts develop system prototypes and understand the business scenario and process. In order to capture business requirements, Cameo Requirements+ Storyboard provides a story player that will play the real scenario as a story and part of the system activity.

A storyboard consists of a sequence of story steps that consists of descriptions, images, audio files, actors, events, actions, and so on. It tells you what will happen at each point, how, and when it will happen, and who the players are.

A story represents a specific, concrete instance of your interaction with a system in order to achieve a specific goal. Stories lie at the low end of the abstraction scale. You as an analyst can abstract upward to generalize a story or a set of similar stories, or scenarios, to cover a variety of mailing label possibilities within the same use case.
Stories and scenarios turn out to be powerful ways of hearing and understanding a customer’s side, and valuable tools to capture his/her goals and visions.

4.1.1.1 Storyboard Concepts

Cameo Requirements+ perspectives can help you create a storyboard. A perspective is a visual container for a set of views and editors (parts). The layout of views and editors contained in each perspective can be reorganized to suit your specific needs. Cameo Requirements+ provides two perspectives: Storyboard Creation and Storyboard Workspace, and two major wizards: Story Creation Wizard and Action Creation Wizard, to help you create a story and action and manage the storyboard resources.

4.1.1.2 Understanding Story

A story is a sequence of events that happens in one or multiple scenes. Actors may be needed to make decisions at one of the story steps. They always make choices and continue with the story. For example, a manager may describe an e-mail system security login in two different stories:

Story 1: when a user has successfully logged in to the system.

Story 2: when a user failed to log in and rejected by the system.

Thus, a story step must have references to a scene to indicate when and where the step happens. At every step you can define who the subject (actor) of the story is at a given time (step) and the action that the actor will take. If you want to go further, you can also describe whether or not the action has an effect on a state variable that may hold a value.

For example, Sam is in the room with an air conditioner and the room temperature is 35 degrees Celsius. Sam turns on the air conditioner and sets the temperature at 25 degrees Celsius. The air conditioner will start and keep on going until the room temperature goes down to 25 degrees Celsius then it will stop. Table 28 shows the Cameo Requirements+ representation of this simple story:
As a business analyst you may not want to go through the very detail of the story, but as a system analyst, you will want to get into details as much as possible. Cameo Requirements+ provides a Story Creation Wizard to accommodate a business analyst’s needs by helping him/her describe his/her story in the simplest way. However, at the same time it also makes it possible for a system analyst to use the Storyboard Creation perspective to define as much technical information as needed (actors, state variables, and action, etc.) and create links in the story, steps, state variables, actors, and action by using requirements, use cases, etc.

It is always easier for a non-technical manager to describe business processes or needs by telling a story. Cameo Requirements+ Storyboard provides the business process recognition to help a business analyst discover problems in the story told by a customer.

You can create a storyboard by using: (i) Storyboard Creation wizard, (ii) Storyboard Creation perspective, and (iii) Default perspective.

(i) Storyboard Creation Wizard

The Storyboard Creation wizard provides the easiest way to create a storyboard. The wizard will take care of all the node dependencies and prerequisites that you need in order to create a storyboard.

(ii) Storyboard Creation Perspective

The Storyboard Creation perspective is a default perspective to work with any storyboard. You must have a basic understanding of how the storyboard and its nodes work.
(iii) Default Perspective

You must have a very good understanding of how the storyboard and its nodes work in order to build the storyboard with a default perspective.

4.1.1.3 Storyboard Container Nodes

All storyboard nodes can only be created within each container node.

![Figure 103 -- Storyboard Structure from the Default Perspective](image)
### 4.1.1.4 Storyboard Workspace Nodes

This is the default workspace for all related stories and shareable resources.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of Storyboard Workspace.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of Storyboard Workspace.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship that the workspace links to. It supports both derived and TraceTo relationships.</td>
</tr>
</tbody>
</table>

---

**Table 30 -- Storyboard Container Nodes**

<table>
<thead>
<tr>
<th>Storyboard Nodes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storyboard Workspace</td>
<td>The main container that contains everything for the storyboard.</td>
</tr>
<tr>
<td>Storyboard Libraries</td>
<td>The container that contains all sharable storyboard resources such as Scenes, Actions, Actors, and State Variables.</td>
</tr>
<tr>
<td>Scene Collection</td>
<td>A container that contains all scenes that can be reused by all stories within the same workspace.</td>
</tr>
<tr>
<td>Action Collection</td>
<td>A container that contains all actions that can be reused by all stories within the same workspace.</td>
</tr>
<tr>
<td>Actor Collection</td>
<td>A container that contains all actors that can be reused by all stories.</td>
</tr>
<tr>
<td>State Variable Collection</td>
<td>A container that contains all state variables that can be reused by all stories.</td>
</tr>
<tr>
<td>Storyboard</td>
<td>The container that contains all stories. This node must be created in Storyboard Workspace.</td>
</tr>
<tr>
<td>Story</td>
<td>A story that contains a set of story steps.</td>
</tr>
</tbody>
</table>
4.1.1.5 Storyboard Libraries Nodes

Table 32 -- Storyboard Libraries Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the storyboard libraries.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the storyboard libraries.</td>
</tr>
</tbody>
</table>

4.1.1.6 Scene, Action, Actor, and State Variable Collections, and Storyboard Nodes

Scene, Action, and State Variable collections, and Storyboard nodes share the same attributes.

Table 33 -- Scene, Action, Actor, and State Variable Collections, and Storyboard Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the node.</td>
</tr>
<tr>
<td>Number</td>
<td>The node number.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the node.</td>
</tr>
</tbody>
</table>

4.1.1.7 Story Nodes

A story node is used to contain all stories.

Table 34 -- Story Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the story.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the story.</td>
</tr>
<tr>
<td>Author</td>
<td>The author of the story.</td>
</tr>
<tr>
<td>Owner</td>
<td>The owner of the story or the one who tells the story.</td>
</tr>
<tr>
<td>Image</td>
<td>The story image (optional).</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship that the workspace links to. It supports both derived and TraceTo relationships.</td>
</tr>
</tbody>
</table>
4.1.1.8 Storyboard Nodes

Figure 98 shows the storyboard nodes in the Data Containment Tree.

Figure 104 -- Storyboard Nodes in the Data Containment Tree
Table 35 -- *Storyboard Nodes*

<table>
<thead>
<tr>
<th>Storyboard Nodes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Narrative Scene</td>
<td>A scene that represents a normal process or procedure. It can contain an image. A scene gives us an environment for the steps to describe an action.</td>
</tr>
<tr>
<td>Decision Scene</td>
<td>A scene that represents a decision and choice that an actor has to make. It does not contain any image.</td>
</tr>
<tr>
<td>Action template</td>
<td>A detailed description of the action that actors will perform at a particular step.</td>
</tr>
<tr>
<td>Actor template</td>
<td>An actor template represents the role that an actor plays at a given step.</td>
</tr>
<tr>
<td>State Variable template</td>
<td>The state variable represents a public variable, the value of which can be redefined by the state variable at a step.</td>
</tr>
<tr>
<td>Story Step</td>
<td>The story steps.</td>
</tr>
<tr>
<td>Action</td>
<td>Actions performed by an actor at a particular step.</td>
</tr>
<tr>
<td>Actor</td>
<td>A person or subject that performs at a particular step.</td>
</tr>
<tr>
<td>State Variable</td>
<td>The state variable represents a variable for a scene that is changeable during the story step action.</td>
</tr>
</tbody>
</table>

4.1.1.9 Narrative Scene Nodes

A Narrative Scene represents the environment where a story step takes place. A story may use one or more narrative scenes. A narrative scene can contain an image that can be used at a later step as the story step background. It is shareable among story steps in the same workspace.

You can also use a narrative scene to represent every step in the story if you have a sequence of screen shots that represents the user’s interactivity with the system. For this purpose you have to use a narrative scene to keep the images and used them at every step.

A narrative node is sharable among stories in the same workspace.
Table 36 -- Narrative Scene Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
<td>An image in the scene.</td>
</tr>
<tr>
<td></td>
<td>An audio file. It will be a default audio for a step that has reference to</td>
</tr>
<tr>
<td></td>
<td>a scene in which an audio file is added. However, if the step has its own</td>
</tr>
<tr>
<td></td>
<td>audio file then the audio will overwrite the scene’s audio file.</td>
</tr>
<tr>
<td>Name</td>
<td>The name of the scene.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the scene.</td>
</tr>
<tr>
<td>Category</td>
<td>The category of the scene. It will be used for the Scene collection filtering</td>
</tr>
<tr>
<td></td>
<td>in the Storyboard libraries perspective.</td>
</tr>
<tr>
<td>Story step</td>
<td>A list of story steps that have references to narrative scenes.</td>
</tr>
<tr>
<td>Outgoing relations</td>
<td>The outgoing relationship which the workspace links to. It supports both</td>
</tr>
<tr>
<td></td>
<td>derived and TraceTo relationships.</td>
</tr>
</tbody>
</table>

4.1.1.10 Decision Scene Nodes

A decision scene node represents a choice that an actor has to make in a story. In stories, actors always make choices. So a decision node is to represent the decision that the actor has to make. It is shareable among all story steps in the same storyboard workspace.

A story step can also have references to a decision scene instead of a narrative scene. You use a decision scene for a story step when you know that at a particular step the actor is actually making a decision.

| NOTE             | In a story, actors always make decisions at the step that has a decision scene. |
**CORE FEATURES**

Cameo Requirements+ Core Features

---

### Table 37 -- Decision Scene Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guard Values</td>
<td>Specific conditions that must be fulfilled in order to pass a decision (optional).</td>
</tr>
<tr>
<td>Name</td>
<td>The name of the decision.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the decision.</td>
</tr>
<tr>
<td>Category</td>
<td>The category of the scene. It will be used for the Scene collection filtering in the Storyboard libraries perspective</td>
</tr>
<tr>
<td>Story step</td>
<td>A list of story steps that have references to decision scenes.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship which the workspace links to. It supports both derived and TraceTo relationships.</td>
</tr>
</tbody>
</table>

### 4.1.1.11 Action Template Nodes

An action template node represents a task or a function an actor performs at a particular step. It also represents the action structure and business logic. It is shareable among all actions within the same storyboard workspace.

---

### Table 38 -- Action Template Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the action.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the action.</td>
</tr>
<tr>
<td>Category</td>
<td>The category of the action template that will be used by the storyboard libraries filtering.</td>
</tr>
<tr>
<td>Action body</td>
<td>A description of the action that can be written in either programming languages or non-programming languages, for example, English.</td>
</tr>
<tr>
<td>Language</td>
<td>The language of choice for the action body (free text).</td>
</tr>
<tr>
<td>Actions</td>
<td>A list of actions that the main action will call.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship which the workspace links to. It supports both derived and TraceTo relationships.</td>
</tr>
</tbody>
</table>
4.1.1.12 Actor Template Node

An Actor Template node represents an actor role or an actor in a general abstract form. The actor that is being described at a story step will always be performing a role in the context of the story. For example, Sam can be a manager at the office and a father at home; therefore, two different actor templates will be used to represent two different roles in the story.

Table 39 -- Actor Template Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role</td>
<td>The role that an actor plays.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the role.</td>
</tr>
<tr>
<td>Category</td>
<td>The category of the actor template that will be used by the storyboard</td>
</tr>
<tr>
<td></td>
<td>libraries filtering.</td>
</tr>
<tr>
<td>Image</td>
<td>The image of the actor role.</td>
</tr>
<tr>
<td>Actors</td>
<td>A list of all actors that have references to an actor template.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship which the workspace links to. It supports</td>
</tr>
<tr>
<td></td>
<td>TraceTo relationship.</td>
</tr>
</tbody>
</table>

4.1.1.13 State Variable Template

The state variable template represents a public state variable that can be redefined later by the state variable at a story step. It is sharable among all state variables at a story step.
Table 40 -- State Variable Template Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the state variable template.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of the state variable.</td>
</tr>
<tr>
<td>Category</td>
<td>A category of the state variable template that will be used by the story-</td>
</tr>
<tr>
<td></td>
<td>board libraries filter.</td>
</tr>
<tr>
<td>Default Value</td>
<td>A default value that the state variable will use.</td>
</tr>
<tr>
<td>Value type</td>
<td>A data type for the variable.</td>
</tr>
<tr>
<td>State Variables</td>
<td>A list of state variables that have references to a state variable tem-</td>
</tr>
<tr>
<td></td>
<td>plate.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship which the workspace links to. It supports TraceTo relationship.</td>
</tr>
</tbody>
</table>

4.1.1.14 Story Step Nodes

A story step represents an actual step in a story. It must have references to either a narrative scene or a decision scene because every story step must take place either in a narrative or a decision scene. A story step can also contain multiple Action nodes.
CORE FEATURES
Cameo Requirements+ Core Features

Table 41 -- Story Step Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of a story step.</td>
</tr>
<tr>
<td>Scene</td>
<td>A particular scene which a story step has references to. It is absolutely necessary. If a story step has no reference to a scene, some features in the storyboard may not function correctly.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of a story step.</td>
</tr>
<tr>
<td>Audio File</td>
<td>An audio file embedded in a story. It starts when the story plays. When you add an audio file to a story step, the delay time will be overwritten by the duration of the audio file. A default audio file that comes with the narrative scene will be overwritten as well.</td>
</tr>
<tr>
<td>Time Delay (sec)</td>
<td>It is the delay time in the story player. It counts in seconds.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship which the workspace links to. It supports Derived, TraceTo, and StepTransition relationships.</td>
</tr>
</tbody>
</table>

NOTE
Cameo Requirements+ uses StepTransition to link each story step when you compose a story. When you delete a story step, the StepTransition link to that step will be removed automatically.

4.1.1.15 Action Nodes

An action node is the representation of an action that will be performed at a story step. An action can contain multiple actors and state variables. Cameo Requirements+ provides the Action Creation wizard to help you define and create an action at every story step.
Table 42 -- Action Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action template</td>
<td>An action always has references to an action template that contains a detailed description of the action. Action templates are mandatory.</td>
</tr>
<tr>
<td>Input</td>
<td>An input data.</td>
</tr>
<tr>
<td>Result</td>
<td>The result of an action.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The outgoing relationship which the workspace links to. It supports TraceTo relationships.</td>
</tr>
</tbody>
</table>

4.1.1.16 Actor Nodes

An Actor node represents an actual actor that takes part in a story step. Every actor is supposed to pick a role that relates to each story step.

Table 43 -- Actor Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>An actor’s name, for example, Sam or air conditioner.</td>
</tr>
<tr>
<td>Actor template</td>
<td>An actor template that the actor has references to. An Actor template also has information about the role that the actor is currently playing.</td>
</tr>
<tr>
<td>Actor complexity</td>
<td>The actor complexity includes the complexity of his interaction with the system or the story.</td>
</tr>
</tbody>
</table>

4.1.1.17 State Variable Nodes

Actions use state variables at story steps to redefine the state variable template value.

Table 44 -- State Variable Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State Variable template</td>
<td>The reference of a state variable template. It is mandatory.</td>
</tr>
<tr>
<td>New value</td>
<td>A new redefined value.</td>
</tr>
</tbody>
</table>
4.1.1.18 Region Nodes

A Region node is used to define images in a narrative scene. It has a dependency relationship with the image in the narrative scene. If you change or remove an image from a narrative scene, all region nodes in that scene will be removed as well.

