Inquirement

From Brainstorm to Breakthrough

User Manual

V1.1.1 2026-10-03

1 Table of Contents

1 Introduction 7
2 The process 8
2.1 Usage Scenarios 8
2.1.1 Brainstorm: Generate ideas without limits 8
2.1.2 Analyse: Reveal insight layer by layer 9
2.1.3 Explore: Uncover new or existing know-how 9
2.1.4 Simplify: Prevent over-engineering 9
2.1.5 Reuse: Stop reinventing the wheel 9
2.1.6 Identify and mitigate risks 9
2.2 Inquirement objects 10
2.2.1 Project 10
2.2.2 User Story 10
2.2.3 Mandate 10
2.2.4 Requirement 11
2.2.5 Integration Requirement 11
2.2.6 Design Problem 12
2.2.7 Option 12
2.2.8 Argument 12
2.2.9 Evidence 13
2.3 Semantic Relations 14
2.3.1 Subject 14
2.3.2 Verb 14
2.3.3 Capability 14
2.4 Readiness Level 15
2.5 Project definition 16
2.6 Problem statement 16
2.7 Define Solutions 17
2.8 Validate the Solutions 18
2.9 Inspect the Requirements 19
2.10 Select the best Solution 19
3 Inquirement window 20
3.1 Tree View 20
3.2 Workspace 21
3.2.1 Project tab page 21

3.2.2 Problems tab page 22
3.2.3 Options tab page 23
3.2.4 Validation tab page 24
3.2.5 Requirements tab page 25
3.2.6 V-diagram tab page 26
4 Menu Bar 27
4.1 File Menu 27
4.1.1 New 27
4.1.2 Open 27
4.1.3 Save 27
4.1.4 Save As 27
4.1.5 Export pdf 28
4.1.6 Export Requirements pdf 28
4.1.7 Export ReqIF 28
4.1.8 Export SysML 28
4.1.9 Import comments 29
4.1.10 Close 29
4.1.11 Exit 29
4.1.12 Recent files 29
4.2 Edit Menu 30
4.2.1 Undo 30
4.2.2 Redo 30
4.2.3 Find all comments 30
4.2.4 Find Text 31
4.2.5 Find Object With UID 32
4.3 View Menu 33
4.3.1 Appearance 33
4.3.2 Set appearance as default 34
4.4 Tools Menu 35
4.4.1 Customize Parameters 35
4.4.2 Project References 36
4.4.3 Review Integration Requirements 37
4.4.4 Review Evidence 37
4.4.5 Renew License 38
4.4.6 Buy License 38
4.5 Help Menu 39

4.5.1 About ... 39
4.5.2 Privacy Policy ... 39
4.5.3 Example Projects ... 39
4.5.4 Explain Objects ... 40
4.5.5 Keyboard Shortcuts or Hotkeys ... 40
5 Toolbar ... 41
5.1 New ... 41
5.2 Open ... 41
5.3 Save ... 41
5.4 Save As ... 41
5.5 Undo ... 41
5.6 Redo ... 41
5.7 Find ... 42
5.8 Find object with UID ... 42
5.9 Object Filter ... 42
5.10 Image of Current Tab Page ... 43
5.11 Take Screen Shot Image ... 44
5.12 Decrease Font Size ... 45
5.13 Increase Font Size ... 45
5.14 Zoom Out ... 46
5.15 Reset Zoom ... 46
5.16 Zoom In ... 47
6 Treeview Menu ... 48
6.1 Show related objects ... 48
6.2 Show and edit comments ... 48
7 Objects ... 49
7.1 Parameter ... 49
7.2 Reference ... 49
7.3 Generic context menu ... 50
7.3.1 Show and Edit comments ... 50
7.3.2 Readiness Level Icon ... 51
7.3.3 Select object in tree ... 51
7.3.4 Deselect object in tree ... 51
7.3.5 Delete object ... 51
7.3.6 Link a Reference ... 52
7.3.7 Add a Reference ... 52

7.3.8 Open a Reference 52
7.3.9 Edit References 53
7.3.10 Remove Reference 53
7.3.11 Copy object 54
7.3.12 Paste object 54
7.3.13 Edit tooltip text 55
7.3.14 Show related objects 55
7.4 Project Object 56
7.4.1 Change Name 56
7.4.2 Change Description 56
7.4.3 Add User Story 56
7.4.4 Double Click 57
7.5 User Story Object 58
7.5.1 Change the Subject 58
7.5.2 Change the Verb 58
7.5.3 Change the Capability 59
7.5.4 Edit description 59
7.5.5 Derive Requirement 59
7.5.6 Define Requirement 60
7.5.7 Define Goal 61
7.5.8 Double Click 61
7.6 Mandate Object 62
7.6.1 Change Remark 62
7.6.2 Change Required Readiness Level 62
7.6.3 Derive Requirements from User Stories 62
7.6.4 Review Evidence 63
7.7 Requirement Object 64
7.7.1 Change the Subject 64
7.7.2 Change the Verb 64
7.7.3 Change the Capability 65
7.7.4 Change the Quantification 65
7.7.5 Change to a Goal 66
7.7.6 Change to a Functional Requirement 66
7.7.7 Edit Importance 66
7.7.8 Copy AI prompt 67
7.7.9 Paste AI answer 67

7.7.10 Allocate Requirement 68
7.7.11 Show Why 69
7.7.12 Add Design Problem 69
7.7.13 Add to Existing Design Problem 70
7.7.14 Hide or Show Nodes 70
7.7.15 Add Argument 71
7.7.16 Double Click 71
7.8 Design Problem Object 72
7.8.1 Remove Requirement 72
7.8.2 Add Requirement 73
7.8.3 Add Option 73
7.8.4 Copy AI prompt 74
7.8.5 Paste AI answer 75
7.9 Option Object 76
7.9.1 Option Readiness Level Icon 76
7.9.2 Change the Subject 77
7.9.3 Add Integration Requirement 77
7.9.4 Add Sub Option 77
7.9.5 Add Argument 78
7.9.6 Add all Arguments 78
7.9.7 Hide or Show Nodes 79
7.9.8 Double Click 80
7.10 Sub-Option Object 80
7.11 Argument Object 81
7.11.1 Toggle Validates – Invalidates 81
7.11.2 Add Goal 82
7.11.3 Add Requirement 82
7.11.4 Link to Existing Requirement 83
7.11.5 Unlink child Requirement 83
7.11.6 Add Evidence 84
7.11.7 Link to Existing Evidence 84
7.11.8 Unlink child Evidence 85
7.11.9 Double Click 85
7.12 Evidence Object 86
7.12.1 Change the Subject 86
7.12.2 Change the Verb 86

7.12.3 Change the Capability 87
7.12.4 Change the Quantification 87
7.12.5 Change the Readiness Level 87
7.12.6 Double Click 88
8 Inquirement files 89
8.1 Default Possible Parameter Activity 89
8.2 Custom Possible Parameter 89
8.3 Project Object 90
8.4 User Story Object 91
8.5 Mandate Object 91
8.6 Design Problem 92
8.7 Requirement 92
8.8 Option 93
8.9 Argument 94
8.10 Evidence 94
9 Installing Inquirement 95
10 Uninstall Inquirement 99
11 Versions 100
12 Privacy policy 103
13 License Agreement 106
14 How to report a bug 107
15 Known Issues 108
16 Inquirement community 109

1 Introduction

Inquirement enables product teams, designers, and innovators to explore competing solutions, evolving ideas, and requirements in increasing levels of detail with full traceability. It creates insight that shapes the right questions and supports translating them into AI prompts with readable response formats.

Inquirement is not just a tool — it's where know-how thrives.

Automation and AI radically reduce the cost of the application of know-how. This is why well-structured and human readable know-how has gained so much value. Inquirement supports this, while still being flexible enough to be used during brainstorm sessions. Its simplicity and semantic core help you organize the ideas and ask the right questions – supporting creativity and transform it to value.

Learn more about Inquirement on our website.

2 The process

The process is built to find a Solution based on Requirements. It is built to gradually explore possible solutions in increasing levels of detail and to discover derived Requirements that are applicable for each Option. The process creates a branch structure, where one of the branches leads to the Optimal Solution.

The process roughly follows the tabs shown in the upper right corner of the Inquirement window (see chapter 3 for explanation of the layout). The following sections will explain the Inquirement steps in more detail.

2.1 Usage Scenarios

Inquirement supports human interaction with know-how. It helps to organize ideas, while asking the right questions – supporting creativity and transform it to value.

Inquirement can be used individually and in groups. The sections below discuss Inquirement's usage scenarios.

2.1.1 Brainstorm: Generate ideas without limits

The Inquirement process allows generation of Options for complex systems. Work in a group or individually from concept level to detail Requirements that matter. Keep track of alternative Options and find the optimal solution.

2.1.2 Analyse: Reveal insight layer by layer

Experienced people know that the details determine the outcome. Work in a group or individually to formalise the causal connections between Options, detail Requirements and Top Level Requirements. This allows analysis of why a Requirement exists and which Option is invalidated if not met. The validity of each Option is clearly linked through Arguments to Evidence. Options are not dismissed before the killer (detail) Requirement is found. This prevents missing viable Options without a clear reason.

2.1.3 Explore: Uncover new or existing know-how

Inquirement helps you to capture the critical know-how involved – documenting not only the solution, but also the reasons why alternatives were not selected. Uncover the hidden truths of existing solutions, while any inconsistencies are uncovered and potential improvements identified.

2.1.4 Simplify: Prevent over-engineering

Every solution has a reasoning behind it that is not always clear. Unstructured reasoning tends to cause solutions that solve problems that should not be there in the first place. Work in a group or individually to keep only required solutions.

2.1.5 Reuse: Stop reinventing the wheel

A problem will be solved again and again if knowledge is not properly documented. Work in a group or individually to build upon existing knowledge. Take the next step forward instead of reinventing the wheel.

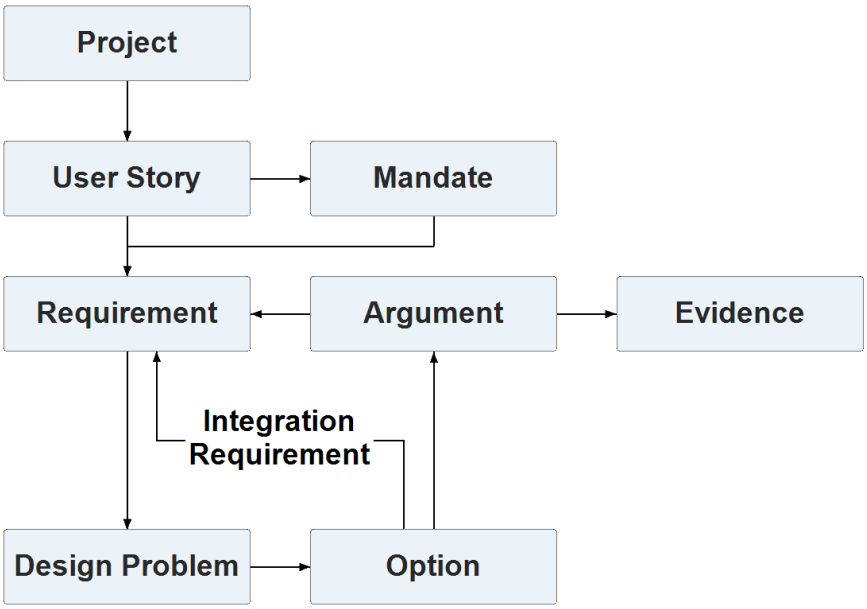
2.1.6 Identify and mitigate risks

Risks are identified by introducing a Requirement that it should not happen. This approach to risk puts people in solving mode instead of remaining focused on everything that can go wrong. Introducing Requirements makes the Option invalid if the risk is not mitigated. Work in a group or individually to identify the risks as Requirements. Define Options that can prevent those risks. Find Evidence that proves the Options are safe to pursue.

See also §2.8.