Table 45 -- Region Node Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of a region.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of a region.</td>
</tr>
<tr>
<td>Outgoing Relations</td>
<td>The node relationship with other nodes.</td>
</tr>
</tbody>
</table>

4.1.2 Creating Storyboards with Storyboard Creation Wizard

As a business analyst you may want to create a storyboard right away without worrying about the detail and structure of the Cameo Requirements+ storyboard. The Storyboard Creation wizard is designed to help you create a story faster.

To create a storyboard using the Storyboard Creation wizard:

1. Right-click the Root node or Project node in the Data Containment Tree.
2. Select Add Storyboard Workspace Using Wizard from the shortcut menu (Figure 105) or click Data > Add Storyboard Workspace Using Wizard (Figure 106).
Figure 105 -- Storyboard Workspace Wizard in the Shortcut Menu

Figure 106 -- Storyboard Workspace Wizard in the Main Menu
3. Type the name, description, author, and owner (Figure 107).

4. Click to import an image. The Select Image file... dialog will open (Figure 108).
5. Select an image and click **Open**. The selected image will be displayed in the image area of the **Story Creation** wizard (Figure 109).
Figure 109 -- Step 1 Story Creation Wizard Completed

6. Click Next. The Specify Location page of the Story Creation Wizard dialog will open (Figure 110).
7. There are two options to save your story, either select (i) Add to new Storyboard Workspace or (ii) Add to existing Storyboard workspace. Select the Add to new Storyboard Workspace button to create a new storyboard workspace. Type the story name and description.

8. Click Next. The Create Steps page of the Story Creation wizard will open (Figure 111).
9. Double-click the **Story Steps NARRATIVE** image to add an image to a story step (Figure 112). The **Nodes Reusing** page will open. You can select an existing image for a story step or add a new one to the Reuse node for future reuse in another story.
CORE FEATURES
Cameo Requirements+ Core Features

10. Click **Add Image** to add a single image for the story step. The **Select Image file**... dialog will open (Figure 113).
Figure 113 -- Adding an Image to the Story Step in the Reuse Nodes Dialog

11. Select an image and click Open. The Nodes Reusing page will open and display the selected image.
12. Click Finish. The Story Creation wizard will open and display the selected image.
13. Type the name and description of the story (Figure 114).
14. To add another image, click **Add Step** and repeat steps 10 to 14 (Figure 115).

15. When you finish creating all story steps, click **Finish** to continue with the **Storyboard Creation** perspective.
4.1.2.1 Working with the Storyboard Creation Perspective

The Cameo Requirements+ Storyboard Creation perspective (Figure 116) is designed to help you create, edit, and delete stories. After you finish creating your initial story with the Storyboard Workspace wizard (Story Creation wizard), Cameo Requirements+ will open the Storyboard Creation perspective to let you continue working on your story.

You can use the storyboard perspective to create an entire new story and manage all other stories. This section will show you how to add a story step with narrative and decision scenes, play the story with a story player, see the suggested business process with scene and action flows, and so on.
CORE FEATURES
Cameo Requirements+ Core Features

(i) Creating Story Steps with Narrative Scene

To add more steps to a story that has just been created using the Story Creation wizard:

1. Select a story from the Data collection view within the Storyboard Creation perspective (Figure 116).
2. Click **Add Step** in the story view to add a new step (Figure 117).

3. Enter the Name, Description, Audio file, and Time Delay for the story step in the Properties view (Figure 118). See section **4.1.1.14** for more information on the story step node.
4. Click ![Icon](image) to add a detailed description of the scene (a Narrative or Decision scene). In this example, a narrative scene is selected. Each scene represents a story step image.
Figure 119 -- Story Step in the Properties View
5. The **Properties** view will open the **Scene Node** view for you to add detailed information about the scene that will be reused and referenced by the story step (Figure 120).

*Figure 120 -- Scene Node in the Properties View*
6. Type the name, description, and category for the scene properties.
7. Click the **Import File** button to add an image (Figure 121).

![Figure 121 -- Adding an Image to the Scene Using the Import File Button](image)

8. Click the **Back** button on the top of the **Properties** view to go back to the story step properties and continue (Figure 122).

![Figure 122 -- Returning to the Story Step Properties View](image)

9. A story step with the Storyboard perspective will be created.
(ii) Creating Story Steps with Decision Scene

A Decision Story Step represents a decision made within a story. A decision scene is a scene without an image. Every story step that represents the decision will have reference to the decision scene.

To create a story step with a decision scene:

1. Select a story from the Data collection view within the Storyboard Creation perspective.
2. Click the Add Step button in the story view to add a new step.
3. Fill in the Name, Description, Audio file, and Time Delay for the story step in the Properties view. See section 4.1.1.14 for more information about the story step node.
4. Enter a decision scene in the Scene box (Figure 123).

5. Enter the Name, Description, Audio file, and Time Delay for the step properties.
Figure 124 -- Decision Scene in the Properties View
6. A story step with a decision scene will be created in the Storyboard perspective.

| NOTE | Create a decision scene in the Storyboard Libraries perspective before using it for a story step. |

(iii) Creating Action for Story Step Using Action Creation Wizard

An action is something that is being performed by an actor in a story step. A user entering a login name in the login field is an example of an action. An action can be performed by different actors at different scenarios. For example, when a user with a business analyst role enters a username, it illustrates an action performed by an Actor with the Business Analyst role. At the same time another user with an architect role can login to the same login page, in this case the actor is the Architect. Therefore, different actors can perform the same action.

To create an action for a story step:

1. Select a story step node and click the Properties view.
2. Go to the Additional features section in the Properties view (Figure 125). The Additional features pane allows you to add an action to a story step.
3. Click . The **Action Creation** wizard will open to help you create an action.

4. Select either **Add new Action Template** or **Use existing Action template** in the **Create Action** page of the **Action Creation** wizard. In this example **Add new Action Template** will be selected to create a new action (Figure 126).
5. Click **Next**. The **Action Information** page will open (Figure 127).
6. Fill in the Name, Description, Action body, Language, Input, and Result boxes, and then click **Next**. The **Create Actor** page will open.
Table 46 -- Action Template Attributes

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>An Action Template name. It is a mandatory field.</td>
</tr>
<tr>
<td>Description</td>
<td>A detailed description of an Action Template.</td>
</tr>
<tr>
<td>Action Body</td>
<td>It can be used to enter the logic of the action or its source code.</td>
</tr>
<tr>
<td>Language</td>
<td>The coding language used to implement an action like Java, C++, or English.</td>
</tr>
<tr>
<td>Input</td>
<td>An input to the Action. (This is the property of the action, not the template).</td>
</tr>
<tr>
<td>Result</td>
<td>The result of the Action. (This is the property of the action, not the template).</td>
</tr>
</tbody>
</table>

7. Click **Add** to add an actor (Figure 128).
8. Enter the actor’s name in the first row of the **Name** column.

9. Click the first row of the **Role** column to add a role for the actor and select either **New Role** to create a new role or select an existing role. In this example, a new role will be created (Figure 129).
10. When **New Role** is selected, the **Edit Actor Template** dialog will open (Figure 130).

11. Type the Role, Description, and Category attributes (Figure 131). Click **OK** to add the role (clicking the **Cancel** button will not create the new role).
12. The actor will be added to the story step (Figure 132). Click **Next**.

**NOTE**
The *Create Actor* page will add the newly created actor template to the Actor Template collection list in the Storyboard libraries, so that you can reuse it later.
13. The **Create State Variable** page of the **Action Creation** wizard will open (Figure 133).
14. Click **Add** to add a State Variable.
Figure 134 -- Adding a New State Variable

15. Click the first row of the **New Value** column to add the State Variable value.

Figure 135 -- Selecting a New Value
16. Click the first row of the **Value Type** column to add a value type and select **New Type** to add a new value type or select an existing value type from the list. In this example a new value type will be selected.

17. The **Edit State Variable Template** dialog will open. Type the name, default value, and value type of the state variable (Figure 136).

![Figure 136 -- Entering the Details of the State Variable](image)

18. Click **OK** and the state variable will be added to the story step.
19. Click **Finish** to complete adding the action to this story step. You can always add another action to the story.
Figure 138 shows a scenario when the login process has been successfully completed. This scenario is shown as a story.

(iv) Creating Regions in Narrative Scene Using Image Editor

Cameo Requirements+ allows you to define details of your narrative scene by providing you the region capabilities to define an image. Every region on an image is dependent on the image, so
whenever the image is removed, all regions will be removed as well. You can have relationships among regions, use cases, and requirements.

To create a region in a narrative scene:

1. Open an image. Click to create a region (Figure 139).

2. Use your pointer to capture the region of the image (Figure 140).
3. Click the captured region to open the Properties view of the region. To cancel the selected region, click 🗑️. The Remove Region button will be enabled when you click the captured region.
(v) Playing Story Using Story Player

To see the flow of story as a presentation, you can play the story using the Story Player. The story player interface consists of two sections: (a) the name of the story step and (b) the description of the story step.

(a) The Name of the Story Step

The name of the story step will be displayed at the lower part of the story player.

Figure 141 -- The Region Attributes in the Properties View
(b) The Description of the Story Step

The description of the story step will be displayed under the name of the story step at the Description section of the story player.

*Figure 142 -- The Story Player Interface*

The Story Player toolbar consists of the following buttons (Figure 143):

*Figure 143 -- Story Player Toolbar*
### Table 47 -- Story Player Toolbar Buttons and Functions

<table>
<thead>
<tr>
<th>Buttons</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Previous Step" /></td>
<td>To return to the previous story step while playing the story.</td>
</tr>
<tr>
<td><img src="image" alt="Current Step" /></td>
<td>To refer to the step being played in the story.</td>
</tr>
<tr>
<td><img src="image" alt="Next Step" /></td>
<td>To move to the next story step while playing the story.</td>
</tr>
<tr>
<td><img src="image" alt="Play" /></td>
<td>To turn on the story player.</td>
</tr>
<tr>
<td><img src="image" alt="Pause" /></td>
<td>To turn off the story player.</td>
</tr>
<tr>
<td><img src="image" alt="Fast Rewind" /></td>
<td>To fast rewind the story.</td>
</tr>
<tr>
<td><img src="image" alt="Rewind" /></td>
<td>To rewind the story.</td>
</tr>
<tr>
<td><img src="image" alt="Forward" /></td>
<td>To forward the story.</td>
</tr>
<tr>
<td><img src="image" alt="Fast Forward" /></td>
<td>To fast forward the story.</td>
</tr>
<tr>
<td><img src="image" alt="Volume" /></td>
<td>To control the audio volume of the story.</td>
</tr>
</tbody>
</table>
CORE FEATURES
Cameo Requirements+ Core Features

To play a story:

1. Open the story node (that will be played) in the Data Collection view
2. Click the **Story Play** button (Figure 144).

3. The Story Player will open.

*Figure 144 -- The Story Play Button in the Story View Toolbar*
4. Click on the screen to continue. The first story step will appear (Figure 146).
5. Click the **Play** button to turn on the story player and play the story.

4.1.2.2 Discovering Business Process Using Stories

Cameo Requirements+ has the capability to gather business processes from all the stories you created. Cameo Requirements+ provides two flows to represent the suggested business process captured from the stories: (i) Scene Flow and (ii) Action Flow.

(i) Scene Flow

Cameo Requirements+ provides a scene flow for your story. Because every story step has reference to either narrative or decision scene, it gives Cameo Requirements+ information on how different stories are actually related.

A narrative scene represents an environment in which every story step takes place. The sequence of each narrative scene from each story gives enough information to Cameo Requirements+ to generate
a path from all story steps. Also when you create a story that involves a decision scene, it will tell Cameo Requirements+ that a decision is being made and how many outcomes are actually made by the actor.

With this information Cameo Requirements+ will provide you a scene flow to present a suggested business process to help you validate the business process from the captured stories.

To show a scene flow (you must have created at least one story):

1. Go to the Storyboard Creation perspective.
2. Select the story from the Data Collection view.
3. Click the **Scene Flow** button on the Story view toolbar to open the Scene Flow view (Figure 147).

4. The scene flow view will open.

![Figure 147 -- The Scene Flow Button](image)
Figure 148 -- Scene Flow View
(ii) Action Flow

As a system analyst you may want to define requirements with more technical details. Cameo Requirements+ allows you to create actions, actors, and variables to represent detailed information at each story step.

So these features also give Cameo Requirements+ enough information to represent you a flow based on all actions that you defined in every story. You can use an action flow to validate the actual business process or give a tester a suggested test path to ensure a thorough test case coverage.

To generate an action flow (you must have created at least one story and one action):

1. Open the Storyboard Creation perspective.
2. Select a story from the Data Collection view and click the Action Flow button (Figure 149).
3. The action flow will appear.

4.1.2.3 Managing Storyboard Resources with Storyboard Libraries Perspective

The Storyboard Libraries perspective is mainly used to work with reusable components of a storyboard. It allows you to manage all shareable resources and create a story from the shareable scenes.
4.1.2.3.1 Creating Scenes with the Storyboard Libraries View

You can create Actor, Action, and State Variable templates by using the same steps with the help of different libraries.

To create a scene using the Storyboard Libraries view:

1. Open the Storyboard Libraries perspective.
2. Select the Storyboard Workspace in the Data Collection view.
3. Select the Storyboard Libraries view (Figure 151).
4. Select **SceneCollection** from the **Libraries** box (Figure 152).
5. Click the **Add Scene** button on the toolbar and select either (i) **Add Narrative Scene** or (ii) **Add Decision Scene** (Figure 153). In this example, **Add Narrative Scene** will be selected. The Properties view will change to be a narrative scene properties view for you to enter information about the scene.

**NOTE**

You can use the same steps to create a decision scene. You can create a decision scene from the Storyboard Libraries view only.
4.1.2.3.2 Filtering the Libraries' Collections

To filter storyboard collections:

1. Open the **Storyboard Libraries** perspective.
2. Select the **Storyboard Libraries** view.
3. Select a collection type from the **Libraries** box.
4. Select a category from the **Categories** box (Figure 154).

5. All scenes belong to the selected category will appear in the **Story Libraries** view (Figure 155).
4.1.2.3.3 Working with Selection View

You can also create a story from the Storyboard Libraries perspective.

To create a story using the Storyboard Libraries perspective:

1. Open the Storyboard Libraries perspective.
2. Select the Storyboard Libraries view.
3. Select an order list of scenes to represent your story.

**NOTE**
The category filtering function will work only if you enter a category when creating a scene, actor, action, or state variable templates.
Figure 156 -- Selecting Sequence of Scenes for the Story in Storyboard Libraries View

4. The selected scenes will appear in order in the Selection view (Figure 157).

5. Either click (i) the Create New Story button from the Select view toolbar or (ii) the Add to Existing Story button to add the scenes to an existing story.
4.1.2.4 Storyboard Report Templates

Cameo Requirements+ provides two Storyboard report templates for you to generate Storyboard handouts and Storyboard document.

You can use these templates to generate Web or RTF reports (see the Working with Report Wizard section for more information on creating a report).

4.1.2.4.1 Generating Storyboard Handouts

You can generate storyboard reports in the form of handouts. Each slide in a handout contains a step, scene image, and actions body. There can be more than one slide on a page and they are ordered according to your story steps (Figure 158).
To generate Storyboard report handouts:

1. Select **Storyboard** from the data collection in the **Storyboard Creation** perspective or default perspective.

2. Click **Tool > Report Wizard**…. The **Report** wizard will open.

### NOTE

Use the same step to generate storyboard reports in a document template by selecting the **Storyboard Document** from the **Select a template** page.

3. Select **Storyboard Handout** from the **Select Template** pane and click **Next** (Figure 159). The **Select a Report profile**... (Figure 160) page will open.
Figure 159 -- Selecting Storyboard Handout
4. Select a report profile and click **Next**.
Figure 161 -- Saving the Storyboard Report

5. Click **Browse** to select a location to save the storyboard report. Click **Save** to confirm the location and filename.

6. Click **Next** to proceed to the next step and click **Finish** to generate the report.
Figure 162 -- The Generated Storyboard Report Handouts
4.1.3 Working with Query Builder

The Cameo Requirements+ query function allows you to query a large amount of data within a data file. The query function uses logical operators and constraints as a way to determine how it should search the data file. The Cameo Requirements+ query uses a logical tree to compare data within a data file and provide the results for the query. Tables 48 and 49 describe the logical operators and constraints of the Cameo Requirements+ query.

Table 48 -- Query Logical Operators

<table>
<thead>
<tr>
<th>Query Logical Operators</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>The logical AND operator</td>
</tr>
<tr>
<td>OR</td>
<td>The logical OR operator</td>
</tr>
<tr>
<td>NOT</td>
<td>The logical NOT operator</td>
</tr>
</tbody>
</table>
Cameo Requirements+ 4.1 enhances the search function by supporting both Boolean and enumeration data types search. A new Basic Mode Query Builder Wizard is also available for general users.

### 4.1.3.1 Creating Queries with Query Builder Basic Mode

Cameo Requirements+ provides a new basic mode as the default user interface of the Query Builder wizard. This new basic mode Query Builder hides most of the complexities of the query and allows you to create a new query in fewer steps.

You can create three constraint types in basic mode: (i) Type Constraint, (ii) Attribute Constraint, and (iii) Relation Constraint.

#### (i) Type Constraint

Type Constraint is used to find all nodes that belong to a node type or sub node type specified in the constraint. When you select a node type in the tree, all sub node types of the node will be selected as well. You can select more than one node type.

#### (ii) Attribute Constraint

Attribute Constraint is used to find all nodes specified in the attribute table that match with the attribute in result nodes. The Add button will be enabled only when all constraints in table is completed (all required fields are defined).
(iii) Relation Constraint

Relation Constraint is used to find nodes with a specific relationship type specified in the constraint, with any of the nodes selected by the query specified in the constraint. This constraint will be included in the query constraint results when the Query field has been defined.

You can select more than one constraint type to execute a single query. When you use all three constraint types for a single query in basic mode, the Query Builder wizard will use an OR operator to combine them.

You can create a simple to moderately complex query in the Query Builder basic mode. For more information about creating a more advanced query, see the Creating Query with Query Builder Wizard Advanced Mode section.

4.1.3.2 Creating Type Constraint Query in Query Builder Basic Mode

**Type Constraint** allows you to limit the search scope to a specific node type. For instance, selecting an ImageNode type as the Type Constraint will cause Cameo Requirements+ to search only for nodes that belong to the ImageNode type. You can use Type Constraint in conjunction with the other constraint types in creating a query. When you use all three constraint types in basic mode, Query Builder will place an OR operator between each constraint.

To create a new **Type Constraint** query in basic mode:

1. Select Query Collection.
2. Right-click and select Add Query Using Wizard from the shortcut menu (Figure 164) or select Data > Add Query Using Wizard from the main menu (Figure 165).
CORE FEATURES

Cameo Requirements+ Core Features

Figure 164 -- Opening Query Builder Wizard from the Shortcut Menu

Figure 165 -- Opening Query Builder Wizard from the Main Menu
3. The **Query Builder** wizard will open in basic mode (Figure 166). The first page in the wizard is **Query information**.

![Figure 166 -- The Query Builder Wizard in Basic Mode](image)

4. Enter the query name and click **Next**. The **Query Scopes** page will open (Figure 167). You can see the overall query summary in the **Summary** text box.
5. Click **Select all scopes** to include all nodes in **Query Scopes** or you can select only the container node to be the search scope. Click **Next**. The Query Constraints page will open (Figure 168).
6. The **Query Constraints** step will determine which nodes to be searched and showed in the result. Click the **Type Constraint** tab and select the node type that will be used as the search constraint.
Figure 169 -- Selecting a Node Type in the Type Constraint Tab
7. You can see the constraints summary in the Query Summary box while creating the constraints. Click Execute query and the Query Result table will show all the nodes searched by the wizard.
4.1.3.3 Creating Attribute Constraint Queries in Query Builder Basic Mode

To create a new Attribute Constraint query:

1. Select Query Collection.
2. Either right-click and select Add Query Using Wizard from the shortcut menu or click Data > Add Query Using Wizard from the main menu.
3. The Query Builder wizard will open in basic mode. The first page in the wizard is Query Information.
4. Enter the query name and click Next. The Query Scopes page will open. You can see the overall query summary in the Query summary box.

**NOTE**

- You can switch from basic to advanced mode to define more complex constraints by clicking the Advanced query button. All query information modified in the basic mode will be transferred to the advanced mode.
- You can also combine all three constraint types to compose your own query logic.
5. Either click **Select all scopes** to include all nodes in **Query Scopes** or select only the container node as the search scope. Click **Next**.

6. Select which nodes to be searched and showed in the result in the **Query Constraints** page. Click the **Attribute constraint** tab.

7. Click **Add** to add an attribute constraint (Figure 172). (You can select one out of five support attribute data type constraints: Text, Number, Decimal, Date, or Boolean).

![Figure 172 -- The Attribute Constraint Tab in Query Builder Wizard Basic Mode](image)

**Figure 172 -- The Attribute Constraint Tab in Query Builder Wizard Basic Mode**
4.1.3.3.1 Creating New Queries with Text Attribute Type

Text constraint allows you to search for nodes containing a specific string of characters (alphanumeric) or enumeration. You can also limit the search to a certain attribute type by selecting that particular attribute type from the Attribute type box.

To create a Text Attribute type constraint query:

1. Follow the steps in section 4.2.1.2 Creating an Attribute Constraint Query in the Query Builder Basic Mode.
2. Select Text from the Attribute type box (Figure 173).
3. Select the attribute and function from the Text constraint.
4. Enter the string that you are looking for in the Constraint value field.
5. Select true if the query constraint is case-sensitive or false if it is not.
6. Click Finish to create the query or click Execute query to execute the query and see the result in the Query result table.

![Figure 173 -- Text Attribute Type Constraint in Query Builder Wizard Basic Mode](image-url)
CORE FEATURES
Cameo Requirements+ Core Features

4.1.3.3.2 Creating New Queries with Boolean Attribute Type

Cameo Requirements+ allows you to search attributes with the Boolean data type.

To create a Boolean Attribute type constraint query:

1. Follow the steps in section 4.2.3 Creating Attribute Constraint Queries in Query Builder Basic Mode.
2. Select Boolean from the Attribute type box (Figure 174).
3. Select an attribute from the Attribute box. All attributes in a schema that contain the Boolean data type will be shown in the list.
4. Select either true or false as the constraint value.
5. Click Finish to create the query or click Execute query to execute the query and see the result in the Query result table.

### Table 50 -- Text Constraint

<table>
<thead>
<tr>
<th>Text constraint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>All node attributes in a schema where each attribute contains string and enumeration as the data type in the entire schema.</td>
</tr>
<tr>
<td>Function</td>
<td>Three function options: Contain, Start with, or Regular expression.</td>
</tr>
<tr>
<td>Constraint value</td>
<td>The word that you want to search.</td>
</tr>
<tr>
<td>Case sensitive</td>
<td>To determine whether or not it is a case-sensitive search. You can select either true or false.</td>
</tr>
</tbody>
</table>

**NOTE**
You can search for an enumeration data type with Text constraint.
4.1.3.3.3 Creating New Queries with Decimal Attribute Type

**Decimal constraint** allows you to search for nodes that contain an attribute with a number equal to, greater than, greater than or equal to, less than, or less than or equal to the number you specified. You can also limit the attribute to limit the search.

To create a Decimal Attribute type constraint query:

1. Follow the steps in section 4.2.3 Creating Attribute Constraint Queries in Query Builder Basic Mode.
2. In the Attribute type box, enter Decimal (Figure 175).

**Table 51 -- Boolean Constraint**

<table>
<thead>
<tr>
<th>Boolean Constraint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>All node attributes in a schema where each attribute contains the Boolean data type.</td>
</tr>
<tr>
<td>Constraint value</td>
<td>Two values: true and false.</td>
</tr>
</tbody>
</table>
3. Select an attribute from the **Attribute** box. All attributes in a schema that contain the Decimal data type will be shown in the list.

4. Select a function from the **Function** box.

5. Type a constraint value in the **Constraint Value** box.

6. Click **Finish** to create the query or click **Execute query** to execute the query and see the result in the **Query result** table.

![Query Constraints](image)

**Figure 175 -- Decimal Attribute Type in Query Builder Wizard Basic Mode**

**Table 52 -- Decimal Constraint**

<table>
<thead>
<tr>
<th>Decimal Constraint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>All node attributes in a schema where each attribute contains the Decimal data type.</td>
</tr>
<tr>
<td>Function</td>
<td>=, &gt;, &gt;=, &lt;, and &lt;=</td>
</tr>
<tr>
<td>Constraint value</td>
<td>The value that you want to search.</td>
</tr>
</tbody>
</table>

4.1.3.3.4 Creating New Queries with Number Attribute Type

**Number constraint** allows you to search for nodes that contain an attribute with a number equal to, greater than, greater than or equal to, less than, or less than or equal to the number you specified. You can also limit the attribute to limit the search.
To create a Number Attribute type constraint query:

1. Follow the steps in section 4.2.3 Creating Attribute Constraint Queries in Query Builder Basic Mode.
2. Select Number from the Attribute type box (Figure 176).
3. Select an attribute from the Attribute box. All attributes in a schema that contains Number as the data type will be shown in the list.
4. Select a Function constraint.
5. Type a constraint value in the Constraint Value box.
6. Click Finish to create the query or click Execute query to execute the query and see the result in the Query result table.

Figure 176 -- Number Attribute Type in Query Builder Wizard Basic Mode
4.1.3.4 Creating Relation Constraint Queries in Query Builder Basic Mode

Relation constraint is used to find nodes that contain a particular relationship type specified in the constraint, with any of the nodes selected by the query specified in the constraint. This constraint will be included in the query constraint results when the query field has already been defined.

To create a Relation constraint query:

1. Select Query Collection.
2. Either right-click and select Add Query Using Wizard from the shortcut menu or select Data > Add Query Using Wizard from the main menu.
3. The Query Builder wizard will open in basic mode. The first page in the wizard is Query Information.
4. Enter the query name and click Next. The Query Scopes page will open. You can see the overall query summary in the Query summary box.
5. Either click Select all scopes to include all nodes in Query Scopes or select only the container node to be the search scope. Click Next.
6. Select which nodes to be searched and showed in the result in the Query Constraints page. Click the Relation constraint tab.

Table 53 -- Number Constraint

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All node attributes in a schema where each attribute contains the Number data type.</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>=, &gt;, &gt;=, &lt;, and &lt;=</td>
</tr>
<tr>
<td>Constraint value</td>
<td>The value that you want to search.</td>
</tr>
</tbody>
</table>
4.1.3.5 Editing Existing Queries in Query Builder Basic Mode

To edit an existing query in basic mode:

1. Right-click a query node type and select **Edit Query Using Wizard** (Figure 178).

<table>
<thead>
<tr>
<th>Relation Constraint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relation</td>
<td>All available relationships in the schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The <strong>Related to</strong> or <strong>Is related</strong> type.</td>
</tr>
<tr>
<td>Query</td>
<td>You can select another query to join in the relation constraint. For example, you might have already defined a type constraint query to look for all the requirements, but you still can add this query as a part of the Relation constraint, so that you can look for the relationship that is either related to the requirement or is a related node of all requirements.</td>
</tr>
</tbody>
</table>
4.1.3.6 Creating Queries with Query Builder Advanced Mode

You can use the Query Builder wizard advanced mode to create all types of queries. The wizard can help you create high complexity query logic that cannot be done in basic mode.

Before you start using the advanced mode, it is necessary that you understand how the Query Builder wizard logical operator works.
4.1.3.6.1 Logical AND Operator

A logical AND operator requires two constraints. You can also create a nested logical AND operator in which two or more logical operators are used. When using the logical AND operator, Cameo Requirements+ will search for the nodes that satisfy (in this case) both constraints.

Figure 173 shows an example of a logical AND operator in set diagrams (Venn diagrams) that represent Constraints A and B. The intersection, shaded in black, where Constraints A and B met represents a condition where the nodes satisfy both constraints.

![Figure 179 -- The Logical AND Operator in Set Diagrams](image)

4.1.3.6.2 Logical OR Operator

A OR logical operator requires two constraints. You can also create a nested logical OR operator in which two or more logical operators are used. When using the OR logical operator, Cameo Requirements+ will search for the nodes that satisfy either constraint.

Figure 174 illustrates an example of a logical OR operator in set diagrams (Venn diagrams) that represent Constraints A and B. The union, in red, of Constraints A and B represents a condition where the nodes satisfy either Constraint A or B.
4.1.3.6.3 Logical NOT Operator

A **logical NOT operator** requires one constraint. When using the logical NOT operator, Cameo Requirements+ will search for the nodes that will not satisfy the constraint.

Figure 175 shows an example of a logical NOT operator in a set diagram (Venn diagram) that represents Constraint A. The region outside the diagram, shaded in red, represents all nodes that do not satisfy Constraint A.

To create a query in the Query Builder wizard advanced mode:

1. Right-click the **Query Collection** node.
2. Select **Add query using wizard** from the shortcut menu or **Data > Add Query Using Wizard** from the main menu.

3. Click the **Advance mode** button to enter the Query Builder Advanced mode.

4. Type: **All business Requirement** in the **Name** box.

5. Click **Next**. The **Query Scopes** page will open.
6. Click **Select all scopes** to search for all nodes or select only the nodes that you want by selecting the check boxes.

7. Click **Next**. The **Query Constraints** page will open.
8. Select **Constraints** from the Query Constraint tree.
9. Select a constraint or logical operator from the **Query Constraint** box.
10. Click **Add** to add a type constraint (you can also add another constraint or logical operator to the **Constraint** tree).
11. Add the type node for the constraint by selecting the Business Requirement node from the Constraint summary box.

12. Click Finish to complete the wizard.

### 4.1.3.7 Creating Queries with Logical Operator in Advanced Mode

You can build just about any kind of logic in your query using the Query Builder Advanced mode. Cameo Requirements+ supports logical operators in the Query Builder wizard to help you construct your query.

To add a logical operator in a query:

1. Right-click the Query collection node and select Add query Using Wizard from the shortcut menu.

2. Enter Get All Multimedia Nodes in the Query name box and click Next. The Query Scopes page will open.
3. Click **Select all scopes** to let search the multimedia node from the repository and click **Next**. The **Query Constraints** page will open (Figure 180).

![Figure 186 -- Selecting Constraints from the Constraint Tree](image)

4. Select **Constraints** from the **Constraint** tree.
5. Select a constraint or logical operator from the **Query Constraint** box.
6. Click **Add** to add the constraint or logical operator to the Constraint tree.
7. You can either add or change the variable description on the property pane located to the right of the Constraint tree.
8. Click Finish to complete the wizard.