2.2 Inquirement objects

The Inquirement process is interwoven with the Inquirement objects. The graphical representations of the Inquirement objects are discussed in Chapter 7. This section explains the purpose of the Inquirement objects in the process.



2.2.1 Project

A Project node is the top node. It defines the name and a description of the discovery project.

See also §7.4 Project Object.

2.2.2 User Story

A User Story node is an informal description of what the discovery project should deliver. The User Story node is translated into a formal Requirement through a Mandate Node.

See also §7.5 User Story Object.

2.2.3 Mandate

A Mandate node is the approval of the Project owner to execute the Project. A Mandate defines to which Readiness Level the final project solution must be proven. The levels are derived from technology readiness levels. The Evidence node provides a certain level of confidence that a solution is proven up to a Readiness Level.

See also §7.6 Mandate Object.

2.2.4 Requirement

A Requirement node defines a Requirement that can be:

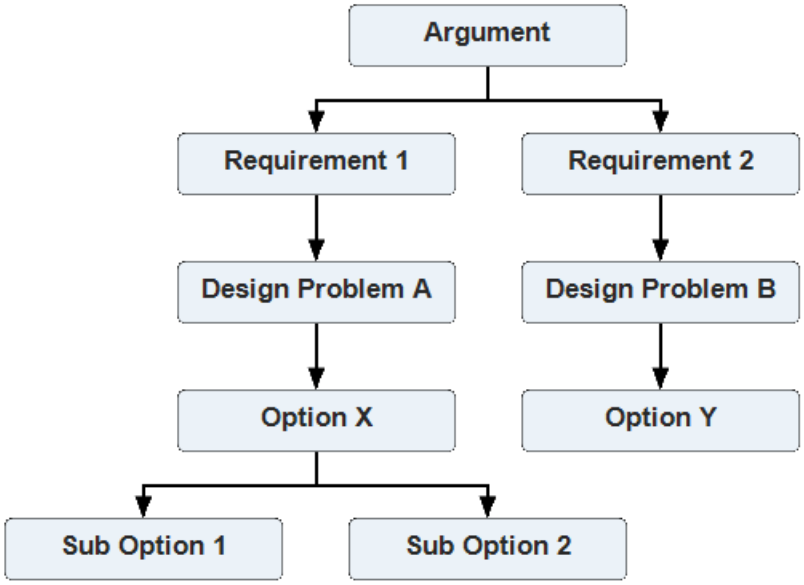
- Functional: If the Requirement does not define any quantitative value.
- Goal: If a number or property must be as small/large as possible.
- Range: If a number or property must be smaller or larger than a certain value.
- Fixed Value: If a number or property must be a certain value.

A Top Level Requirement is derived from a User Story node directly. Integration Requirements are added if multiple sub-options must be integrated into one.

See also §7.7 Requirement Object.

2.2.5 Integration Requirement

It is possible to break up complex problems into multiple Design Problems that only try to solve a subset of the Requirements.



An Integration Requirement is automatically added if more than one Design Problem exists in one branch as shown in the example above.
Option X by itself must satisfy Requirement 1
Option Y by itself must satisfy Requirement 2
The Argument in the example above is invalid if the combination of Design Problems A and B do not satisfy both Requirement 1 and 2. Therefore Option X and Y need an Integration Requirement.

It is also possible to divide Options into Sub-Option Objects. In such cases each Sub-Option Object will automatically receive an Integration Requirement as well.

2.2.6 Design Problem

A Design Problem is the node that collects certain Requirements to properly define the challenge at hand. The Design Problem should collect all Requirements that need to be satisfied. Any Options proposed to solve the Design Problem should satisfy all linked Requirements.

See also §7.8 Design Problem Object.

2.2.7 Option

An Option node describes a possible Solution for a Design Problem. Argument nodes are used to explain why.

It is possible to define Sub-Option Objects of an Option. For instance, to divide the Option into functional elements. In such cases each Sub-Option Object will automatically receive an Integration Requirement because the Sub-Option Object must be integrated with other Sub-Option Objects into the Option.

Options can be:

- Invalid — if the Option does not meet the Requirements
- Valid — if the Option meets the Requirements
- Optimal — if the Option meets the Requirements and either
  has the best Parameter values with respect to the Goals, or
  has the highest Readiness Level if no Goals are defined.

The combination of all preferred valid Options defines the solution for the Project. This combination can be selected and visualized in the V-diagram tab page.

See also §7.9 Option Object.

2.2.8 Argument

An Argument node explains how an Option solves a certain Requirement. The Argument also states the conditions for this Requirement to be valid. These conditions are defined as Requirements or Evidence. While brainstorming, any risk or scepticism about an Option can be efficiently expressed as these conditions. Any doubt is captured and turned into a challenge that can later be solved. Validation tests are required if no direct Evidence is available. Test method Options and Evidence that the test method can deliver proof then need to be found. The cost of the test can be added in the Evidence Parameter value.

See also §7.11 Argument Object.

2.2.9 Evidence

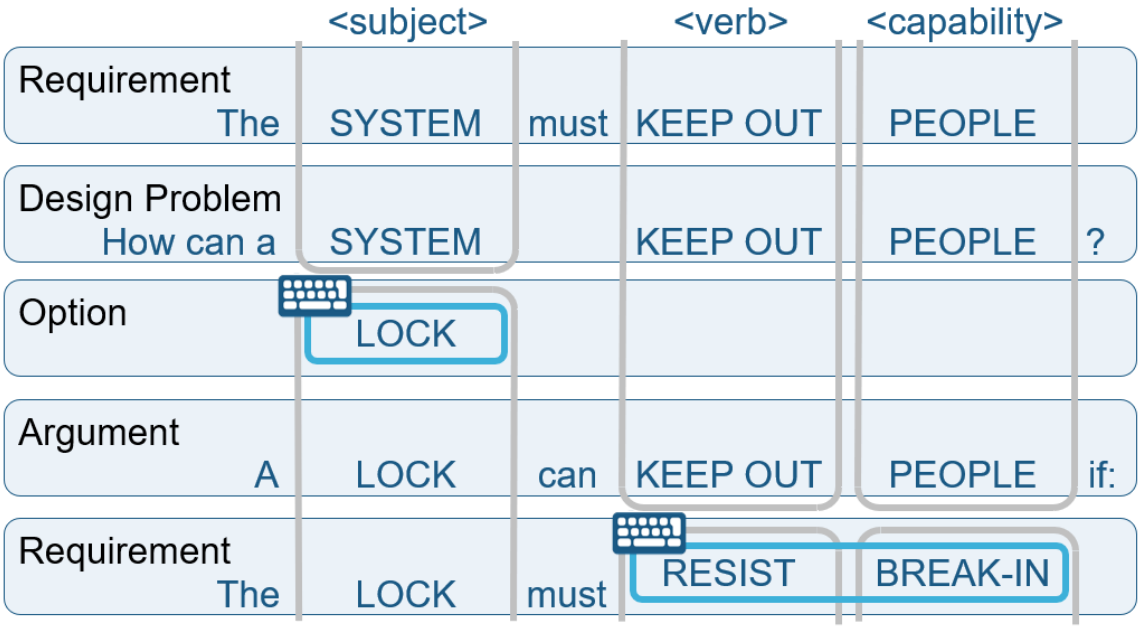
An Evidence node is the reference to Evidence that a certain Option can satisfy a Requirement. An Evidence Object does not share a Subject, Verb or Capability with any other object; it exists as a separate piece of evidence.

Evidence proves the Option validity to a certain Readiness Level. The levels are derived from technology readiness levels. The Mandate node defines to what Readiness Level the final solution must be proven. As the Readiness Levels suggest, the Evidence can have several sources: ‘analysis’, ‘test’ and ‘…’ up to ‘proven in actual environment’.

See also §7.12 Evidence Object.

2.3 Semantic Relations

The Inquirement objects have underlying semantic links. For instance, the subject of a Requirement reappears in a Design Problem as shown in the example below.



The semantic rules within Inquirement will, for instance, define the Design Problem based on the parent Requirement as shown in the example above. The semantic relations drastically reduce the amount of user input needed and ensure data consistency because, for instance, a shared subject is defined only once and reused where needed.

2.3.1 Subject

The Subject is the description of the solution or Option. The Subject is defined in a User Story or Option and shared with relevant objects. Accordingly, a Subject can only be changed in a User Story or Option. Changing the subject will immediately change the subject of all related objects as well, ensuring data consistency.

2.3.2 Verb

The Verb defines what the Subject must do. Never use the word ‘must’, since that is always put before the Verb in a Requirement. The Verb must always be specified.

2.3.3 Capability

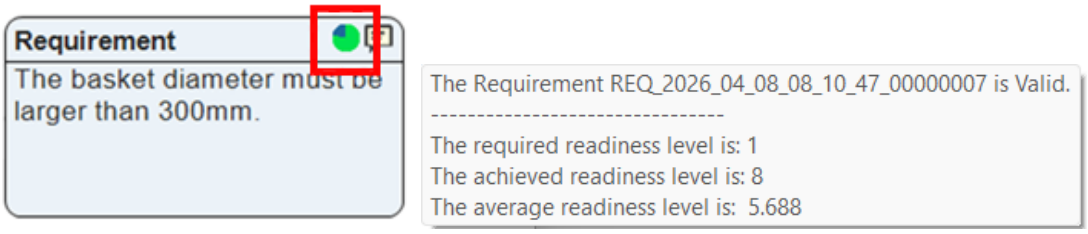
The Capability is a further description of what the Subject must do. In rare occasions the Capability can be left empty if the Verb description is already clear enough.

2.4 Readiness Level

The Readiness Level defines the readiness level to which the solution must be proven. The definitions of each level represent a level of confidence that the solution will actually work as expected. The required Readiness Level, to which level the solution must be proven, is defined in the Mandate. Evidence proves that a solution is valid up to a certain Readiness Level. The overview below shows the Readiness Level and corresponding Evidence type.

Level	Evidence type	Readiness Level
0:	Expectation	No level set
1:	Observed Principle	Basic principles observed
2:	Concept Analysis	Technology concept formulated
3:	Concept Simulation	Experimental proof of concept
4:	Lab Test	Technology validated in a lab
5:	Relevant Env Test	Technology validated in relevant environment
6:	Relevant Env Demo	Technology demonstrated in relevant environment
7:	Operational Demo	System prototype demonstration in operational env
8:	Fully Qualified	System complete and qualified
9:	Fully Operational	Actual system proven in operational environment

An icon (highlighted with a red square) in each Inquirement object shows the achieved Readiness Level. Hovering over the icon will show the Readiness Level of the object (see lower right picture).



See also §7.3.2 for more on the Readiness Level Icon

2.5 Project definition

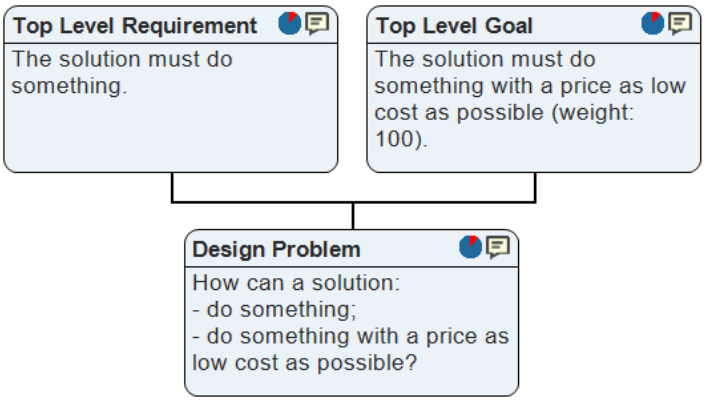
This phase defines the project goal and scope. The project is described in the Project object. The outcome of the project is defined in User Story objects. The Mandate captures the approval of the project owner and defines to which level the final project solution must be proven.