4.1.3.8 Editing Existing Queries in Advanced Mode

To edit an existing query in the advanced mode:

1. Right-click a query node and select Edit Query using wizard.
2. To complete the whole wizard, follow the steps in section 4.2.2 Creating Queries with Query Builder Advanced Mode.
4.1.3.9 Executing Queries

To execute a query:

1. Right-click a query node and select **Execute Query**.
2. You can see the query results under the selected **Query** node in the tree or double-click the **Query** node to view the result in the editor pane.

4.1.4 Working with Dependency Matrix

Impact analysis functionality is essential to all requirement tools. Impact analysis helps you manage and notice the effect of each change made to the requirements. Cameo Requirements+ provides a way to do an impact analysis by creating a dependency matrix.

4.1.4.1 Creating Dependency Matrix

To create a dependency matrix:

1. Right-click the **Matrix Collection** node and select **Add Dependency Matrix Using Wizard** (Figure 189).

![Figure 189 -- Adding Dependency Matrix Using Wizard Shortcut Menu](image)
2. Type the name and description of the matrix in the **Name** and **Description** boxes (Figure 190).

3. Click **Next**.

*Figure 190 -- The Dependency Matrix Builder Wizard*
4. Either select one or more queries by selecting the query name check box from the tree or create a new query for the dependency matrix by selecting Select a new criteria > Create new criteria.

**NOTE**
The Row Criteria tree will show a list of previously created queries. When you click Create new criteria, the Query wizard will open. To create a query using the wizard, see Working with Query Builder section.
5. Click Next.

6. Select a query by selecting the query name check box, create a new, or choose the same queries selected for the Row Criteria by selecting the Same as row criteria check box.

7. Click Next.
8. On the **Allowed Node Types** page, either select the **Allow all types** check box to allow Cameo Requirements+ to detect all changes in the intermediate relationship between nodes, or select check boxes next to the node names to allow Cameo Requirements+ to detect changes in the selected nodes only.

9. Click **Next**.
10. On the **Allowed Link Names** page, either select the **Allow all links** check box to allow Cameo Requirements+ to detect changes in all of the links or select only some specific nodes.

To add a link to the **Allowed** list:
- Select an item from the **Disallowed** list and click **Add**.

To remove a link from the **Allowed** list:
- Select an item from the **Allowed** list and click **Remove**.
11. Click **Next**.

12. On the **Allow Relations** page, either select the **Allow all relations** check box or select specific relations from the **Disallowed** box to enable **Cameo Requirements+** to detect any changes in the relationship node, if any (through the attribute value change).

13. Click **Next**.
14. On the Relation attribute filter page, select specific attributes for Cameo Requirements+ to detect by selecting a node from the Relation attribute filter box and select the check box of each attribute in the table to the right to enable Cameo Requirements+ to detect changes in that attribute.

15. Click Next.
16. Adjust the UI Settings on the UI Settings page and click Finish.

4.1.4.2 Generating Dependency Matrix Report in CSV Format

To generate a CSV-format report:

1. Right-click a Dependency Matrix node.
4.1.4.3 Generating Dependency Matrix Report in HTML Format

To generate an HTML-format report:

1. Right-click a **DependencyMatrix** node.
2. Either select **Generate Dependency Matrix Report > Generate HTML Report** or click **Tools > Dependency Matrix > Generate HTML report**.

4.1.5 Working with Glossary

A glossary is used for storing specific names or technical words and their meanings. Cameo Requirements+ classifies a glossary as three node types: (i) Glossary Domain node, (ii) Glossary Term node, and (iii) Glossary Definition node.

(i) Glossary Domain Node

The Glossary Domain node is used for storing domain-related glossary definitions.

(ii) Glossary Term Node

The Glossary Term node is used for storing text to describe something. A glossary term may be used in more than one glossary domain. However, when a particular glossary term is used in a different glossary domain, it may give a different definition and that definition is valid only when the glossary term is used in that particular domain.

For example, when the glossary term “Book” is used in the glossary domain “Book Store”, it means “A product for sale” (glossary definition). However, when the term “Book” is used by the “Administration” domain then it means “Data of the book that is stored in the database” (glossary definition).

So a glossary definition depends on what glossary domain and glossary term are used. And for each domain, a glossary term should only have one meaning (only one glossary definition).

(iii) Glossary Definition Node

The Glossary Definition node is used for storing the meaning of glossary terms that depends on the glossary domain.
4.1.5.1 Creating Glossary Terms

The Cameo requirements+ default schema provides a set of fields for your glossary terms.

To create a glossary term:

1. Select the **Root** node from the tree.
2. Either right-click and select **Add Glossaries** or click **Edit > Add Glossaries**.
3. Either select the **Glossaries** node or right-click the **Glossaries** node > **Add Glossary Term**.
4. Enter the details of your glossary term in the editor pane.
CORE FEATURES
Cameo Requirements+ Core Features

Figure 199 -- Glossary Details in the Editor Pane

Table 55 -- Glossary Term Node Attributes

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Editor Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Single line text</td>
<td>The name of the glossary term.</td>
</tr>
<tr>
<td>Number</td>
<td>Single line text</td>
<td>The number of the glossary term.</td>
</tr>
<tr>
<td>Description</td>
<td>Multiline text</td>
<td>The default description/meaning of the glossary term.</td>
</tr>
<tr>
<td>Glossary definitions reference</td>
<td>Multiline text</td>
<td>A list of glossary definitions that have references.</td>
</tr>
</tbody>
</table>
4.1.5.2 Creating Glossary Domains

To create a glossary domain node type:

1. Select the Glossaries node from the Data Containment Tree.
2. Either right-click and select Add Glossary Domain or click Edit > Add Glossary Domain.
3. Enter the details of your glossary domain in the editor pane.

![Glossary Domain Node in the Editor Pane](image)

Table 56 -- Glossary Domain Node Attributes

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Editor Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Single line text</td>
<td>The name of the glossary term.</td>
</tr>
<tr>
<td>Number</td>
<td>Single line text</td>
<td>The number of the glossary term.</td>
</tr>
<tr>
<td>Description</td>
<td>Multi-line text</td>
<td>The default description/meaning of the glossary term.</td>
</tr>
</tbody>
</table>
4.1.5.3 Adding Definitions to Glossary Terms

If you enter the same text as the glossary term in the Cameo Requirements+ glossary, that particular text will be highlighted in yellow. This means that the text is a duplicate glossary term that has not yet been assigned a definition. You can assign a definition to the yellow-highlighted text.

To assign a definition to the yellow highlighted text:

1. Right-click the yellow-highlighted text.
2. Select Set Definition to… > (Glossary Term) > (Glossary Definition). The text will be highlighted in blue.
3. Bring your pointer over the blue-highlighted text to see the glossary domain details. The term definition will also appear in the glossary tool tip.

Figure 201 -- Assigning Definitions Using the Shortcut Menu

Figure 202 -- Glossary Tool Tip with Domain Definition
4.1.6 Working with Report Wizard

The Cameo Requirements+ Report wizard allows you to create a document or web-based document to share the requirements information with other stakeholders. Not only can the Report wizard help you to generate documents but it can also assist you to modify the report template and profile to fit your needs.

To generate a report using the Report Publisher wizard:

1. Select a node to be published.
2. Either right-click and select Report Wizard (Figure 203) or click Tools > Report Wizard... (Figure 204). The Report wizard will open.
Figure 203 -- Opening Report Wizard from Shortcut Menu
3. Select a **Report Template** from the **Select Template** tree on the **Select a template** page.

4. After selecting one of the templates (in this example, **Web Publisher 2.0**), click **Next**.
5. Select a **Report Profile** from the **Report Profile** table and enter the profile information, for example, the author and title. You can also click **New Value** to create new values for the report profile.

6. Click **Next**.
7. Either select the **Save to local machine** or select the **Save to FTP server** button to select a location to save the report.
### Table 57 -- Save Option Page Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Mandatory</th>
<th>Input Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save to Local Machine</td>
<td>A save option to save the report to a file in a local machine.</td>
<td>Yes</td>
<td>Boolean</td>
</tr>
<tr>
<td>Save to FTP Server</td>
<td>A save option to save the report to an FTP server.</td>
<td>Yes</td>
<td>Save to FTP</td>
</tr>
<tr>
<td>Report File</td>
<td>Text Input Field: used for directory input.</td>
<td>Yes</td>
<td>Text</td>
</tr>
<tr>
<td>Address</td>
<td>Text Input Field: to enter an FTP address.</td>
<td>Yes</td>
<td>Text</td>
</tr>
<tr>
<td>Login Name</td>
<td>Text Input Field: to enter an FTP account login name.</td>
<td>Yes</td>
<td>Text</td>
</tr>
<tr>
<td>Password</td>
<td>Text Input Field: to enter an FTP account password.</td>
<td>Yes</td>
<td>Text</td>
</tr>
<tr>
<td>Directory</td>
<td>Text Input Field: to enter an FTP directory.</td>
<td>Yes</td>
<td>Text</td>
</tr>
<tr>
<td>Implicit SSL</td>
<td>A security option to encrypt the Boolean report while sending it to the server.</td>
<td>No</td>
<td>Boolean</td>
</tr>
</tbody>
</table>
8. Click **Next** to proceed to the **Miscellaneous Options** page.
9. Click **Finish** to generate the report.
Figure 203 shows the generated Web Publisher report.

Cameo Requirements+ provides some default built-in templates that you can use to report the project summary and document views. These reports help you to exchange data between Cameo Requirements+ and other applications.

The following section will list available templates and their usage.

4.1.6.1 Built-In Templates
### Table 58 -- Default Templates

<table>
<thead>
<tr>
<th>Template Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Publisher 2.0 [HTML format]</td>
<td>This template allows you to publish Cameo models in HTML with rich features, so you can view them using a standard browser, such as Internet Explorer, Opera, or Firefox.</td>
</tr>
<tr>
<td>Document Report [RTF format]</td>
<td>List of all requirements' relationships and their properties.</td>
</tr>
</tbody>
</table>

### Table 59 -- Storyboard Templates

<table>
<thead>
<tr>
<th>Template Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storyboard Handout</td>
<td>A storyboard report in the form of handouts. Each handout contains six story steps.</td>
</tr>
<tr>
<td>Storyboard Document</td>
<td>A storyboard report in a formal document format.</td>
</tr>
</tbody>
</table>

### Table 60 -- Requirements Related Relationship Templates in HTML

<table>
<thead>
<tr>
<th>Template Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement list</td>
<td>A list of all requirements and their properties in a table format.</td>
</tr>
<tr>
<td>Requirement relationships</td>
<td>A list of all requirements' relationships and their properties.</td>
</tr>
<tr>
<td>Requirement Copy relationship</td>
<td>A list of all requirements' “Copy” relationship and their properties.</td>
</tr>
<tr>
<td>Requirement Derived relationship</td>
<td>A list of all requirements' &quot;Derived&quot; relationships and their properties.</td>
</tr>
<tr>
<td>Requirement Extend relationship</td>
<td>A list of all use cases' “Extend” relationships and their properties.</td>
</tr>
<tr>
<td>Requirement Include relationship</td>
<td>A list of all use cases' “Include” relationships and their properties.</td>
</tr>
<tr>
<td>Requirement Refine relationship</td>
<td>A list of all requirements' “Refine” relationships and their properties.</td>
</tr>
<tr>
<td>Requirement Satisfy relationship</td>
<td>A list of all requirements' “Satisfy” relationships and their properties.</td>
</tr>
<tr>
<td>Requirement TraceTo relationship</td>
<td>A list of all requirements' &quot;TraceTo&quot; relationships and their properties.</td>
</tr>
</tbody>
</table>
4.1.6.2 Report Templates

Cameo Requirements+ provides a very flexible template engine that allows you to create your own report template in an rtf, html, or open office format. The Report wizard template engine uses the velocity template engine so that you can customize report templates together with the Velocity script language.

4.1.6.3 Creating and Editing Report Templates

To create a report template:

1. Open the Report wizard. The Report wizard will open.
2. Click New Template. The New template dialog will open (Figure 210).
To edit a report template:

1. Open the Report wizard. The Report wizard will open.
2. Select a report template from the Select Template tree on the Select a template page and click Edit Template (Figure 211). The Edit Template dialog will open.

---

Figure 211 -- Editing the Selected Report Template
3. Edit the name and description of the template (Figure 212).

4. Click “…” to browse for a template file and select a category from the Category box or type a new category name to create a new one.

5. Click Change.

### 4.1.6.4 Creating and Editing Report Profile and Its Value

To create a report profile:

1. Click the **New Profile** button on the **Select a Report Profile...** page in the **Report** wizard (Figure 213).
CORE FEATURES
Cameo Requirements+ Core Features

Figure 213 -- Select a Report Profile Page

Figure 214 -- New Profile Dialog
2. Fill in the **Name** and **Description** boxes, and then click **Create**.

To create a new value to a report profile:

1. Select a profile from the **Report Profile** table on the **Select a Report Profile** page and click **New Value** to add a value to the profile. The **New Value** dialog will open.

![Figure 215 -- Adding New Value to the Profile](image)

2. Fill in the **Name** and **Value** boxes.
3. Click **Create** and the new value will be created (Figure 216).
To edit a report profile:

1. Select a profile on the Select a Report profile page in the Report wizard and click Edit Profile. The Edit Profile dialog will open.
2. Edit the Name and Description of the report profile.
3. Click Change.

To edit a report profile value:

1. On the Select a Report profile page, select a node value from the Value column in the Report Profile table and click Edit Value. The Edit Value dialog will open.
2. Edit the Name and Value boxes.
3. Click Change.
5 OTHER FEATURES

5.1 Cameo Requirements+ Other Features

This section will cover other features of Cameo Requirements+ that do not depend on schemas. You can use the features without considering the schema structure.

5.1.1 Working with Schema Builder

Before creating a schema, it is necessary to understand that there are base nodes that cannot be deleted in Cameo Requirements+ and are the only node types that Cameo Requirements+ works with. It is not possible to create any new node type without deriving it from one of these base nodes. Therefore, it is important to understand what each base node does. Table 60 lists the most frequently used base nodes in Cameo Requirements+.

Table 61 -- Cameo Requirements+ Frequently Used Base Nodes

<table>
<thead>
<tr>
<th>Base Node Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>A root node that contains all nodes in Cameo Requirements+.</td>
</tr>
<tr>
<td>Node</td>
<td>An abstract node that is the base node of every node with the exception of Root and Enumeration nodes. Every node in Cameo Requirements+ is derived from this node either directly or indirectly.</td>
</tr>
<tr>
<td>RelatableNode</td>
<td>A node that can contain a &quot;Relationship&quot; node.</td>
</tr>
<tr>
<td>Relationship</td>
<td>A node that presents the information about a relationship between RelatableNodes.</td>
</tr>
<tr>
<td>DirectRelationship</td>
<td>A node that represents a one-way relationship.</td>
</tr>
<tr>
<td>ImageNode</td>
<td>A node that is used to store images and their description.</td>
</tr>
<tr>
<td>DocumentNode</td>
<td>A node that is used to store a document file.</td>
</tr>
<tr>
<td>MultimediaNode</td>
<td>A node that is used to store a multimedia file.</td>
</tr>
</tbody>
</table>
OTHER FEATURES
Cameo Requirements+ Other Features

To open the Schema Builder wizard:

- Click **Tools > Schema Builder Wizard**...

![Figure 217 -- Opening Schema Builder Wizard](image)

To create a new schema:

1. Click **Tools > Schema Builder Wizard**.... The **Schema Builder** wizard will open.
2. Select the **Create a new schema** button and click **Next** (Figure 218).

![Figure 218 -- Creating New Schema with Schema Builder Wizard](image)
3. Enter the new schema **Signature name** and **Description** on the **Nodetype** page (a signature name cannot contain a blank space).

4. You can (i) click **Add** to create a new node type, (ii) select a node type from the **Nodetype list** table and click **Delete** to delete a node type, or (iii) select a node type from the **Nodetype list** table and click **Edit** to edit it. (See “Creating and Editing Node Types” section for more information on how to edit a node type.)

![Figure 219 -- Adding New Node Using Schema Builder Wizard](image)

To modify an existing schema:

1. Open the **Schema Builder** wizard. The **Schema Builder** wizard will open.
2. Select the **Edit a schema in the library** button. The **Schema Library** table will appear (Figure 220).
3. Select a schema from the table and click **Next**. The **Nodetype** page will open.

4. You can (i) click **Add** to add a new node type, (ii) select a node type from the **Nodetype list** table and click **Delete** to delete a node type, or (iii) select a node type from the **Nodetype list** table and click **Edit** to edit it.

**NOTE**

You will be prompted with a different wizard depending on the type of the node type you selected. See “Creating and Editing Node Types” section for more information on how to create and edit a node type.

### 5.1.1.1 Creating and Editing Node Types

You can create three kinds of node types in Cameo Requirements+: (i) a normal node type that stores most of the information, (ii) an enumeration node type that stores the enumeration data, and (iii) a relationship node type that stores the relationship data.
In the Schema Builder wizard, either (i) click the **Create a new schema** button on the Schema File page and click **Add** on the Nodetype page or (ii) click **Edit a schema in the library** button on the Schema File page and click **Next > Add**, to open the Nodetype Member wizard to select which node type you want to create (Figure 221).

| NOTE | No space is allowed when naming a schema, node, attribute, enumeration, and relationship in Cameo Requirements+. |

To create a normal node type:

1. Select **Create a normal or relatable nodetype** and click **Next**. The Nodetype Information page will open.
2. Type the nodetype name, for example, **Requirement**.
3. Select the base nodetype to derive from and click **Next**.
OTHER FEATURES
Cameo Requirements+ Other Features

4. You can (i) click **Add** to add an attribute, (ii) select an attribute and click **Delete** to delete it, or (iii) modify the name of the newly created attribute by double-clicking it in the **Attribute name** column. The selected attribute will be highlighted and you can then rename it.

5. You can modify an attribute type by clicking the arrow button next to the Attribute type and select: String, Boolean, Number, Date, Decimal, or User-defined enumeration.

6. Click **Next**. The Allowed Parents page will open (Figure 224).
7. Select a parent node check box from the **Nodetype Hierarchical** tree. Click **Next**. The **Allowed Children** page will open (Figure 225).

| NOTE | If the node you want to create has no parent node, you will not be able to use that node. |
8. Select any node type check box to create a child for the node type from the **Nodetype hierarchical** tree. Click **Next**. The **Nodetype Links** page will open (Figure 226).
9. Click **Add** to create a reference attribute for the node and click **Finish**.

To create an enumeration node type:

1. Open the **Nodetype Member Wizard** page (Figure 227).
2. Select the Create an enumeration nodetype button and click Next. The Enumeration Nodetype page of the Nodetype Member wizard will open (Figure 228).
OTHER FEATURES
Cameo Requirements+ Other Features

Figure 228 -- Assigning Literal Name and Value

3. Type the enumeration node type name in the **Name** box.
4. Click **Add** to add a literal enumeration.
5. Double-click a literal name in the **Literal name** column to modify the literal name.
6. Double-click a literal value in the **Literal value** column to modify the literal value.
7. Click **Finish**.

To create a relationship:

1. Open the **Nodetype Member Wizard** page.
2. Select the **Create a relationship nodetype** button and click **Next**. The **Relationship Information** page will open (Figure 229).
Figure 229 -- Assigning Relationship Name and Selecting Base Relationship Node

3. Type the relationship node type name in the Relationship name box.
4. Select the base node type for the relationship to be derived from and click Next. The Nodetype Attributes page will open (Figure 230).
Figure 230 -- Adding Attribute to the Relationship

5. Click **Add** to add an attribute to the relationship. Click **Next**. The **Allowed Sources** page will open (Figure 231).
6. Select the allowed sources of the nodetype for the relationship by selecting the check boxes. The allowed sources indicate the node types that can be used as the start point of the relationship. Click Next. The Allowed Targets page will open (Figure 232).
7. Select the allowed targets for the relationship by selecting the check boxes. The allowed targets indicate the node types that can be used as the end point of the relationship.

8. Click Finish.

5.1.1.2 Schema Display Name

The Cameo Requirements+ schema, node, attribute, enumeration, and relationship names contain no space; therefore, sometimes it causes the editor to generate hard-to-read labels.

You can use the Schema Display Name feature to make those names easier to read. If you activate the schema's node display name feature, Cameo Requirements+ will automatically insert a space before the uppercase letter in the node name. For example, a node named “testCase” in your schema will be displayed as “Test Case” if the Schema Display Name is on.
When you activate the Schema Display Name feature, Cameo Requirements+ will update name labels in:

- Data Containment Tree
- Context menu
- Editor pane’s property and table views
- Toolbar item name
- Editor pane’s heading
- Data menu
- Wizards and dialog boxes

To activate the schema display name feature:

1. On the Cameo Requirements+ main menu, either (i) click Edit > Preferences… (Windows) (Figure 233) or (ii) click Cameo Requirements+ > Preferences… (Mac OS X) (Figure 234). The Preferences window will open (Figure 235).
2. Click **Cameo Requirements+** to open the Cameo’s default options.

3. Select the **Use schema node’s display name** check box to activate the schema node display name feature.

4. Click **OK** to save the change and return to Cameo Requirements+.
5.1.2 Working with Schema Library Management

The Schema Library Management feature of Cameo Requirements+ allows you to view, import, export, or delete schemas.

5.1.2.1 Opening Schema Library Management

To open the Schema Library Management wizard:

- Click File > Schema Library Management... on the Cameo Requirements+ main menu (Figure 236).

![Opening Schema Library Management from the Main Menu](image)

5.1.2.2 Importing Schemas

To import a schema to Schema Library:

1. Open the Schema Library Management wizard.
2. On the Schema Library Management wizard page, click the Import tab and Import button (Figure 237). The Importing file dialog will open.
3. Select the Cameo Requirements+ archive file to import the schema from and click **Open**. The newly imported schema will appear in the Schema Library list.

### 5.1.2.3 Exporting Schemas

To export a schema to Schema Library:

1. Open the **Schema Library Management** wizard.
2. On the **Schema Library Management** wizard page, click the **Export** tab.
3. Select either (i) **From library** or (ii) **From data repository** and select a schema. (Figure 238).
4. Click **Export**. The **Exporting file** dialog will open.

5. Enter a filename and click **Save**.

### 5.1.3 Working with Data Repository Merge

When two or more people work on the same repository, they often have to merge data. **Data Repository Merge** allows you to merge your work with others’ into the original data repository. This feature also allows you to choose what changes to be included and merged into the original data repository.

For example, **User A** is working on a data repository named **Project1.cmzip**. While working on the project, **User A** also wants to let **User B** simultaneously work on the same project. In order to share...
his original data repository with User B. User A needs to first export Project1.cmzip as baseline so as to change the status of all data to baseline before the merger takes place.

User A can either export Project1.cmzip as (i) Baseline or (ii) not as Baseline (normal). If user A selects the Export as Baseline option, only the nodes with Baseline status will be exported. If user A does not select the Export as Baseline option, the status of the nodes will be the same before and after export.

To export a project:

1. Either (i) select a project node and click File > Export Data Repository... or (ii) right-click the project node and select Export... (Figure 239). The Export Data Repository wizard will open.
2. Browse the location to export the data repository to and select either one of the export options. Click \textbf{Finish}.

To export a data repository as baseline:

1. Select a project node and open the \textbf{Export Data Repository} wizard. The \textbf{Export Data Repository} wizard will open.
2. Click the **Browse** button. A dialog will open to allow you to select the location for the data repository you want to export.

![Figure 240 -- Selecting the Location to Export a Data Repository](image)

3. Select a location, for example **Project2.cmzip**, enter the filename, and click **Save**.

![Figure 241 -- Exporting Data Repository as Baseline](image)
4. Select the **Export as Baseline** check box.
5. Click **Finish** to export the data repository as baseline.

When the export of **Project2.cmzip** has been completed, you can start sharing it with other team members. The status of the changes made by other team members in **Project2.cmzip** will be **Modify**, and **Project2.cmzip** and its changes can be merged into the original data repository (**Project1.cmzip**) using the **Data Repository Merge** wizard.

| NOTE | The exported **Project2.cmzip** data repository will contain all data with the Baseline change status. |

To export a data repository normally:

1. Select a project node and open the **Export Data Repository** wizard.
2. Click the **Browse** button. A dialog will open to allow you to select the location for the data repository you want to export.
3. Select a location and save the data repository you want to export as, for example, **Project3.cmzip**.
4. Clear the **Export as Baseline** check box and click **Finish** to export the data repository (Figure 242).
To merge a data repository using the Data Repository Merge wizard:

1. Open Project1.cmzip and click File > Data Repository Merge.
2. Choose Project2.cmzip that has been updated with changes made by other team members and click Open. The Merge Data Repository wizard will open and display two sections: (i) **Conflicting data** section that contains conflicts between the current and merging nodes and (ii) **Detail** section (Figure 243).
3. Click **Finish** to merge data.
5.1.3.1 Conflicting Data

The Conflicting data section will display conflicts that occurred between the current and merged nodes. Although the merger will compare all nodes, only the nodes that have been changed, newly-added, or removed will be displayed in this Conflicting data section.

To review each change before accepting it:

1. Click a node in the Conflicting data table. The summary view of the nodes in the Project1.cmzip (original file) and Project2.cmzip (merged file) will be displayed in the Detail section.

2. The details of the original file (Project1.cmzip) will be displayed in the lower left pane.

3. The details of the merged or modified file (Project2.cmzip) will be displayed on the (i) upper and (ii) lower panes of the Detail section (Figure 245).
OTHER FEATURES
Cameo Requirements+ Other Features

Figure 245 -- Conflicting Data Details in Data Repository Merge Wizard

(i) Upper pane

- The first line on the upper pane of the Detail section displays the path or tree view structure of the node that has been selected.
- The second line is the Time Stamp that shows the time of the latest modification made on the selected node. If you want to select the latest modified data, choose the latest Time Stamp.
- The third line is the Change Status. The Change Status can either be MODIFIED or BASELINE.

Figure 246 -- Time Stamp and Change Status Details in Data Repository Merge Wizard

For example, user A wants to share the Project1.cmzip data repository with user B by exporting it as Project2.cmzip. Since the Export as Baseline option is selected, the status of all nodes in Project2.cmzip will be BASELINE. When user B edits the Project1.cmzip data repository, the status of all edited nodes will be changed to MODIFIED. But the status of the rest of the nodes (that are not edited) will still be BASELINE. Nodes with the status MODIFIED in Project2.cmzip will be considered as conflicting nodes and displayed in the Conflicting data section.
(ii) Lower pane

- The lower pane of the **Detail** section contains two summary views that display each attribute of the selected nodes and their values. The **Summary** view also displays the child nodes of the selected nodes. This will allow you to check if any child node has been added, deleted, or changed. If you accept a node whose child node has been changed, this child node will be automatically added to the data repository.

No changes from the data repository (Project2.cmzip) to be merged will be accepted by default (no data will be merged before you actually review and accept it). You can review the changes and accept them one by one or accept them all at once.

To review the changes and accept them one by one:

1. Click the first conflicting node in the **Conflicting data** section.
2. Compare and review the changes in the **Detail** section.
3. Either (i) select the **Accept all changes** button (to merge data in both Project2 and Project1) or (ii) clear the **Accept all changes** button (no data will be merged).
4. Repeat steps 1 to 3 until all the conflicting nodes are reviewed.

| NOTE | If you do not want to accept any change, DO NOTHING as Data Repository Merge does not accept any change by default. |

### 5.1.4 Working with Data Repository Migration Wizard

During requirements gathering, a domain expert may find that the schema they are working with is inaccurate or incomplete. To solve this problem he or she has to either delete or add nodes to the current schema so that it can accurately represent the business model. Modifying a schema in Cameo Requirements+ and deciding to use it instead of the old version sometimes can cause problems for data in the old version of the schema will be incompatible with that of the new one, thus making it unusable. Fortunately, Cameo Requirements+ provides a wizard that will help you migrate all data from the old schema to a new one.

To open the Data Repository Migration wizard:

1. On the Cameo Requirements main menu, click **Tools > Data Repository Migration**… (Figure 247). The **Data Repository Migration** wizard will open.
To migrate data from the old to a new schema version:

1. On the Cameo Requirements+ main menu, click **Tools > Data Repository Migration**. The **Data Repository Migration** wizard will open (Figure 248).

2. On the **Migration Information** page, click **Next**. The **Migrate a Data Repository** page will open (Figure 249).
3. Either enter the new schema filename to be migrated in the **Schema file for migration** box or click **Browse** to open a file dialog and browse for the schema file (Figure 250).

4. Click **Add**.

5. Select a `.cmzip` file that contains data you want to migrate and click **Open**. The `.cmzip` file will be added to the **Data repository files** table (Figure 251).
Figure 251 -- Adding Data Repository for Schema Migration

6. Click Migrate to complete the migration process.
To remove a .cmzip file from the table:

1. Select a file from the Data Repository files table.
2. Click Remove > Close.

**NOTE**

If the system fails to migrate the data repository, a dialog will open to inform you that some files cannot be migrated. The result will be shown in the Data Repository files table.

Figure 252 -- Failed to Migrate Data Repository
5.1.5 Working with Restore Backup File Wizard

The Restore Backup File wizard allows you to restore backup files easily. There are two types of backup file that you can restore: (i) the data repository backup file and (ii) the schema backup file.

A backup data repository file will be created every time you migrate a data repository from an old schema to a newer one. A schema backup file will be created whenever you modify an existing schema. This section will show you how to use each backup file.

5.1.5.1 Restoring Data Repository

To open the Restore Backup File wizard:

1. On the main menu, click File > Restore Backup File… (Figure 253). The Restore Backup File wizard will open (Figure 254).

![Figure 253 -- Restore Backup File Menu]
2. Click the **Data** tab to restore a backup data file.

3. Select either (i) **From backup folder** or (ii) **From file**.

   (i) To restore a data repository from the backup folder:

   1. Select the **From backup folder** button (Figure 255).
2. Select a file from the **Backup Data Files** table.

3. Click **Restore** to restore the backup data file.

(ii) To restore a data repository from file:

1. Select the **From file** button

2. Type the file location or click **Browse...** to search for the file (Figure 256).
3. Click **Restore** to restore the backup data file.