The Top Level Requirements are defined in the project definition step as well. They are derived from the User Story objects. Do not try to create a complete set of Requirements in this phase. Restrict the number of Requirements to Top Level Requirements that are directly derived from User Story objects. The full set of Requirements will evolve during the Inquirement process.

See §3.2.1 for a detailed description of the Project tab page.

2.6 Problem statement

In this phase the Problem Statement is defined in Design Problems. A (set of) Requirements can be selected to define a Design Problem. This can be repeated until all Requirements are covered in a Design Problem. It is best practice to tackle the most critical or most contradicting Requirements first in order to find a lightweight, yet strong solution.



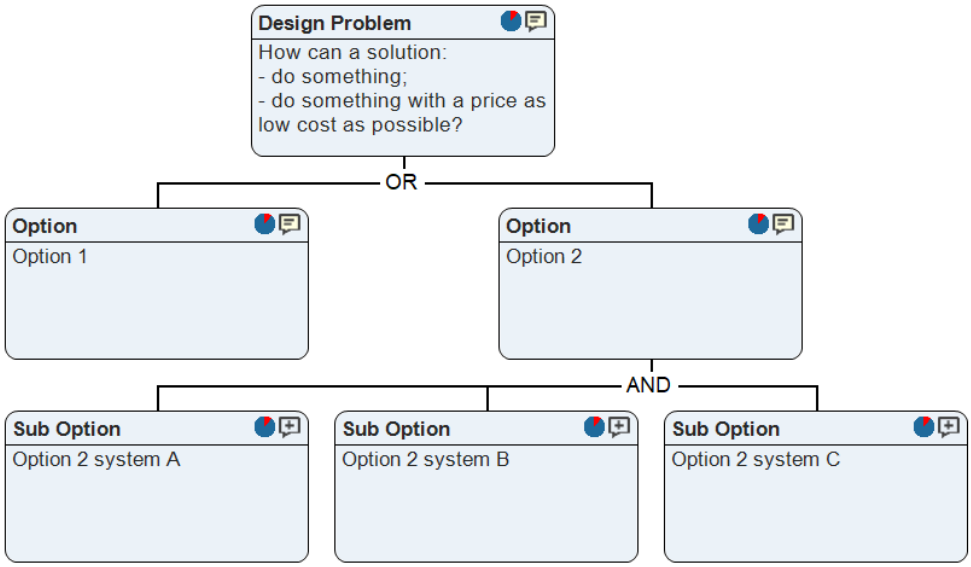
New Requirements will evolve during the validation step (see §2.8). The process will therefore restart at this step until all Requirements have been solved.

An Integration Requirement is automatically created if more than one Design Problem exists in one branch as shown in the example in §2.2.5.

See §3.2.2 for a detailed description of the Problems tab page.

2.7 Define Solutions

In this phase the solutions to the challenges, defined in Design Problems, are defined in the Options.



Options can be generated manually, during a brainstorm, but can also be generated with the help of Large Language Models (LLM). Inquirement can create a LLM prompt on the clipboard based on the Design Problem by using the 'Copy AI prompt' menu item in the Design Problem Object. The prompt can be pasted into any LLM interface. The LLM is instructed to provide the answer in a specific format, so Option objects can be defined by just pasting the LLM answer back into the Design Problem Object by using the 'Paste AI answer' menu item.

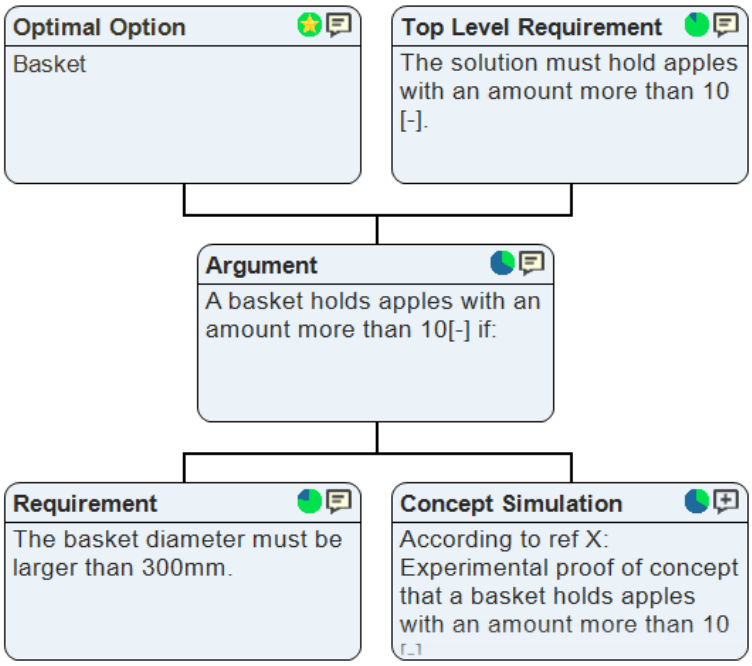
The LLM can only be aware of the specific questions that were asked through this manual step. Full integration with an LLM would expose the entire project. This is not desirable given the confidentiality the information might have.

It is possible to define Sub-Option Objects of an Option to divide it into functional elements for instance. If Sub-Option Objects are added, they will automatically receive an Integration Requirement because the Sub-Option Object must be integrated with other Sub-Option Objects into the Option. See also the example in §2.2.5.

See §3.2.3 for a detailed description of the Options tab page.

2.8 Validate the Solutions

In this phase, Options are evaluated against the Requirements. The Option must satisfy all Requirements that are included in the Design Problem. The evaluation is performed using Argument objects, which state that the Option satisfies a Requirement based on Evidence or assumptions. The assumptions are captured as new Requirements as shown in the example below.



The example shows a typical validation step. The suggested Option ‘basket’ is assumed to be able to ‘hold 10 apples’ if the diameter is at least 300mm. As many Requirements can be added as deemed necessary.

Risks can be addressed as new Requirements too. For instance, the Requirement that the ‘basket’ must not have any concentrated pressure points on the apples can be added to address the risk that the apples might be damaged by the ‘basket’.

Any supportive Evidence can be linked to the Arguments as well. The header of the Evidence shows the Readiness Level to which the Option is proven. Evidence that proves that the Option is not a good solution can be added as well. In that case the Argument has to be set to ‘invalidates’. The Argument and Option will receive a red Readiness Level icon to indicate that the Option is proven to be invalid.

See §3.2.4 for a detailed description of the Validation tab page.

The solutions define several branches in the tree structure. At some point it can happen that the same things need to be defined in several branches. This is generally a sign that these branches share a common parent that has not been identified. In those cases (and in general) the common parent should be defined first. The specifics of this parent can be defined once and passed to its children instead of defining it several times in each child.

2.9 Inspect the Requirements

In this phase, the current set of Requirements is inspected. This can be done in the Requirements tab page, which is an overview of how Arguments led to each Requirement. This allows for a close inspection of the origin and validity of each Requirement. The next step in the solution discovery can be identified and subsequently implemented by going to the Problem statement to continue the process.

Each Requirement has a reason why it exists. This reason can be shown by using the 'Show Why' function in each Requirement. See §3.2.5 for a detailed description of the Requirements tab page.

2.10 Select the best Solution

This phase defines the best solution using the V-diagram tab page. The branch of Inquirement objects diverges into multiple Options. In this step, the best branch can be identified and visualized.

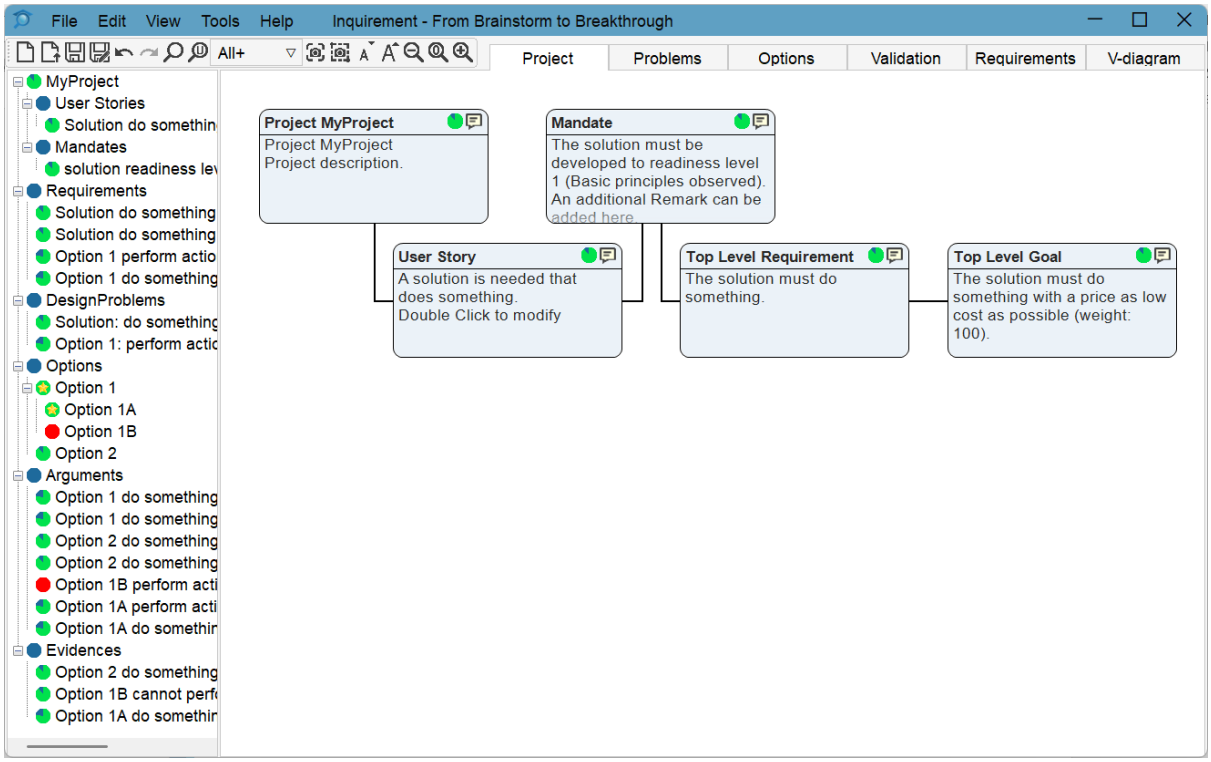
The V-diagram tab page starts with repeating the objects from the Project definition. It then shows successive levels of more and more detailed Options in the left branch of the V-diagram. The right branch shows all Evidence.

The V-diagram displays the optimal Option in each level if an applicable goal Requirement is available. The first valid Option is shown if no goal Requirement is applicable. Options can be switched, so that the V-diagram shows the preferred Option on each level. The combination of all Options from different levels in the V-diagram define the final solution of the Project.

See §3.2.6 for a detailed description of the V-diagram tab page.

3 Inquirement window

The Inquirement window has two main parts. The Tree View on the left and the Workspace on the right. Six different tab pages can be chosen for the Workspace. There is also a Menu Bar and a Toolbar with buttons at the top of the window.



3.1 Tree View

The Tree View on the left shows all Inquirement objects of the Project. The icons in front of each object show the Readiness Level and validity of the object.

The objects in the Tree View can be selected and deselected. This acts as a filter for the view in the Workspace. For instance, the Validation tab page will show the validation for the selected Option.