### 5.1.5.2 Restoring Schema

To open the **Restore Backup File** wizard:

1. On the main menu, click **File > Restore Backup File**... The **Restore Backup File** wizard will open.
2. Click the **Schema** tab.
3. Select either (i) **From backup folder** or (ii) **From file**.
(i) To restore a schema from the backup folder:

1. Select the **From backup folder** button (Figure 257).

![Figure 257 -- Restoring Schema from Backup Folder](image)

2. Select a schema from the **Backup Schema Files** table.
3. Click **Restore** to restore the backup schema file.

(ii) To restore a schema from file:

1. Click the **From file** button (Figure 258).
OTHER FEATURES
Cameo Requirements+ Other Features

Figure 258 -- Restoring Schema from File

2. Type the file location or click Browse… to search for the schema file.
3. Click Restore to restore the backup schema file (a backup schema file has a file extension name: .schemabackup).

5.1.6 Working with Spelling Checker

Cameo Requirements+ provides the Spelling Checker feature to help you check and correct spelling mistakes in all editor panes and property views. Cameo Requirements+ underlines misspelled words in red (Figure 259).
5.1.6.1 Checking Spelling Using Spelling Checker

To check spelling using the spelling checker:

1. Right-click the misspelled word and select **Spell Check**. You will see the suggested word on the top of the menu. (Figure 260).
2. You can select the suggested word, **Ignore all**, or **Add to dictionary** to add the word you typed to the dictionary.

### 5.1.6.2 Changing Spelling Checker Preferences

You can change the spelling checker preferences such as the highlight color, ignore one letter word, or ignore mixed case rule, and so on.

To open the Spelling Checker preferences:

1. Click **Edit > Preferences**.... The **Preferences** dialog will open (Figure 261).
2. Select your preferences by selecting the check box or entering the information in the box.
3. Click **Apply > OK**.
OTHER FEATURES
Cameo Requirements+ Other Features

Figure 261 -- Changing Spelling Checker Preferences
### Table 62 -- Spelling Checker Preferences Settings

<table>
<thead>
<tr>
<th>Settings</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spelling Dictionary File</td>
<td>To change the tool’s main dictionary.</td>
</tr>
<tr>
<td>User Dictionary File Suffix</td>
<td>To automatically show each word added in the custom dictionary.</td>
</tr>
<tr>
<td>Ignore Numbers</td>
<td>To ignore words that contain numbers. If this option is selected, the spelling checker does will not detect, for example, <em>Project2.cmzip</em> as a mistake.</td>
</tr>
<tr>
<td>Ignore one letter word</td>
<td>To ignore one-letter words, for example, <em>a</em> and <em>I</em>.</td>
</tr>
<tr>
<td>Ignore Mixed Case</td>
<td>To ignore a word that contains mixed uppercase and lowercase letters, for example, <em>UseCase</em>.</td>
</tr>
<tr>
<td>Ignore Sentence Capitalization</td>
<td>To ignore the capitalization of the first letter of a sentence.</td>
</tr>
<tr>
<td>Ignore Uppercase</td>
<td>To ignore words in which all letters are uppercase, for example <em>CSV</em>.</td>
</tr>
<tr>
<td>Ignore Compounds containing these Characters</td>
<td>To ignore compound sentences containing some specific characters.</td>
</tr>
<tr>
<td>Markup color for spelling errors</td>
<td>To select a color to underline misspelled words.</td>
</tr>
<tr>
<td>Max. number of errors per document</td>
<td>To count the number of misspelled words in each document.</td>
</tr>
</tbody>
</table>
6 CLIENT-SERVER FEATURES

6.1 Cameo Requirements+ Features on Cameo Team Server

Cameo Team Server allows you to work collaboratively with metamodels and models in Cameo Suite. Metamodels and models are stored in a shared repository, which can be used simultaneously by model designers.

Each Cameo Team Server client (for example Cameo Requirements+) has a built-in Cameo Team Server support. To be able to connect to Cameo Team Server, you need to install it from a separate installation file.

6.1.1 Logging into Cameo Team Server

To gain access to the Cameo Team Server resources and also create new resources on Cameo Team Server you need to first log in to Cameo Team Server.

To log in to Cameo Team Server:

1. Either (i) click Server > Login on the main menu or (ii) press the keyboard shortcut Ctrl+Shift+L. The Login dialog will open.

| NOTE | You need to start Cameo Team Server first before logging in. |
2. Type your **Username** and **Password**, and select the **Server** name (the Cameo Team Server name).

3. Click **OK**. The **Login** dialog will close and you will be logged in.
6.1.2 Schema Builder Wizard: The Server’s Schema

6.1.2.1 Creating New Schemas

You can create new remote schemas that will be stored on the remote server by using the Schema Builder wizard.

To create a new schema on the server:


![Schema Builder Wizard Menu](image)

*Figure 264 -- Schema Builder Wizard Menu*
2. Select **Remote** and click **Next**. The **Schema Selection** page will open.
3. Select the Create a new schema button and click Next. The newly created schema will be stored on the server.

4. Click Finish.

NOTE
See Working with Schema Builder section, Chapter 5, Cameo Requirements+ User Guide, to define the entities and attributes of your new schema (this operation works the same regardless of the location where the schema is stored, either local or remote).

6.1.2.2 Editing Existing Schema
You can also edit an existing schema stored on the remote server using the Schema Builder wizard.
To edit an existing schema on the server:

1. Click **Tools > Schema Builder Wizard**. The **Schema Builder** wizard will open.

![Figure 267 -- Schema Builder Wizard Menu](image)

2. Select **Remote** and click **Next**. The **Schema Selection** page will open.

![Figure 268 -- Schema Builder Wizard](image)
3. Select the *Edit a schema in the library* button. All of the schemas stored on the remote server in the *Schema library* table will open. Select a schema and click *Next.*

4. Modify the entities and attributes of the selected schema and click *Finish.* The edited schema will be stored on the server and, if it has been modified, it will be assigned a new version number.

**NOTE**

See the *Working with Schema Builder* section, Chapter 5 above to modify the entities and attributes of a schema (this operation works the same regardless of where the schema is stored, either local or remote).

### 6.1.2.3 Migrating Data Repository to Server’s Schema

When you create or edit an existing schema on the server, you can migrate an existing remote repository stored on that server to use that new schema. In this way, data that is stored in the remote repository that uses the old schema will not be lost, but will be copied to the new schema, mapping the old entities and attributes to the new ones.
To migrate a remote repository to a new remote schema on the server:

1. Click **Tools > Migrate Data Repository** on the menu. The **Migrate Data Repository** wizard will open.

2. Select **Remote** and click **Next**. The **Migrate a Data Repository** page will open.
Figure 272 -- Migrate a Data Repository Page

3. Click Select to select a remote schema that will be used in the repository and be migrated. The Select Schema dialog will open.
4. A list of schemas stored on the server will open. Select a schema and click **OK**.
5. Click **Next** to select a data repository to be migrated to the new schema.
6. Click **Add** if the **Data Repositories** table does not show the data repository you want to migrate. The **Select repository** page will open.
7. Select a remote repository to be migrated to the selected schema from the Repository library table and click OK. The selected remote repository will open on the Migrate a Data Repository page.
8. You can add as many repositories as you want by repeating steps 6 to 7.
9. Select a repository to be migrated from the Data repositories table and click Migrate. The system will start migrating the selected remote repository.
6.1.3 Schema Library Management for Remote Schema

6.1.3.1 Importing Schema from Local Library

To import a schema from a local library to Cameo Team Server Schema Library:

2. Select **Remote** and click **Next**.

3. If you are not currently logged in, the **Login** dialog will open prompting you to type the username, password, and select a server. If you are logged in, the **Manage a Server Schema Library** page will open.
4. Click the **Import** button. The **Importing file** dialog will open.

---

**Figure 281 -- Login Dialog**

**Figure 282 -- Manage a Server Schema Library Page**
5. Select a schema from the local Schema Library (on Windows: C:\Program Files\Cameo-Requirements+4.1\Schemas).

6. Click Open. The schema will be added to Cameo Team Server Schema Library.

### 6.1.3.2 Adding Schema

Alternatively, you can import a local Schema to Cameo Team Server Schema Library by clicking **Server > Add Schema to Server** on the main menu.

To add a schema to Cameo Team Server from the server menu:

1. Click **Server > Add Schema to Server** on the menu. The **Add Local Schema to Cameo Team Server** dialog will open.
2. The **Schema library** section will display all the local schemas available from the local Schema Library folder. Select a **Schema** to be uploaded to **Cameo Team Server Schema Library**. If you want to save the schema in the server with a different name, change the schema name in the **Server schema name** box.

3. Click **OK**. The schema will be added to Cameo Team Server Schema Library.
6.1.3.3 Exporting Schema from Cameo Team Server Schema Library

To export a schema from Cameo Team Server Schema Library to a local machine:

1. Repeat steps 1 to 4 of section 6.1.3.1 Importing Schema from Local Library above.

2. Click the **Export** button. The **Exporting file** dialog will open.
3. Select a path to save the schema:
   - suggested path for Windows: C:\Program Files\Cameo Requirements+ 4.1\Schemas
   - for other OS, go to the Installation folder and look for the Schema Folder

4. Click **Save**. The schema will be exported from Schema Library in Cameo Team Server to the selected path in the local machine.

### 6.1.3.4 Deleting Schema

To delete a schema from Cameo Team Server Schema Library:

1. Click **File > Schema Library Management**. The **Schema Library Management** wizard will open.
2. Select **Remote** and click **Next**. If you are not currently logged in, the **Login** dialog will open. If you are logged in, the **Manage Server Schema Library** page will open.
3. Select a schema to be deleted from the Schema library table and click **Delete**. A dialog will open prompting you to delete the schema.
CLIENT-SERVER FEATURES
Cameo Requirements+ Features on Cameo Team Server

4. Click **OK** and the selected schema will be deleted.

6.1.4 Data Repository on Cameo Team Server

6.1.4.1 Creating New Data Repository

If you have a valid license key for Cameo Team Server, then you have two ways to create a data repository, either (i) create a data repository on your local machine or (ii) create a data repository directly on Cameo Team Server.

(i) Create Data Repository on Your Local Machine

To create a data repository on your local machine, see **Creating New Data Repository**, Chapter 3, Cameo Requirements+ User Guide.

(ii) Create Data Repository on Cameo Team Server

To create a new data repository on Cameo Team Server:

1. Click **File > New**, press **Ctrl + N**, or click 📋 on the main toolbar.
2. The **Repository Type selection** page will open.

3. Select **Remote** and click **Next**. The **Create a new data repository** page will open.
4. Type the data repository name, select a schema from the **Schema Library** table, and type in some comment for the first version of repository. Then click **Finish**, the new data repository will open in Cameo Requirements+.
6.1.4.2 Opening Existing Data Repository

To open an existing data repository from Cameo Team Server:

1. Click **File > Open**, click on the main toolbar, or press **Ctrl + O**.

**NOTE**
A new data repository will have a root node (the base node of a schema) and all report templates.
2. The **Repository Type Selection** page will open.
3. Select **Remote** and click **Next**. The Open Repository from Cameo Team Server page will open and show all data repositories that are available for you to open from Cameo Team Server.
4. Select a repository to open from the **Repository library** table and click **Finish**.

5. The selected repository will open in Cameo Requirements+.
To open a server repository from the server menu:

1. Either click (i) **Server > Open Server Repository** or (ii) press **CTRL+Shift+O**. The **Open Repository** dialog will open.
2. Select a repository from the Repository library table and click Open. The selected repository will open in Cameo Requirements+

6.1.4.3 Closing Data Repository

To close a data repository on Cameo Team Server:

1. Click File > Close Data Repository on the menu, click on the toolbar menu, or press Ctrl + W.
2. The **Commit/Discard Changes** dialog will open if there are some changes that have not yet been committed.

3. Either (i) type some comment for new update to repository and click **Commit** or (ii) click **Discard changes**.

### 6.1.4.4 Adding Local Repository to Cameo Team Server

To add a local data repository to the Cameo Team Server Repository library:

1. Open a local Data Repository.
2. On the main menu, click **Server > Add Repository to Server**. The **Add Local <Resource Type> to Server** dialog will open.
3. Type the name of the repository (if you would like to change the existing local repository name), and then click **Next**.
Figure 311 -- Add Local Schema to the Server Option

4. Select either (i) **Add local schema to the server** or (ii) **Use existing schema from server**.

   (i) Select the **Add local schema to the server** button if the **Schema Library** table does not list the schema of the selected data repository, and type the name of the local schema that will be added together with the repository in the **Name** box.

Figure 312 -- Naming Local Schema
(ii) Select the **Use existing Schema from server** button if the **Schema library** table lists the schema of the selected data repository, and select the schema (Figure 312).

5. Click **Finish**. The selected schema with its data repository will be added to the **Cameo Team Server Repository** library and **Schema Library**.

### 6.1.4.5 Converting Cameo Team Server Repository to Local Repository

To convert a server repository to a local machine:

1. Open a server data repository.
2. Click **Server > Convert to Local Repository**.

3. (i) If there are no uncommitted changes in the currently opened data repository, the **Save as repository as** dialog will open. Select a path to save the repository and click **Save**.
(ii) if the selected repository has some uncommitted changes, a warning will open. Click either (i) **Yes**, if you do not want to commit changes, or (ii) **No**, if you would like to commit changes before converting the repository.

---

**NOTE**

When a server data repository is converted to local and uploaded to the server, it will be assigned a new version number. Cameo Requirements+ 4.0 does not support offline mode.

---

### 6.1.4.6 Managing Remote Repositories

The Repository Management features allow you to add, remove, rename, or browse remote repositories. These features are accessible from the **Repository Management** dialog.

To add a new, empty remote repository:

1. Click **Server > Repository Management** on the main menu. The **Repository Management** dialog will open.
2. Click Add. The Add Server Repository dialog will open and display a list of the remote schemas available for the repository that you want to create.
Figure 318 -- Adding Remote Repository

3. Select a remote schema from the **Schema library** table and type the name of the repository that you want to create in the **Name** box.

4. You can also type some comment for the new repository.

5. Click **OK** to add the newly created repository to the server.

To remove a remote repository:

1. Click **Server > Repository Management**. The **Repository Management** dialog will open and display a list of the remote repositories available in the **Repository library** table.
2. Select a repository and click **Remove**. A dialog will open prompting you to confirm the repository removal.

3. Either click (i) **Yes** to remove the repository or (ii) **No** to cancel the operation.

To rename a remote repository:

1. Click **Server > Repository Management**. The **Repository Management** dialog will open and display a list of the remote repositories available in the **Repository library** table.
2. Select a repository you want to rename and click **Rename**. The **Rename Repository** dialog will open.

3. Change the repository name and type some comment for the new version of repository.

4. Click **OK**. The repository will be saved under the new name and no data will be lost. The repository will also have a new version number.
NOTE

Any changes you make to a repository will be recorded in a form of a new version number assigned to the repository and kept in the repository history.

Figure 323 -- Repository's New Name and Version Number

To browse the history of a remote repository:

1. Click **Server > Repository Management**. The **Repository Management** dialog will open and display a list of the remote repositories available in the **Repository library** table.
2. Select a repository and click **History**. The **History** dialog will open and show all the changes made to the selected repository.
3. Select the version that you would like to open and click Open. The selected version will open in read-only mode.