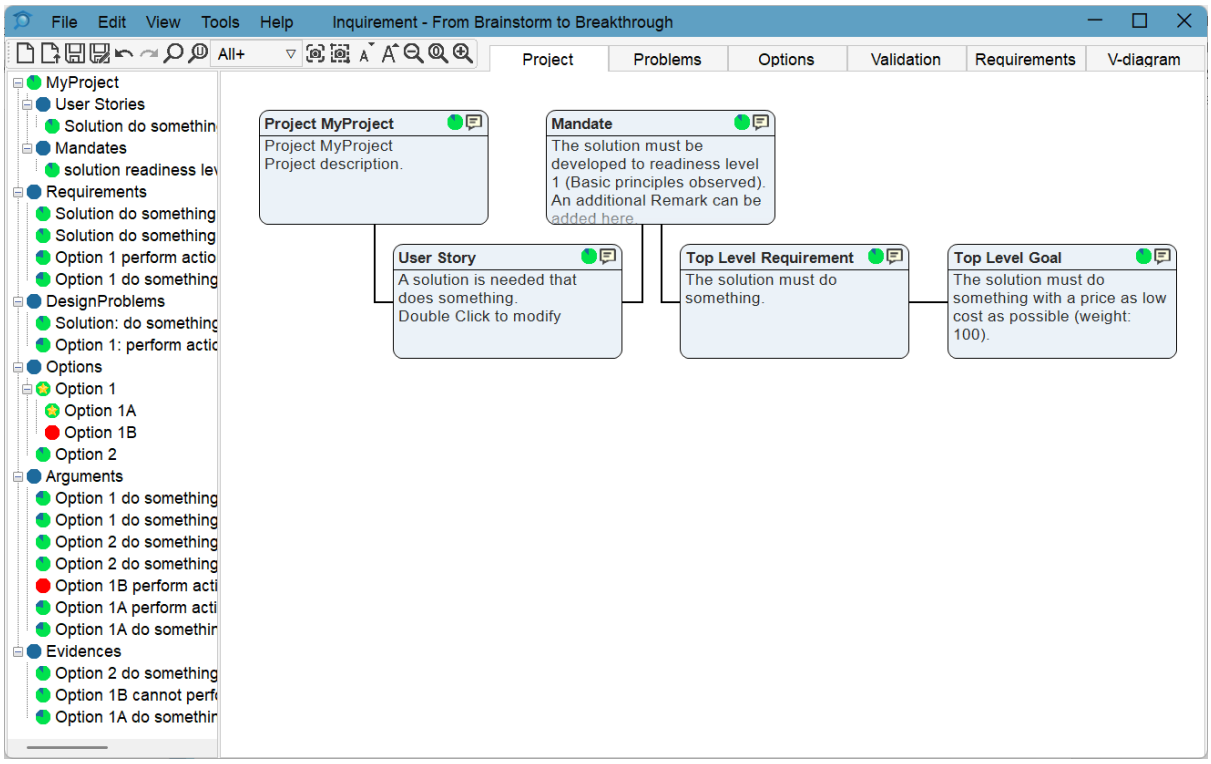
Right-Mouse-Click on each object in the Tree View will show a Treeview Menu. It allows to ‘Show related objects’ or ‘Show and Edit comments’ of each object.

Left-Mouse-Click on each object in the Tree View will select or deselect that object. Multi-Select is activated if CTRL is pressed at the same time.

3.2 Workspace

3.2.1 Project tab page

The Project tab page is intended for the ‘Project definition‘ process step to define the project goal and scope. The tab page shows the Project Object, User Story Objects, Mandate Object and Top Level Requirement Objects.



Start by defining the Project Object by double clicking the Project Object.

Create User Story Objects. Do not go into too much detail, as a broad description of what the Project should do is sufficient in this phase. More detail will be added during later stages of the process.

Define the Mandate Object to state to which Readiness Level the final project solution must be proven.

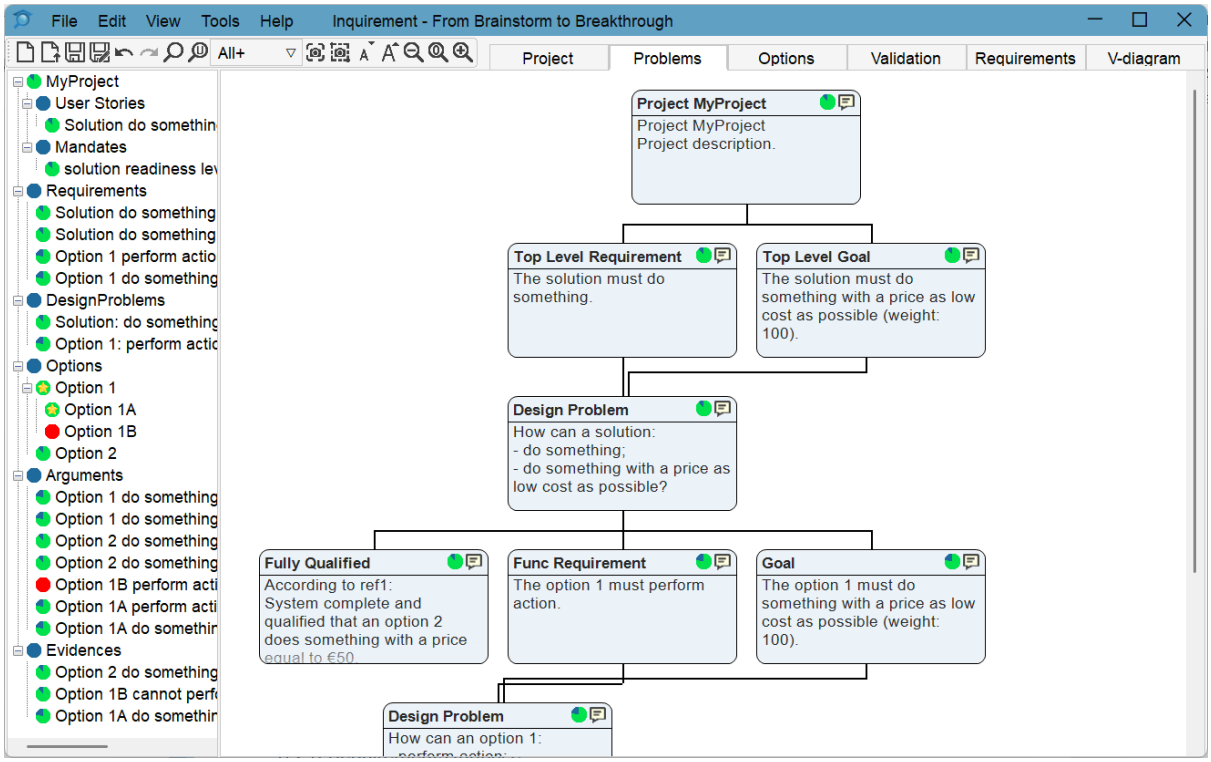
Derive Top Level Requirement Objects from the User Story Objects. Consider adding at least one Goal Requirement. Having a goal enables (automatic) selection of the optimal Option if multiple Options are valid (i.e. satisfy the Requirements).

Determination of the optimal Option uses the Goals in the Top Level Requirements if there are no Goals in the Design Problem that are directly related to the Option.

3.2.2 Problems tab page

The Problems tab page is intended for the Problem statement process step to define which Requirements must be solved by solutions for Design Problems.

The tab page shows the Requirement Objects and Design Problem Objects.



Start with a proper inspection of all Requirement Objects that have not been allocated to a Design Problem Object yet.

Focus on a part of the tree by selecting an object in the Tree View (or Right-Mouse-Button → Select object in tree on the object in the Workspace). The Workspace will show objects below the selected object only.

Try to find contradicting Requirements. For instance, if a certain strength is necessary for an object with a maximum weight. Contradicting Requirement Objects are good candidates for Design Problems.

Aim to tackle the most challenging Design Problems first.

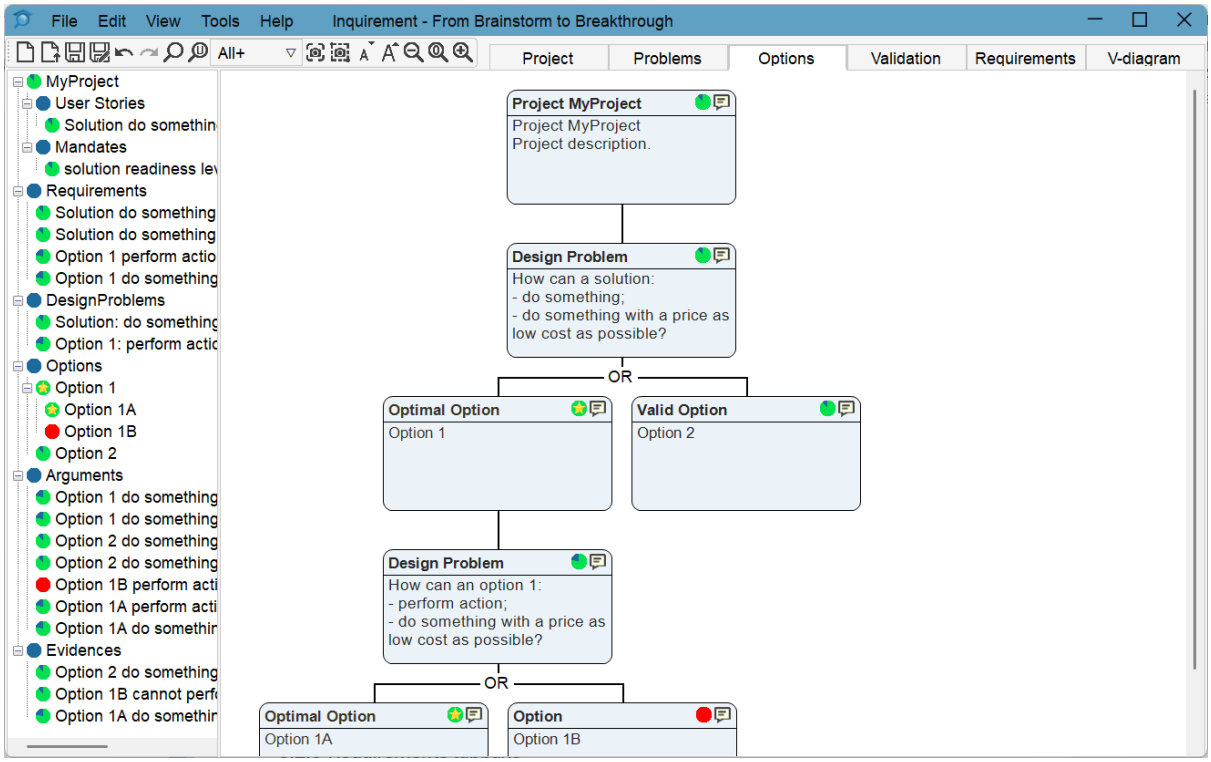
Try not to add all Requirement Objects to one Design Problem Object unless it is absolutely necessary.

Attempt to break up complex problems into multiple Design Problems. Each Design Problem will solve a subset of the Requirements. Be aware that this will automatically create Integration Requirements. These Integration Requirements ensure that all solutions together solve the combination of all Requirements. Automatically created Integration Requirements can be deleted if it is absolutely certain that they do not make sense. Integration Requirements can also be temporarily hidden in the initial phase of the project (see Object Filter button).

3.2.3 Options tab page

The Options tab page is intended for the Define Solutions process step to propose solutions for each Design Problem.

The tab page shows the Design Problem Objects and Option Objects.



Start with a proper inspection of all Design Problem Objects that have not been solved by potential Option Objects yet.

Focus on a part of the tree by selecting an object in the Tree View (or Right-Mouse-Button → Select object in tree on the object in the Workspace). The Workspace will show objects below the selected object only.

Manually add Option Objects with Right-Mouse-Button → Add Option in a Design Problem Object.