4. Click Close to close the History dialog.

To change a repository comment:

1. Click Server > Repository Management. The Repository Management dialog will open and display a list of the remote repositories available in the Repository library table.
2. Select a repository and click **History**. The **History** dialog will open and show all of the changes that have been made to the repository.
3. Select a version whose comment you would like to change and click **Version Properties**....

The **Version Properties** dialog will open.
Figure 328 -- Version Properties Dialog

4. Type your comment and click **OK**. The comment of the selected repository version will be updated, but the version number will remain the same.

To revert a repository to an earlier version:

1. Click **Server > Repository Management**. The **Repository Management** dialog will open and display a list of the remote repositories available in the **Repository library** table.
2. Select a repository and click **History**. The **History** dialog will open and show all of the repository versions.
3. Select an earlier version and click **Set as Latest**. The **Set as Latest** dialog will open.

4. Type your comment for the version you are going to revert to and click **Revert**. The selected earlier repository version will overwrite the current one.
5. Once the repository has been reverted to the earlier version, close the History dialog. The version number of the repository will be updated.

| NOTE | The history of a reverted repository will be displayed in New Version/Reverted Version order. For example, if the current version number of a repository is 6 and you revert to version 4, the new version number that will be displayed in the History dialog is 7/4. |

6.1.4.7 Locking Elements

The Cameo Team Server repositories can be shared and edited by more than one user at a time. Cameo Team Server has the ability to lock elements to prevent more than one user from editing the same node, element, or data repository at a time. You need to have a lock to edit an element or the whole repository.

To lock an element in the tree:

- Right-click an element you want to edit and select Lock > Lock. The element will be locked.
6.1.4.8 Editing Locked Elements

You can edit a node only after you have locked it. This node locking feature allows you to edit a node while preventing others from doing it at the same time. The locked node will be visible in all views and wizards.

To edit a node and its child nodes:

1. Right-click a parent node and click Lock > Lock Recursively.
2. Once you have selected **Lock Recursively** for a selected node, you can (i) edit or delete the selected node, (ii) edit or delete the child nodes of the selected node, or (iii) add new child nodes to the selected node.

### NOTE

- To edit or delete a locked node’s reference, you need to lock the reference node first.
- To edit or delete a locked node’s relationship, you need to lock the target relationship node first.
- You cannot drag a locked node to an unlocked node. To do so, you need to lock the target node first.
- When you lock a query, all of its child nodes will be locked as well. You cannot lock the child nodes of a query individually.
- You can copy an unlocked node and paste it into a locked node. When you copy an unlocked node, the reference node will also be copied.

### 6.1.4.9 Resource Lock

When you migrate a data repository, your username will appear to other users as the one who locks the data repository, and the data repository will be automatically locked in the Resource Lock mode. No other users can lock the data repository or any elements of the repository when it is in the Resource Lock mode.

You cannot migrate, import, or export any data repository whose element(s) is locked by other users at the repository level.

### 6.1.4.10 Viewing Locked Elements in Data Repository

To view elements that are locked in a data repository:

1. Either (i) click **Server > View Locks** or (ii) press **Ctrl+Shift+V**. The **Locks** tab will open next to the **MyRepository** tab in Cameo Requirements+, showing all elements that are locked by all users.
CLIENT-SERVER FEATURES
Cameo Requirements+ Features on Cameo Team Server

Figure 335 -- View Locks Menu

Figure 336 -- Locks Tab
6.1.4.11 Unlocking Elements

If you log out of the server without unlocking the elements that you have locked, they will remain locked until the next time you log in to the server and unlock them.

To unlock all elements:

- Click Server > Unlock All (Figure 337) to unlock all elements in the open data repository.

![Figure 337 -- Unlocking All Elements](image)

To unlock an element individually:

- Right-click a locked node and select Lock > Unlock (Figure 338) to unlock it in the open data repository.
To unlock a node and all of its child nodes:

- Right-click a node and select **Lock > Unlock Recursively** (Figure 339) to unlock the selected element and all its sub-elements.
6.1.4.12 Updating, Committing, and Discarding Changes

You can use the **Update** menu to update any changes you made to a repository on Cameo Team Server. When the **Update** menu is selected, the system will update all changes made by all users to the currently open data repository.

You can use the **Commit** menu to commit all of the changes you made to the currently open data repository.

You can use the **Discard Changes** menu to discard all of the changes that have not been committed.

To update any changes you made to a repository:

- On the main menu, click **Server > Update** (Figure 340).
To commit changes to Cameo Team Server:

- Click **Server > Commit** on the Cameo Requirements+ main menu (Figure 341).
To discard any changes you make to a repository:

- On the main menu, click **Server > Discard Changes** (Figure 342).

### 6.1.5 Viewing Element History

You can view the history of each element in the Properties Editor. The history is represented in the tabular format. Each line in the history table represents each change made to the element. You can also open an earlier version or the version properties of an element through the Properties Editor.

To open the Element History tab of an element:

1. Double-click the element in the Containment tree. The Properties Editor will open.
2. Click the **History** tab (Figure 342).

---

Figure 342 -- Discard Changes Menu

Figure 343 -- Element History Tab
To open an earlier version of an element in the Properties Editor:

1. Double-click an element in the Containment tree. The Properties Editor will open.
2. Click the **History** tab.
3. Right-click the version that you want to open in the Properties Editor and select **Open** (Figure 344). The Properties Editor of the selected version will open in read-only mode.

![Figure 344 -- Opening an Earlier Element Version](image1)

To open the **Version Properties** dialog:

1. Double-click an element in the Containment tree. The Properties Editor will open.
2. Click the **History** tab.
3. Right-click a version whose Version Properties you want to see and select **Version Properties** (Figure 345). The **Version Properties** dialog will open (Figure 346).

![Figure 345 -- Selecting Version Properties](image2)
6.1.6 Logging out of Cameo Team Server

You need to log out to end your access to Cameo Team Server.

To log out of Cameo Team Server:

1. Click **Server > Logout**. If you have made some changes, but have not yet committed them, the **Commit/Discard Changes** dialog will open.
2. You can click (i) **Commit** to commit the unsaved changes as well with the comment, (ii) **Discard changes** to discard the unsaved changes, or (iii) **Cancel** to remain logged in.
7 CREATING AND IMPORTING REPORT TEMPLATES

7.1 Report Template Overview

You can create a report template using Report Wizard and use it in Cameo Requirements+. To make the most of Report Wizard, you need to understand Apache Velocity and Cameo Requirements+ elements. Report Wizard uses Apache Velocity, this means that you will need to know how to create a Velocity template and that every rule that applies to Velocity will also apply to Cameo Requirements+ Report Wizard.

A template, in Velocity, is a text file that tells Velocity how the output should look like. It contains a document page style, layout, header, footer, and static text just like any other templates that come with most of the Word Processor programs. However, a Velocity template also contains specific placeholders for Java objects and Velocity Template Language scripts, which will tell the Velocity Engine what and how to print the information in the output.

Velocity is a Java-based template engine that processes templates and references Java objects to produce output documents. A basic Velocity template can contain static text, layouts, conditional statements, and placeholders for each referenced Java object. When a template is being read by Velocity, conditional statements will be processed and placeholders will be replaced with the value from the referenced Java objects. You can use Velocity, for example, to generate web pages, emails, and read XML documents, but with Report Wizard it is now possible to use Velocity to read other types of templates, such as Rich Text Format documents. The following sections will describe some basic aspects of Velocity, Cameo Requirements+ elements, and Report Wizard directives.

7.2 Working with Templates

To create a report template, you need to open the type and attributes of the Cameo Requirements+ element inside the template and get their values. Once you have created the template, you need to import it in order to use it in Cameo Requirements+. 
7.2.1 Creating a Template

To create a template:

1. Select an element in the Cameo Requirements+ and open the element Specification dialog to see the type and attributes of the element, and get their value (Figure 349).

*Figure 349 -- Use Case Element Specification*
2. Open Microsoft Word and type, for example:

```
#foreach ($usecase in $UseCase)
    Name: $usecase.name
    Priority: $usecase.priority
    Verify Method: $usecase.verifyMethod
#end
```

The above example shows that $UseCase is an array that contains all of the Use Case elements in Selected Element Scope and $usecase is an individual Use Case element inside the array. To access the value of a property in $usecase, type: $usecase followed by: ".", and the attribute’s name (”.name”, “.priority”, and “.verifyMethod” are the names of the $usecase attribute). The syntax to access the value of the attribute can be represented by: $[Referenced object].[Attribute’s name].

When you generate the output of the template for a project, the result will show the values of all Use Cases:

```
Name: FillOrder
Priority: Highest
Visibility: Analysis
```

### 7.2.2 Importing a Template to Cameo Requirements+

You can import a template that you have created to Cameo Requirements+ by using Report Wizard.

To import a template to Cameo Requirements+ using Report Wizard:

1. Click **Tools > Report Wizard...** on the Cameo Requirements+ main menu. The Report Wizard window will open (Figure 350).
2. Click the **New Template** button. The **New Template** dialog will open (Figure 351).
3. Type the name and description of the template, and select a category (Figure 351).
4. Click the “…” button to locate the template file that you have created and click **Select**.
5. Click **Create** to import the template.

To generate a report based on a template, select the template and follow the instructions in the **Report Wizard** window.

### 7.3 Velocity Template Language

Velocity Template Language is a scripting language used only by Velocity Engine to determine how the output should look like. This section is divided into two parts: (7.3.1) Velocity Variable and (7.3.2) Velocity Directive. For more information on the Velocity Template Language, visit [http://velocity.apache.org/engine/releases/velocity-1.6.2/user-guide.html](http://velocity.apache.org/engine/releases/velocity-1.6.2/user-guide.html).
7.3.1 Velocity Variable

A Velocity variable can either be a referenced Java object or a declared variable inside a template. A Velocity variable begins with $ and followed by the name of the variable. Depending on what is added to the Velocity Context (Cameo Requirements+ automatically adds its element to the Velocity Context) the variable can either be a local variable or a reference to a Java object.

To declare a local variable inside a template, type: $ followed by a string beginning with an alphabet. A referenced Java object variable is provided by Report Wizard.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Return Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Requirement</td>
<td>Contains a list of requirement elements.</td>
</tr>
<tr>
<td>$UseCase</td>
<td>Contains a list of use case elements.</td>
</tr>
<tr>
<td>$Section</td>
<td>Contains a list of section elements.</td>
</tr>
</tbody>
</table>

If a Velocity variable is a reference to a Java object, you can call its methods and value either by:

(i) Calling the object method with the Velocity variable directly, for example, $object.method(). This is useful if you want to use a Java object's method to process data and print it out.

(ii) Calling the object properties, for example, $object.name.

Velocity allows you to call a referenced Java object's method directly like you would in Java. This is very useful as you can create methods in Java that can help you create a more flexible Velocity template. For instance, if you need to perform a complex operation, such as sorting a list according to certain variables, you can create a sorting method in Java, and then call that method from a Java object.

7.3.2 Velocity Directives

A Velocity directive is a keyword used by Velocity to perform certain tasks such as looping and controlling the output of a document based on the values from Cameo Requirements+. Directives can help you make a dynamic template and you can do more than just printing out Cameo Requirements+ elements and attributes. A Velocity directive begins with # and followed by the directive’s name. Some of the frequently used Velocity directives are as follows:
(7.3.2.1) **#set**: this directive is used to assign a value to a variable.

(7.3.2.2) **#if, #elseif, and #else**: these directives are used to decide whether some text should be included in the output based on a conditional statement.

(7.3.2.3) **#foreach**: this directive is used to iterate through a list of variables.

(7.3.2.4) **#macro**: this directive is used as a means to create a reusable script. It is especially useful if you need to call a certain line of scripts repeatedly.

### 7.3.2.1 #set statement

In Velocity, you can declare a variable and set its value by using the **#set** directive, for example:

```velocity
###String
#set ($var = "abc")

###Boolean
#set($var2 = true)

###Value
#set($var3 = 10)
```

The above example shows that `$var` has a string value, `$var2` has a Boolean value, and `$var3` has a numerical value. The **#set** directive can also set an array to a declared Velocity variable. To assign an array to a variable, type, for example:

```velocity
#set($var = ["a", "b", "c")
```

### 7.3.2.2 #if, #elseif, and #else statement

The **#if** directive allows you to include some text when generating a document, on condition that the if-statement is true, type, for example:
The variable: $condition will be evaluated to determine whether it is true, which will happen under certain circumstances:

- $condition is a Boolean (true/false) that has a ‘true’ value
- $condition is not ‘null’.
- $condition is a comparison, which is evaluated and returns ‘true’.
- $condition is a compound logical statement, which is evaluated and returns ‘true’.

If $condition returns ‘true’, the content between the #if and #end statements becomes the output. In this case, if $condition is true, the output will be: "Hello World". Conversely, if $condition returns ‘false’, there will be no output.

The #elseif or #else element can be used with an #if element. Note that Velocity Engine will stop at the first expression that is found to be true. The following example shows you how to add #elseif and #else to the #if statement:

```
#if( $condition )
  Content 1
#elseif( $condition2 )
  Content 2
#elseif( $condition3 )
  Content 3
#else
  Content 4
#end
```

From the above example, let us assume that $condition1 is false, $condition2 is true, and $condition3 is true. The output for this conditional block will be Content 2 because $condition2 comes before $condition3 even though both of them are true.

Comparing Values and Logical Operators

So far, $condition in the #if directive is assumed to be a Boolean value, however, like Java, Velocity supports the syntax to compare (greater than (>), less than (<), and is equal to (==)) two variables
and the Logical Operators (logical AND (&&), logical OR (||), and logical NOT (!)) which will return a Boolean value.

In Velocity, the equivalent operator ("==") can be used to compare a string, value, or objects. Note that the semantics of the equivalent operator ("==") are slightly different than those of Java where the equivalent operator ("==") can only be used to test object equality. In Velocity, the equivalent operator ("==") can be used to directly compare numbers, strings, or objects. When comparing two objects with different classes, the string that represents the objects is compared. The following example compares two variables to see whether or not they are equal:

```velocity
#set ($var1 = "cat")
#set($var2 = "dog")
#if ($var1 == $var2)
Var1 equals Var2.
#else
Var1 does not equal Var2
#end
```

The comparison operators ("<" and ">") are the same as the ones used in Java. The following example shows how to use the comparison operators to compare two values in which the #if statement will be evaluated 'true' and "Var1 is less than Var2" will be printed out in the generated report:

```velocity
#set ( $var1 = 6 )
#set ( $var2 = 7 )
#if ( $var1 < $var2 )
Var1 is less than Var2
#end
```

The logical AND operator, in Velocity, is represented by &&, and it must have at least two arguments. To write a conditional statement with the logical AND operator, type, for example:
To better understand the above example, let us assume that \( \$var1 < \$var3 \) is argument 1 and \( \$var3 > \$var2 \) is argument 2 (argument 1 and argument 2 will be evaluated separately and will return true/false). The \#if()\n\end\n\end\n
To better understand the above example, let us assume that \( \$var1 < \$var3 \) is argument 1 and \( \$var3 > \$var2 \) is argument 2. According to the example, argument 2 is ‘false’ because \( \$var2 \) is less than \( \$var3 \) not the other way around. Since argument 1 is true, Velocity Engine does not need to look at argument 2 anymore, either ‘true’ or ‘false’, the expression will be ‘true’ and Content 1 will be the output. Basically, the Logical OR operator requires only one argument to be ‘true’ (either argument 1 or argument 2) to make an expression ‘true’. In the case that both argument 1 and argument 2 are ‘false’, there will be no output because the expression is ‘false’.