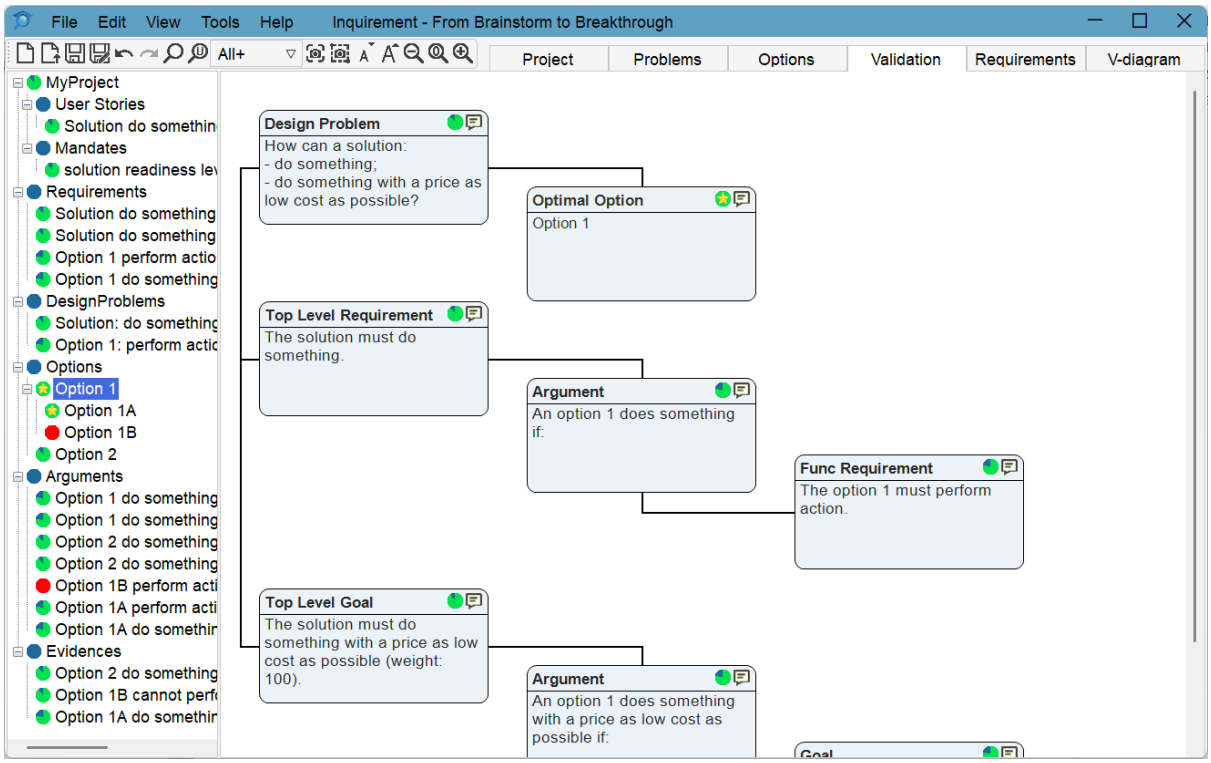
A feature to use a Large Language Model to come up with potential solutions is available in the Design Problem Objects. Right-Mouse-Button → Copy AI prompt will copy an AI prompt on the clipboard. Generate the answer using any Large Language Model. Copy the answer, given in the requested xml format, back in the Design Problem Object using Right-Mouse-Button → Paste AI answer. This will automatically create Option Objects. Inspect the result and modify the resulting Option Objects where needed.

Create a breakdown through Right-Mouse-Button → Add Sub Option in the Option Object. Doing this will automatically create Integration Requirements to ensure that all Sub-Option Objects work together as one to solve all Requirements.

3.2.4 Validation tab page

The Validations tab page is intended for the Validate the Solutions process step to evaluate an Option against the Requirements.

The tab page shows an Option Object, Design Problem Object, Requirement Objects, Argument Objects and Evidence Objects.



Select an Option Object in the Tree View to visualize its validation in the Workspace.

The Option is a solution for a Design Problem, which is shown in the upper left corner.

The Requirements that define the Design Problem are located on the left of the Workspace. The Option must satisfy all Requirements.

The Option can only be valid if the Integration Requirements are valid as well. Integration Requirements are always shown to the right of the Option to make this clear (even if Integration Requirements are hidden in ‘Show related objects’).

Add Argument Objects via Right-Mouse-Button → Add Argument in the Requirement Object or Right-Mouse-Button → Add Argument or Add all Arguments in the Option Object.

Add Requirement Objects using Right-Mouse-Button → Add Requirement or Add Goal in the Argument Object. Add Evidence Object using Right-Mouse-Button → Add Evidence.

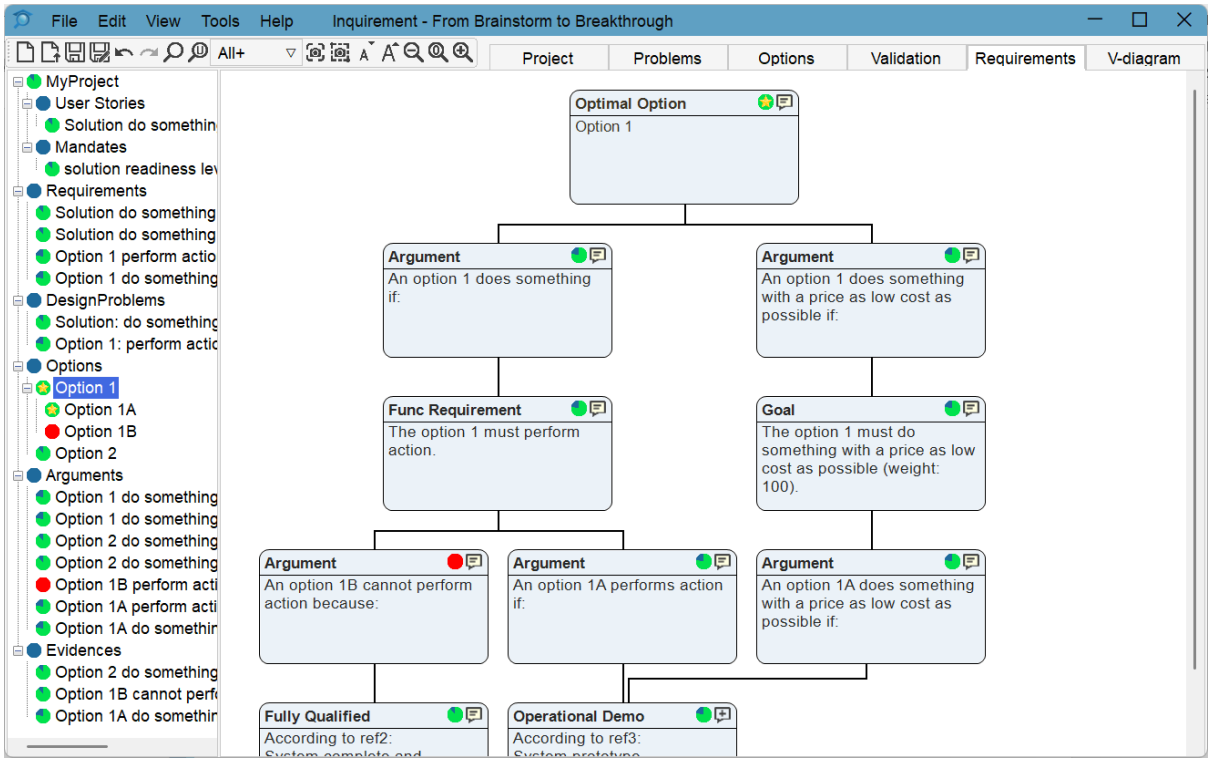
Use help from Large Language Model AI by using ‘Copy AI prompt’ and ’Paste AI ’ in the ‘Requirement Object’ menu items on the left.

See also §7.11.4, §7.11.5, §7.11.7 or §7.11.8 on how to link or unlink existing Requirements or Evidence objects to the Argument Object.

3.2.5 Requirements tab page

The Requirements tab page is intended for the Inspect the Requirements process step to inspect the current set of Requirements.

The tab page shows the Requirement Objects and Argument Objects.



Focus on a part of the tree by selecting an object in the Tree View (or Right-Mouse-Button → Select object in tree on the object in the Workspace). The Workspace will show objects below the selected object only.

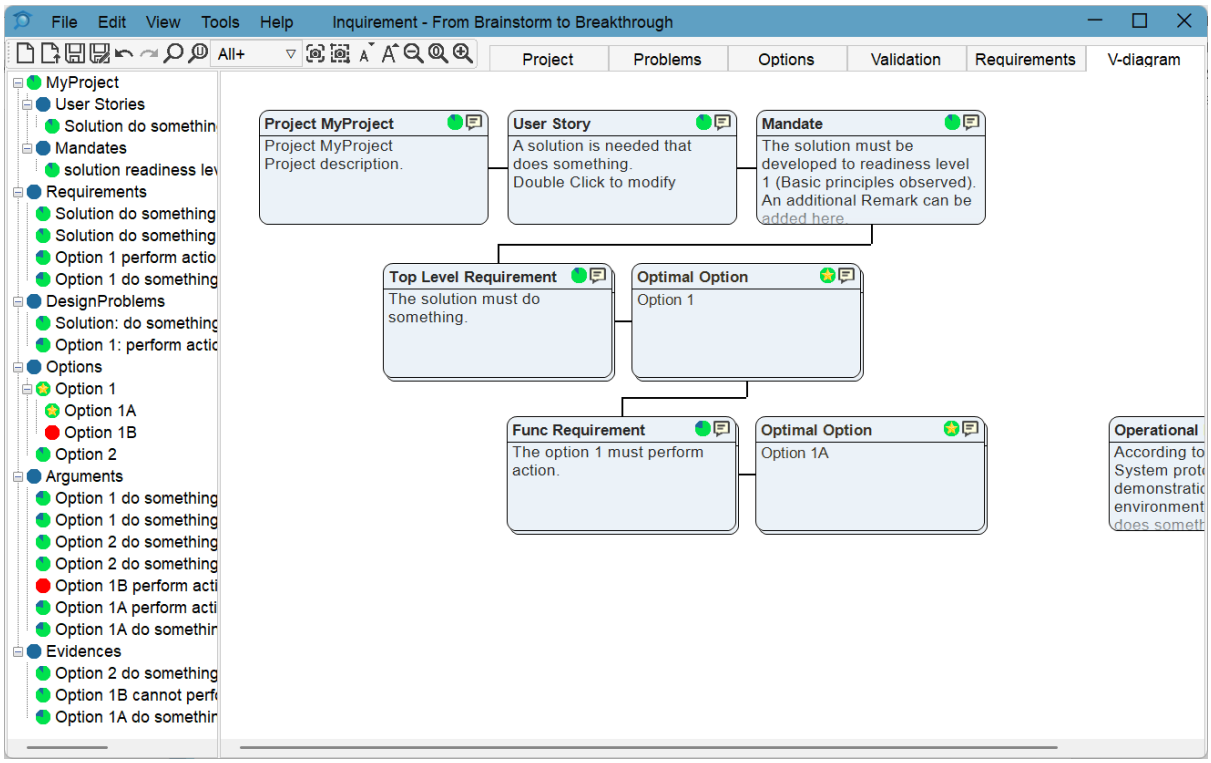
Each Requirement has a reason why it exists. This reason can be shown by using Right-Mouse-Button → Show Why in a Requirement Object.

Integration Requirements originate from Option Objects (The Option must be integrated into the higher level Option) instead from Argument Objects. Therefore the Option Object is shown as parent of an Integration Requirement instead of an Argument Object.

3.2.6 V-diagram tab page

The V-diagram tab page is intended for the Select the best Solution process step to identify and visualize all preferred Option Objects from their alternatives.

This tab page shows preferred Option Objects, Requirement Objects and Evidence Objects. Unless otherwise specified, each row displays the optimal Option Objects.



The V-diagram shows the Project tab page objects in the upper row. The rows below show the Requirement Objects and the Option Objects into increasingly detailed levels. Evidence Objects are visualized in the right branch of the diagram.

Focus on a part of the V-diagram by selecting an object in the Tree View (or Right-Mouse-Button → Select object in tree on the object in the Workspace). The Workspace will show the V-diagram below the selected object only.

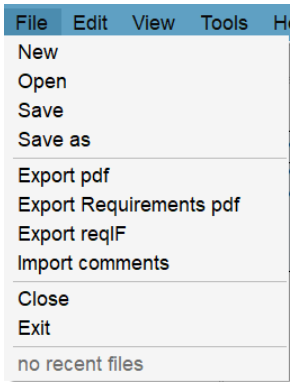
Requirement Objects are displayed behind each other. Use Right-Mouse-Button → Hide or Show Nodes in the Requirement Object to hide or show each Requirement Object in that row.