The Logical NOT operator needs only one argument, for example:
As shown in the above example, !$bool is evaluated ‘true’ and Content 1 is the output. But if $bool is ‘true’, then !$bool will be ‘false’ and there will be no output.

### 7.3.2.3 #foreach statement

The #foreach directive will iterate through a list of variables and assign the values on the list to a local variable, for example:

```
#foreach ($localClass in $Class)
$localClass.name
#end
```

The above example shows that $Class is a list of class elements in Cameo Requirements+. The #foreach directive will iterate through the $Class list and assign the values found on the list to $localClass in each iteration. The code between #foreach and #end will be processed and the result will print the class name in each iteration. For example, if $Class contains class elements named “a”, “b”, and “c”, the output from the above example will be as follows:

```
a
b
c
```

### 7.3.2.4 #macro statement

Under certain circumstances, you might find that you often repeat several lines of VTL codes in multiple areas inside your template. To solve this problem, Velocity provides the #macro directive that allows you to repeat a segment of VTL codes. The first thing you have to do is to declare or create the macro itself, for example:

```
#macro (HelloWorld)
Hello World! It is such a beautiful world!
#end
```
Then within the template, you can call this macro and it will print “Hello World! It is such a beautiful world!”. To call this macro, type, for example:

```text
#HelloWorld()
```

When the macro in the above example is passed on to the Velocity Template Engine, “Hello World” “It is such a beautiful world!” will be printed out as a result. You may wonder why this is important, but imagine having to print a note with 10 lines of text in different parts of the document. The Velocity macro helps you save the space and time. It also allows you to pass the value from a variable on to a method parameter as you would using a Java method. To do this you have to create a macro, for example:

```text
#macro( myMacro $color $somelist )
#foreach( $something in $somelist )
$something is $color.
#end
#end
```

In the above example, you have created a macro called “myMacro” that accepts two variables, the first variable is supposed to be a string and the second variable called “somelist” is supposed to be a list (note that when creating a macro that accepts variables always remember what types of variables it accepts, otherwise an error will occur during report generation). To call myMacro, type, for example:

```text
#set ($list = [“A Rose”, “Blood”, “Strawberry”])
#set ($color = “red”)  
#myMacro($color $list)
```

The above example shows that “myMacro” has been called and there are two variables in the brackets, $color and $list, separated by a space. The result of the example will be as follows:
Report Wizard provides three directives: (7.4.1) **#forrow**, (7.4.2) **#forcol**, and (7.4.3) **#forpage** to allow you to use Velocity to generate outputs other than web documents. With these directives, you can eliminate problems that usually occur when using Velocity to generate an rtf document, for example, **#foreach** generates broken rows in a table.

### 7.4.1 #forrow Directive

This directive is used in Rich Text Format, OpenDocument Text, and OpenDocument Spreadsheet documents. Whenever **#forrow** iterates through a list, it creates a table row.

The Velocity Template Language does not support loops inside a table structure. However, the Report Wizard engine introduces a new custom syntax to allow loops in the table structure in order to clone table rows. The following example will show you how to create a table that will print out the name of a Use Case element and its owner in a table.

To use the **#forrow** directive:

1. Open Microsoft Word or OpenOffice.org Writer.
2. Create a two-column and two-row table (Figure 352).
3. In the second row, first column type: “#forrow ($uc in $UseCase) $uc.name”, and in the second row, second column type: “$uc.owner.humanName #endrow” (Figure 353).
4. Save the template and import it to Report Wizard.

5. Generate a Use Case report. The report will be as shown in Figure 354.
7.4.2 #forcol Directive

This directive is designed only for OpenDocument Spreadsheet (Figure 355). Whenever #forcol iterates through a list, it creates a table column. This directive allows loops over the column.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>#forcol($uc in $UseCase) $uc.name #endcol</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 355 -- Using #forcol in Spreadsheet*
Based on the example in Figure 355, the engine will generate a report with different columns for each Use Case name (Figure 356).

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FillOrder</td>
<td>LocateProdu</td>
<td>ReceiveProd</td>
<td>ShipOrd</td>
<td>StockPro</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 356 -- OpenDocument Spreadsheet #forcol Output*

You can combine both `#forrow` and `#forcol` and produce a more complex output report (Figure 357).

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>#forrow($bound in $SystemBoundary) #forcol($child in $bound.children) $child.name #endcol #endrow</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 357 -- Using #forcol and #forrow*

Let’s break the template (Figure 357) into portions for ease of understanding. This template will list the System Boundary into each row of the first column, and then the next columns will list all the names of children under the System Boundary.
7.4.3 #forpage Directive

This directive is used in Rich Text Format document, OpenDocument Text, OpenDocument Spreadsheet, and OpenDocument Presentation. Whenever #forpage iterates through a list, it creates a new page for a Rich Text Format document and OpenDocument Text, a new sheet for OpenDocument Spreadsheet, or a new slide for OpenDocument Presentation. The #forpage directive is used to create a new slide in OpenOffice.org Impress or a new page in RTF. The implementation of this directive is the same as that of #forrow:

```
#forpage($dia in $Diagram)
$uc.name
#endpage
```

The above example will create many pages, each containing a Use Case element's name. The number of pages created depends on the number of Use Case elements in the $UseCase list.

In ODP, #forpage is created as a means to create a new slide.

To create a new presentation slide:

1. Open OpenOffice.org Impress.
2. Create two text boxes.
3. In one of the boxes type, for example:

```
#forpage($dia in $Diagram)
$dia.name
```
Each page of the generated report will display the name of the diagram as the page title and the image of the diagram (Figure 360).
Inventory Control System

Figure 360 -- Generated result
## Appendix A: Cameo Requirements+ Terminology

Table 64 -- Cameo Requirements+ Terminology

<table>
<thead>
<tr>
<th>Terms</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Action</strong></td>
<td>When a user performs an action in the system. The response of the system is also considered as an action.</td>
</tr>
<tr>
<td><strong>Action collection</strong></td>
<td>A collection of all Action templates created in the Storyboard library.</td>
</tr>
<tr>
<td><strong>Actor</strong></td>
<td>An actor is an entity with a specific role to perform the designated action.</td>
</tr>
<tr>
<td><strong>Actor Collection</strong></td>
<td>A collection of all Actor templates with Predefined roles, available in the Storyboard library.</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>An attribute is a name-value pair. It can represent a property name and value. Attributes of an object is like an adjective that describes the object.</td>
</tr>
<tr>
<td><strong>Base Node Type</strong></td>
<td>A node type that is linked to another node type underneath. Node types are linked within an inheritance hierarchy.</td>
</tr>
<tr>
<td><strong>Child Node</strong></td>
<td>A node that is contained within another node. The containing node is the parent.</td>
</tr>
<tr>
<td><strong>Decision scene</strong></td>
<td>A scene that involves a decision to be taken according to the Guard value.</td>
</tr>
</tbody>
</table>
# Cameo Requirements+ Terminology

**Dependency Matrix**

Cameo Requirements+ Dependency Matrix is a tool to display dependencies between each row and column element. A dependency is a relationship that has a direction. It can also have a name. However, a Cameo Requirements+ dependency ALWAYS has a direction and a name.

A dependency between a row and column element can have three directions:

- From column to row direction
- From row to column direction
- Two-way direction (from column to row and from row to column)

Dependency relationships can be of two types: (i) direct and (ii) indirect.

(i) A direct dependency is a relationship from a source node that directly links to a target node (for example, Requirement R1 traces to Requirement R2, so R1 has a direct dependency to R2). This dependency type will be displayed as a thick arrow.

(ii) An indirect dependency is a relationship from a source that has no direct link to a target. It goes through one or more intermediate nodes before linking to the target node (for example, Requirement R1 traces to Requirement R2 and Test Case T1 tests Requirement R2, so R1 has a direct dependency to R2 and it has an indirect dependency to T1). This dependency type will be displayed as a dotted arrow.

A dependency path is a list of source nodes having a direct or indirect link to intermediate and target nodes (for example, requirement R1 traces to Requirement R2 and Requirement R2 is tested by Test Case T1, so the dependency path is R1 → R2 → T1).

<table>
<thead>
<tr>
<th>Derived Node Type</th>
<th>A node type that is linked to another node type above it in the inheritance hierarchy.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Relationship Node</td>
<td>A direct relationship node is a one-way relation node.</td>
</tr>
<tr>
<td>Glossary</td>
<td>An alphabetical listing of special terms as they are used in a particular subject area. They often have more in-depth explanations than would customarily be provided by dictionary definitions.</td>
</tr>
<tr>
<td>Glossary Definition</td>
<td>An exact meaning of a glossary term, in a specific domain.</td>
</tr>
<tr>
<td>Glossary Domain</td>
<td>A collection of domain-related glossary definitions.</td>
</tr>
<tr>
<td>Glossary Term</td>
<td>A word or phrase used to describe something.</td>
</tr>
</tbody>
</table>
### Impact Analysis
A technique that helps users think through the full impact of a proposed change. As such, it is an essential part of the evaluation process for major decisions. Its difference also gives the ability to detect problems before they occur, so that organizations can develop contingency plans to handle the issues smoothly. This can make a difference to create a well-controlled and seemingly-effortless project management instead of a chaotic one.

### Information
Information is a representation of a collection of data that gives a context.

### Link
A link is anything that connects or makes an association. A link establishes the presence of a relationship.

### Narrative scene
A normal scene in which no decision is required to take either by the system or the user.

### Node
A node is a representation of information within the schema. A representation of a node can be compared to an object in the Object-Oriented concept; a node can inherit from another node and it can contain other nodes.

### Node Type
A type or class in which a node belongs. A single node is a specific instance of a node type. Node types follow an inheritance hierarchy.

### Parent Node
Nodes are linked within a containment hierarchy. A parent node is always higher in the hierarchy than its children.

### Perspective
A perspective is a visual container for a set of views and editors (parts). The layout of views and editors contained in each perspective can be reorganized to suit your own needs.

### Relatable Node
A relatable node is a specialized node that can contain a relationship node.

### Scene
A shot in the story constituting related action. A story step describes what action is performed in a scene, view, or prospect.

### Schema
A schema defines the model and structure of how the information is going to be kept to comply with the business model. A schema identifies the entities and their attributes. A schema also contains information on rules of used and allowed values.

### State Variable
A common value type assigned for a series of story steps. The value of the state variable varies from one step to another, but the value type will be same.
### APPENDICES

**Appendix A: Cameo Requirements+ Terminology**

<table>
<thead>
<tr>
<th><strong>State Variable Collection</strong></th>
<th>A collection of predefined value types in the Storyboard library that can be reused.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Story</strong></td>
<td>A series of steps in storytelling. Note that a story may consist of several scenarios as dictated by different input data and different paths due to values or decisions.</td>
</tr>
<tr>
<td><strong>Story Step</strong></td>
<td>An operation to execute from input data to output data.</td>
</tr>
<tr>
<td><strong>Traceability</strong></td>
<td>In software development, the term traceability (or Requirements Traceability) refers to the ability to link requirements back to stakeholder rationales and forward to corresponding design artifacts, code, and test cases.</td>
</tr>
<tr>
<td><strong>Tree</strong></td>
<td>In graph theory, a tree is a graph in which any two vertices are connected by exactly one path. Alternatively, any connected graph with no cycles is a tree.</td>
</tr>
<tr>
<td><strong>Tree Structure</strong></td>
<td>A tree structure is a way of representing the hierarchical nature of a structure in a graphical form. It is named a &quot;tree structure&quot; because the graph looks a bit like a tree, even though the tree is generally shown upside down compared with a real tree; so the root is at the top and the leaves at the bottom.</td>
</tr>
</tbody>
</table>
Appendix B: Cameo Requirements+ Special Node Type

1. Working with Image Nodes

An image node is used for storing and viewing an image file. You can also use Image Viewer for managing the images.

1.1 Creating Image Nodes

To create an image node:

1. Select a Project, Requirement, Test Case, or Image node from the Data Containment Tree.
2. Either right-click and select Add Image Node or click Edit > Add Image Node.
3. Enter the details and import the image file to Cameo Requirements+.
Figure 361 -- An Image Node in the Editor Pane
### 1.2 Managing Images Using Image Viewer

You can view an image through *Image Viewer*. Cameo Requirements+ Image Viewer has all the functionalities of a standard image viewer program include zoom in and out, zoom to original size, view image in full screen, and so on. To open *Image Viewer*, click the *Open Viewer* button in the editor pane.

<table>
<thead>
<tr>
<th>Name</th>
<th>Editor type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Single Line</td>
<td>The name of the image file.</td>
</tr>
<tr>
<td>Description</td>
<td>Multiline</td>
<td>The description of the image file.</td>
</tr>
<tr>
<td>Image</td>
<td>Image</td>
<td>The area that manages an image file.</td>
</tr>
</tbody>
</table>

**NOTE** The *Open Viewer* button will be enabled when an image has already been imported to the image area.
APPENDICES
Appendix B: Cameo Requirements+ Special Node Type

**Image Viewer** provides a toolbar for managing your images.

![Image Viewer Toolbar](image.png)

*Figure 363 -- The Image Viewer Toolbar*
### Table 66 -- The Image Viewer Toolbar

<table>
<thead>
<tr>
<th>Name</th>
<th>Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export</td>
<td></td>
<td>To export the image from Cameo Requirements+ to the computer file system.</td>
</tr>
<tr>
<td>Best Fit</td>
<td></td>
<td>To fit an image into the dialog size with a fixed aspect ratio.</td>
</tr>
<tr>
<td>Actual size</td>
<td></td>
<td>To display an image in its original size.</td>
</tr>
<tr>
<td>Full Screen</td>
<td></td>
<td>To display an image in full screen mode.</td>
</tr>
<tr>
<td>Zoom In</td>
<td></td>
<td>To increase or decrease an image size.</td>
</tr>
<tr>
<td>Zoom Out</td>
<td></td>
<td>Press the – or + key to zoom in and out.</td>
</tr>
<tr>
<td>Close Dialog</td>
<td></td>
<td>To close Image Viewer.</td>
</tr>
</tbody>
</table>

### 2. Working with Multimedia Nodes

Multimedia nodes are used for storing your multimedia files in Cameo Requirements+ such as audio and video. You can create, modify, or delete these nodes.

#### 2.1 Creating Multimedia Nodes
To create a multimedia node:

1. Select a Project, Requirement, Test Case, Test Case Result, or Multimedia Node node in the tree view.
2. Either right-click and select Add Multimedia Node or click Edit > Add Multimedia Node.
3. Enter the details and import the multimedia file to Cameo Requirements+.

Figure 364 -- A Multimedia Node in Editor Pane

The Cameo Requirements+ Default Schema provides the fields for entering your multimedia details as follows.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Editor Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Single Line Text</td>
<td>The name of the multimedia file.</td>
</tr>
<tr>
<td>Description</td>
<td>Multiline Text</td>
<td>The multimedia file description.</td>
</tr>
<tr>
<td>MultimediaFile</td>
<td>File Import/Export</td>
<td>The field that manages a multimedia file.</td>
</tr>
</tbody>
</table>

Cameo Requirements+ provides a multimedia player for playing multimedia files include SWF, MP4, MP3, and so on. You can play a multimedia file using the Summary tab in the editor pane.
Figure 365 -- Playing a Multimedia Node in the Editor Pane Summary View