Optimal Option Objects are shown by default. Use Right-Mouse-Button → 'Hide or Show Nodes' to manually select another Option Object. The function File → Export Requirements pdf will export the requirements that appear in the V-diagram. It will omit Requirements that belong to Options that are not selected in this tab-page.

4 Menu Bar

4.1 File Menu

The first menu is the File menu.

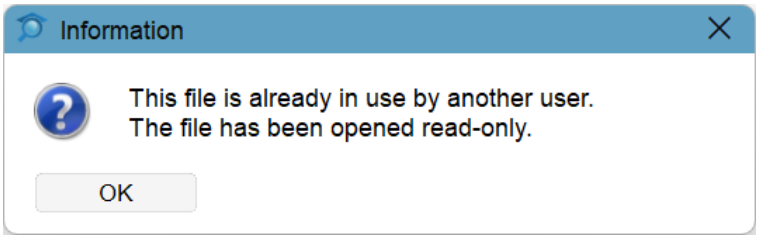


4.1.1 New

Use this function to create a new Project. You will be asked whether to first save the current Project first.

4.1.2 Open

Use this function to open a saved Project. A Project is saved as an xml file. See Chapter 8 for information about Inquirement files. You will be asked to first save the current Project file before opening a new one. A warning is displayed if you try to open a locked file.



4.1.3 Save

Use this function to save the current Project. A Project is saved as an xml file. See Chapter 8 for information about Inquirement files. The function appears grey if no license is available. Please Renew License to restore the Save function.

4.1.4 Save As

Use this function to save the current Project in a new file location or using a different name. A Project is saved as an xml file. See Chapter 8 for information about Inquirement files. The function appears grey if no license is available. Please Renew License to restore the save function.

4.1.5 Export pdf

Use this function to export all information of the Project to a pdf file. The pdf file contains all information of the Project, including Options that were not selected, comments (see §7.3.1) and any tooltip texts (see §7.3.13).

The pdf can be helpful for long term storage, data sharing or, for instance, registration of ideas.

4.1.6 Export Requirements pdf

Use this function to export the Requirements from the selected Options in the V-diagram tab page to a pdf. The pdf contains a subset of the information exported when using the Export pdf function discussed in §4.1.5. Use 'Hide or Show Nodes' to select other Option Objects and export another subset of the Requirements.

The pdf file can be helpful when a choice has been made between all possible Options. The Requirements and Options in the pdf file offer a readable input when migrating to a tool to manage the lifecycle of the Requirements and Options. For direct export and import between tools the reqIF file format is recommended (see §4.1.7).

4.1.7 Export ReqIF

Use this function to export all information of the Project into a ReqIF file. Almost all Requirements Management and SysML modelling tools today support ReqIF import and export.

Since the Inquirement process focusses on the initial creation of Requirements and Options, there currently is no function to import ReqIF files.

4.1.8 Export SysML

Use this function to export all information of the Project into a SysML file. Almost all Requirements Management support SysML import and export.

Since the Inquirement process focusses on the initial creation of Requirements and Options, there currently is no function to import SysML files.

The exporter obtains the current root Project from GCC::RP and creates an XMI 2.5.1 / UML 2.5.1 model with the SysML 1.6 profile applied. The exported model is divided into three UML packages:

- Inquirement Model: the semantic Inquirement objects
- Inquirement Relations: the semantic relations between these objects
- Inquirement Views: vendor-neutral representations of the Inquirement views

4.1.9 Import comments

Use this function to import comments from a similar Project file into the current Project. Comments can be added to each of the Inquirement objects. Other users can then Show and Edit comments to the Project in their local file. These comments can be imported from another file into the original Project file.

This function basically checks for comments in each of the Inquirement objects in the selected file. It copies the comments to the object with the same Unique Identifier.

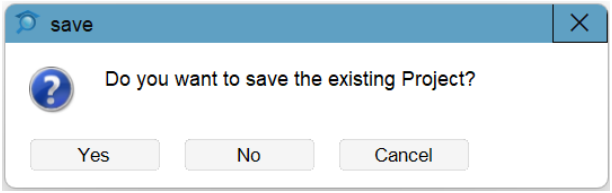
The process is as follows:

1. Create a Project file.
2. Share the Project file with other people.
3. Other people can add comments to all Inquirement objects and save the file.
4. Select 'File Menu' → 'Import comments' in the menu bar.
5. Select the file received from another user.
6. The comments added by the other user are now added in your original file.

The 'Edit Menu' => 'Find all comments' function can be used to inspect all objects that have a comment.

4.1.10 Close

Use this function to close the current Project file. You will be asked whether to first save the current file if there are unsaved items in the current Project file.



A new Project will be created once the Project is closed.

4.1.11 Exit

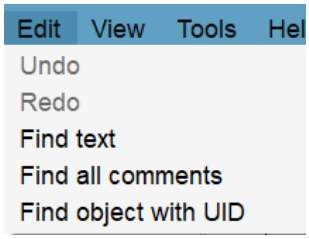
Use this function to close the Inquirement application. You will be asked whether to first save the current file if there are unsaved items in the current Project file.

4.1.12 Recent files

Recent files can be found here. Click on any file to directly open it.

4.2 Edit Menu

The second menu is the Edit menu.



4.2.1 Undo

Use this function to Undo the last action. There is nothing to Undo if Undo is grey.

4.2.2 Redo

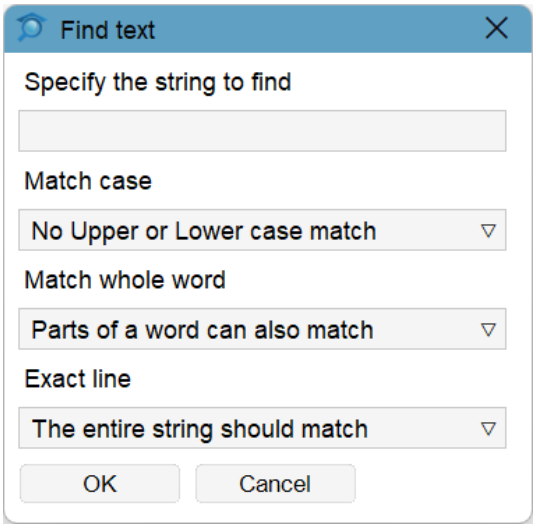
Use this function to Redo the last action. There is nothing to Redo if Redo is grey.

4.2.3 Find all comments

Use this function to find all objects that have a comment (see also Show and Edit comments). This function can be used when the 'Import comments' function has been used.

4.2.4 Find Text

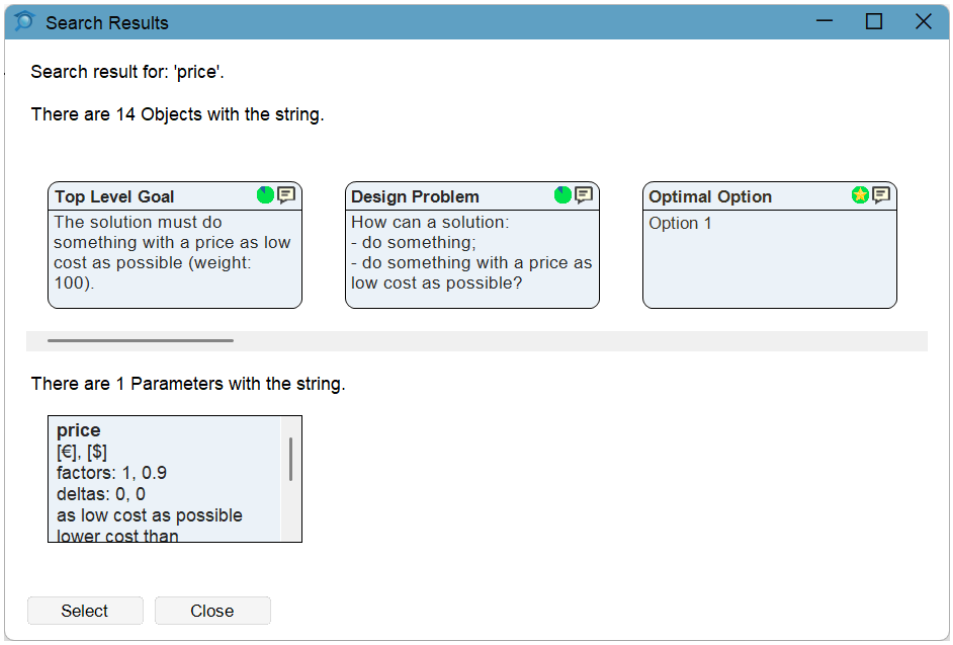
Use this function to find Objects or Parameters with a certain text.



The menu above shows the following options for the text search:

Match case	'No Upper or Lower case match' or 'Upper or Lower case must match'
Match whole word	'Parts of a word can also match' or 'String must match whole word'
Exact line	'The entire string should match' or 'At least one word should match'

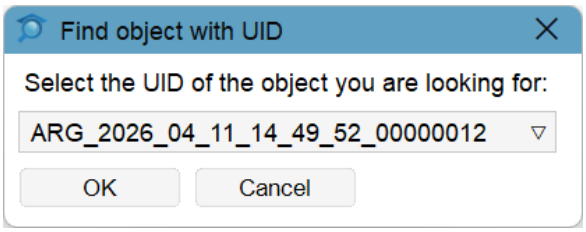
Press 'OK' to search. The search result shows all Objects and Parameters with a certain text.



Press 'Select' to Select object in tree for each object or press 'Close' to close the window.

4.2.5 Find Object With UID

Use this function to find an object using its Unique Identifier (UID). This function displays a window with a pull-down containing all Unique Identifiers in the Project.

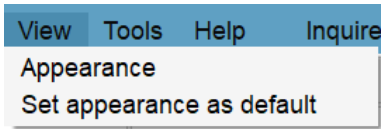


Find and select the Unique Identifier you are looking for and press ‘OK’. This will show the object with the Unique Identifier and its related objects in a new window like in ‘Show related objects’.

The UID can also be pasted into the pull down menu if the UID is on the clipboard.

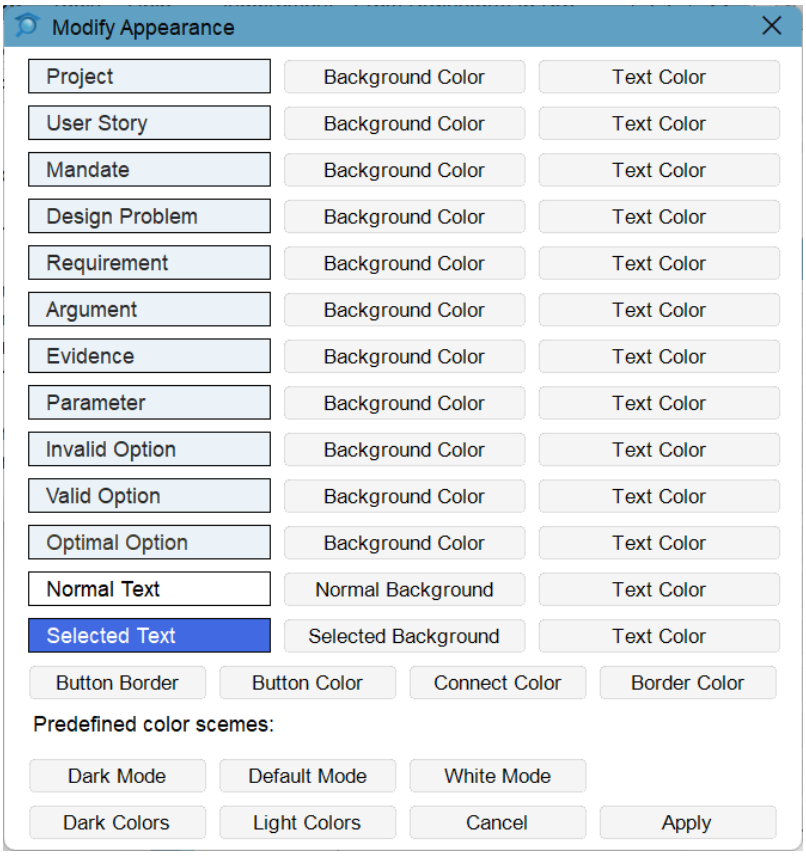
4.3 View Menu

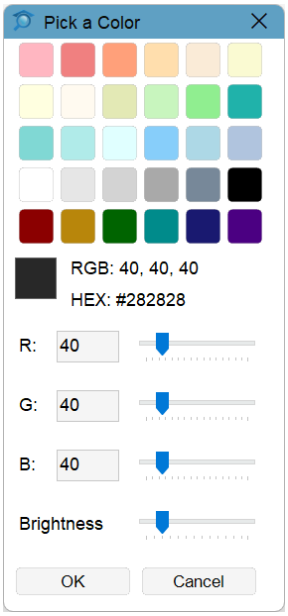
The third menu is the View menu.



4.3.1 Appearance

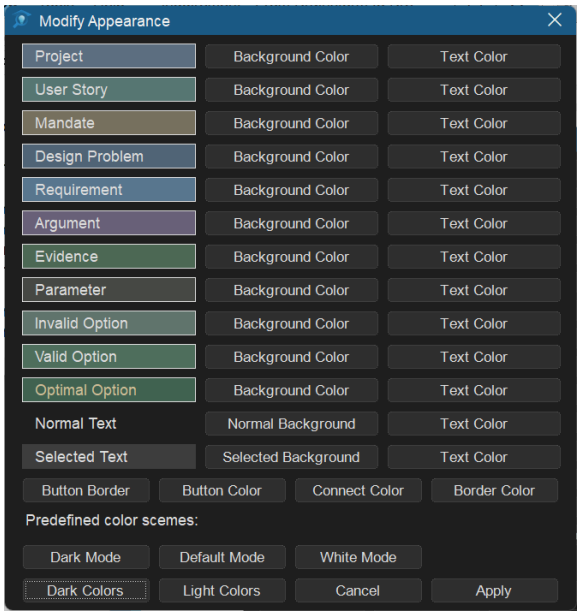
Use this function to change the appearance of the Inquirement Application. Clicking the function will show the window below.





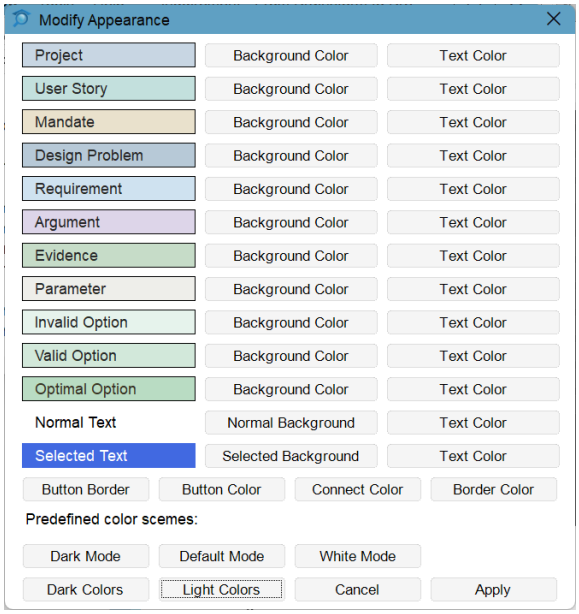
The buttons can be used to change the colors of the items. Objects, text and button apearance can be changed. Clicking a button shows a color picker window as show on the right above, allowing direct selection of pre-defined colors or input of Red Green and Blue in the textboxes.

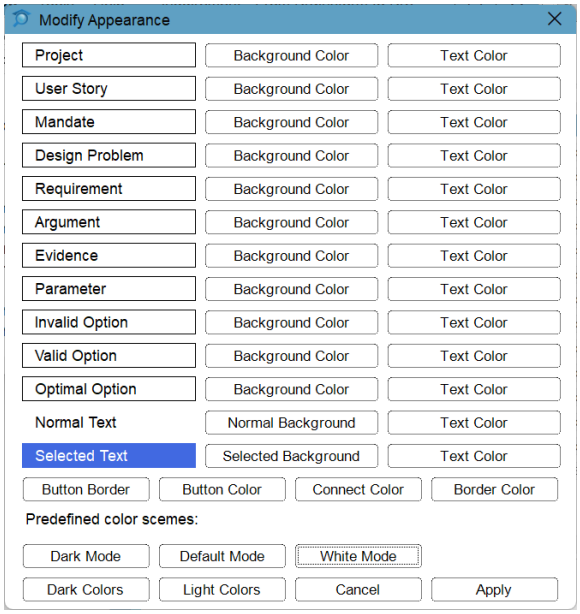
Several predefined color scemes can be selected below in the Appearance window.



Dark Mode (left) and Dark Colors (right, each object has a unique color)

Additionally, an alternative light mode with different colors for each object can be selected. There is also a White Mode that can be used for, for instance, printing.





Light Colors (left, each object has a unique color) and White Mode (right)

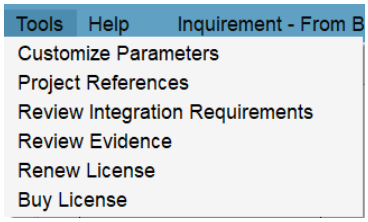
Selecting Apply will set the defined colors to the Inquirement interface and also store the color setting as default for your Inquirement application in the future.

4.3.2 Set appearance as default

Use this function to set the current colors as default for your Inquirement application in the future. This can, for instance, be done when a Project file with color settings that you like is opened.

4.4 Tools Menu

The fourth menu is the Tools menu.



4.4.1 Customize Parameters

Use this function to show a Window that shows all Parameters. Parameters can also be added or modified in this Window.

Customize Parameters

Acti...	Type	Unit Symbol	Conversion Factor	Delta	Target	Action
					at most as bright as	
					equal to	
					at least as bright as	
					brighter than	
					as bright as possible	
		<new unit symbol>	<new unit factor>	<new unit delta>		add unit
☑	velocity	m/s	1	0	as slow as possible	
☑		km/s	1000	0	slower than	
☑		km/h	0.2778	0	at most as fast as	
					equal to	
					at least as fast as	
					faster than	
					as fast as possible	
		<new unit symbol>	<new unit factor>	<new unit delta>		add unit
☑	force	N	1	0	as low as possible	
					lower than	
					at mosts	
					equal to	
					at least	
					larger than	
					as large as possible	
		<new unit symbol>	<new unit factor>	<new unit delta>		add unit
	<new type name>	<new base unit symbol>	1	0	<as small as possible>	
					<lower than>	
					<lower than or equal>	
					<equal to>	
					<larger than or equal>	
					<larger than>	
					<as large as possible>	add type

Inquirement has a standard set of Parameters that cannot be deleted: function, real, length, angle, time, mass, price, temperature, amount of substance, luminous intensity, velocity and force.

The Parameter's activity can be (de)selected. Deselecting does not delete the Parameter, but will prevent the Parameters from showing up in the pull down menu in for instance Change the Quantification. Parameters that are in use in an object cannot be deactivated.

Parameters can be represented in different units. Each unit has a 'symbol' and a 'conversion factor' and a 'delta'. The 'conversion factor' and 'delta' are used to re-calculate the value if the unit is changed. (e.g. ºC = 5/9 x ºF + 17.778, where the 'conversion factor' is 5/9 and the 'delta' is 17.778

Units can be added to each Parameter by clicking the 'add unit' button. Make sure the symbol, factor and delta are properly defined before clicking 'add unit'.

New types can be added as well using the 'add type' button. Check that all fields are defined before pressing this button.

4.4.2 Project References

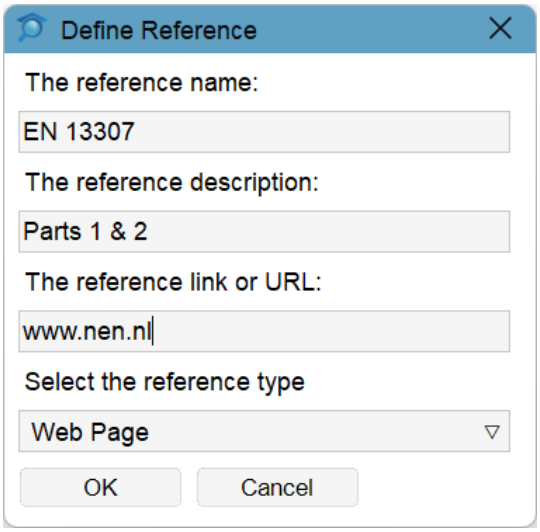
Use this function to show a Window that shows all References that have been defined in the Project. Each Reference can be modified. References can also be added.

Project References

ID	Name	Description	Link	Type
1	EN 13307	Parts 1 & 2	https://www.nen.nl/nen-en-13307-1-2	Web Page
2	EN 356	Class P6B	https://www.tuv.com/netherlands/nl/te	Web Page
3	Calculation 3	FE Analysis	P:\Programs\Gate\UseCase3.analysis	Document
	Click here to add a new Reference			

The Reference will be opened when a cell is clicked that contains the link.

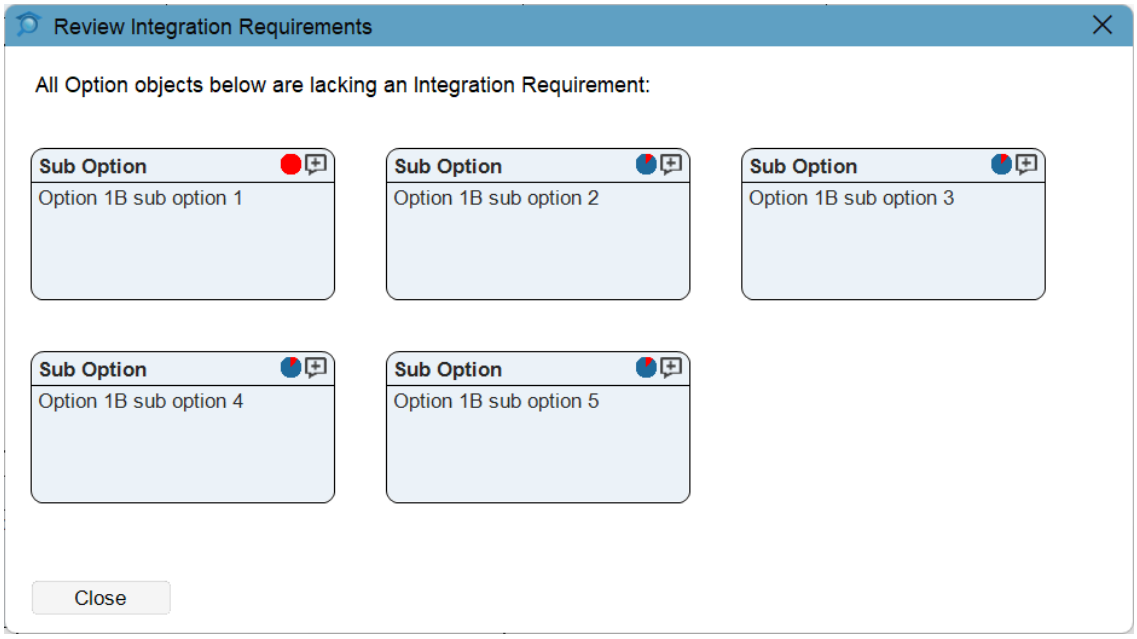
A Window pops up when a Reference is clicked. The Reference can be edited by changing the values in this Window.



A new Reference can be created by clicking on the cell with ‘click here to add a new Reference’. The same Window above pops up to define the new Reference. The new Reference is created by clicking ‘OK’ after filling the fields.

4.4.3 Review Integration Requirements

Use this function to check for any Options that are lacking an Integration Requirement. Integration Requirements can be lacking because the automatically created Integration Requirements can be deleted.



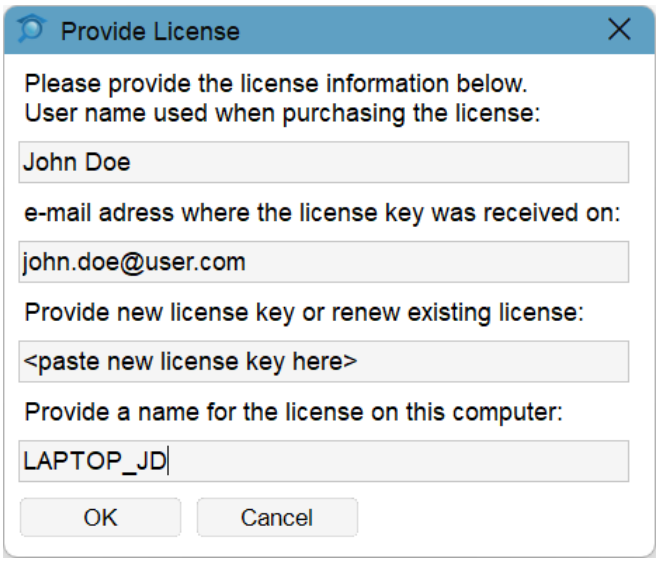
This function allows to justify deleted Integration Requirements. Use the ‘Add Integration Requirement’ context menu function to correct unjustly deleted Integration Requirements.

4.4.4 Review Evidence

Use this function to check the Evidence that is supporting the Project. See §7.6.4 ‘Mandate Object’ -> ‘Review Evidence’.

4.4.5 Renew License

Use this function to update the license information. This window will pop up when the license has expired or when there is no active license yet.



Please provide the user name used when purchasing the license as well as the e-mail address where the license was received on. The license key itself must be pasted or typed in the license key textbox.

You can install Inquirement on as many computers as you want. The license has a name on each computer. This allows deactivation of licenses on certain computers if needed. Go to www.lemonsqueezy.com or follow the link in the license activation e-mail to view the order. Click on the License to go to the activation or deactivation page.

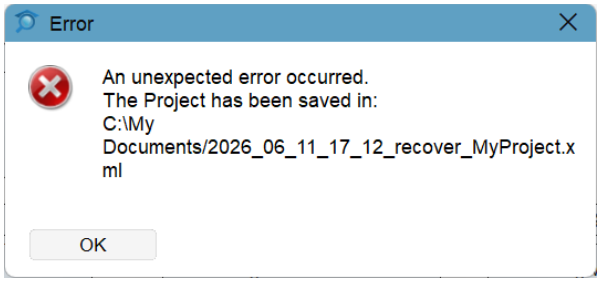
Click 'OK' when the information is provided. The application will try to contact the license provider and grant access to the application if successful.

Only during this process Inquirement needs connection with the internet. Inquirement will try to validate the license every 13 days (see also §12).

Saving a Project will not work without a license. All other options and functions will still work without a license, making it possible to open and investigate Inquirement without a license. The example Projects can be used for this (see also §4.5.3 Example Projects).

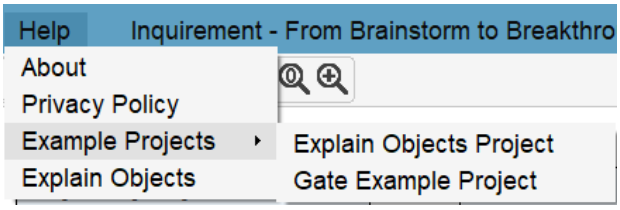
4.4.6 Buy License

The purchase page will be opened in a browser when this button is pressed. Having a license also enables recovery files to be saved if Inquirement crashes.



4.5 Help Menu

The fifth menu is the Help menu



4.5.1 About

Use this function to view the ‘about page’.

4.5.2 Privacy Policy

Use this function to view the ‘privacy policy’. See also Chapter 12.

4.5.3 Example Projects

Having an example makes it easier to play around with the features and possibilities of Inquirement. Use the example projects below to make yourself familiar with the application.

4.5.3.1 Explain Objects Project

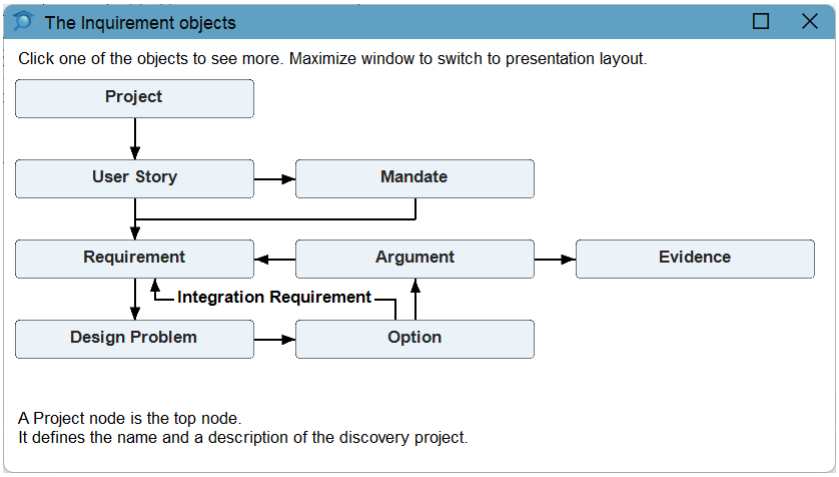
Use this function to create an example Project that shows all Inquirement objects.

4.5.3.2 Gate Example Project

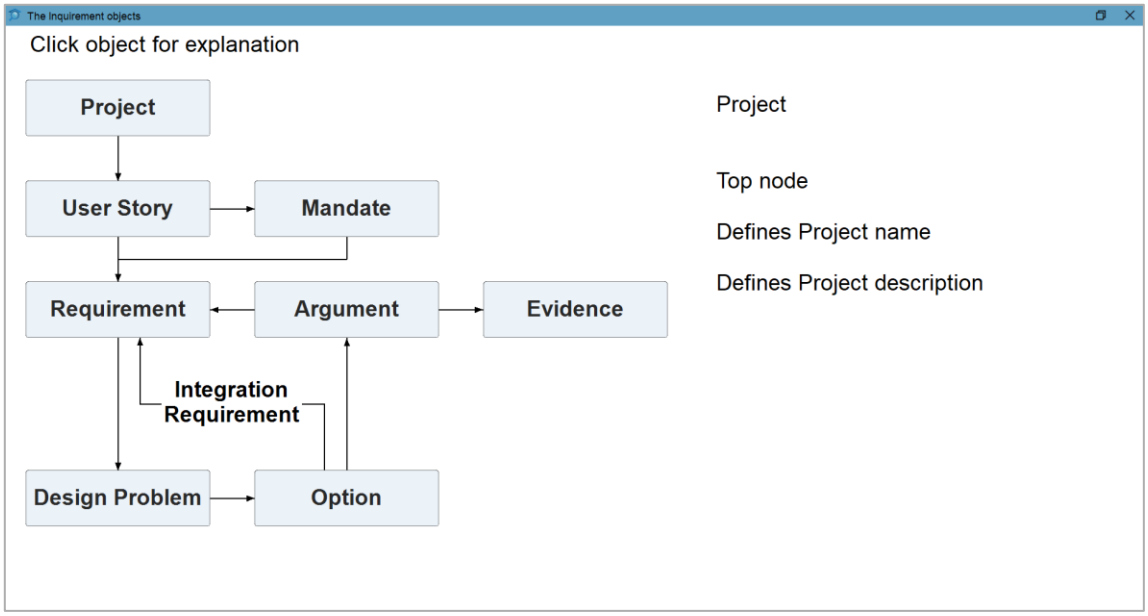
Use this function to create an example Project that shows possible solutions for keeping certain people out and letting certain people through a gate.

4.5.4 Explain Objects

Use this function to show a window that explains all Inquirement objects.



Clicking one of the objects will display further explanation of that object. Clicking the maximize button in the top right will show a full screen version of the window in presentation format.



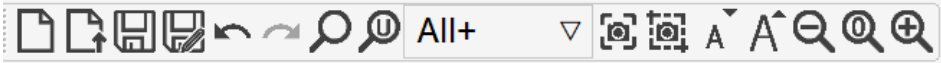
4.5.5 Keyboard Shortcuts or Hotkeys

The following keyboard shortcuts or hotkeys are available:

CTRL + Z: Undo the last action.
CTRL + Y: Redo the last action.
CTRL + S: Save the current Project (only if there is a valid license).
CTRL + F: Access the Find Text function.
CTRL + O: Open a Project.
CTRL + N: Create a New Project.

5 Toolbar

You can find the toolbar in the left upper corner of the Inquirement window.



5.1 New

The button has the same function as the 'New' File Menu item (see §4.1.1).

5.2 Open

The button has the same function as the 'Open' File Menu item (see §4.1.2).

5.3 Save

The button has the same function as the 'Save' File Menu item (see §4.1.3). The button appears grey if no license is available. Please Renew License to restore the save function.

5.4 Save As

The button has the same function as the 'Save As' File Menu item (see §4.1.4). The button appears grey if no license is available. Please Renew License to restore the save function.

5.5 Undo

The button has the same function as the 'Undo' Edit Menu item (see §4.2.1).

5.6 Redo

The button has the same function as the 'Redo' Edit Menu item (see §4.2.2).

5.7 Find

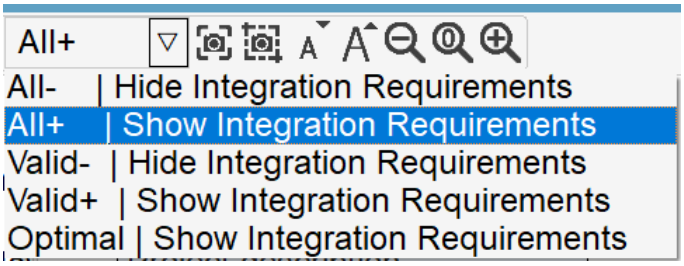
The button has the same function as the ’Find Text’ Edit Menu item (see §4.2.4).

5.8 Find object with UID

The button has the same function as the ’Find Object With UID’ Edit Menu item (see §4.2.5).

5.9 Object Filter

This dropdown menu can be used to control which objects are shown or hidden.



Choices include hiding or showing invalid, valid or optimal Integration Requirements

All-	Hide Integration Requirements	Show all objects
All+	Show Integration Requirements	Show all objects
Valid-	Hide Integration Requirements	Hide invalid objects
Valid+	Show Integration Requirements	Hide invalid objects
Optimal	Show Integration Requirements	Only show the optimal objects

5.10 Image of Current Tab Page

Use this button to take an image of the current entire tab page (also the part that is out of view in the scroll area).

An image is stored on the clipboard if CTRL is pressed when the button is clicked. In this case the png image has the same resolution as the current tab page. It can be directly pasted into any document without first saving the image. Use functions §5.12 to §5.16 to change the size manually if a higher resolution is required for the image on the clipboard.

Detail of 500% zoom (left) versus 100% zoom (right)

A save menu will appear when the button is pressed. The following image formats can be chosen:

PNG Image (*.png)	Normal PNG	Current zoom level
High Quality PNG (*.png)	High-quality PNG	Zoomed level 500%
JPEG Image (*.jpg)	Normal JPEG	Current zoom level
High Quality JPEG (*.jpg)	High-quality JPEG	Zoomed level 500%
Bitmap Image (*.bmp)	BMP	Current zoom level
High Quality BMP (*.bmp)	BMP	Zoomed level 500%
PDF on A4 (*.pdf)	PDF on A4 paper	Zoomed level 500%
PDF custom size (*.pdf)	PDF on custom paper	Zoomed level 500%

Select the folder, provide the image name and select the image format. A follow up window will appear to select the paper size if the ‘PDF on custom paper’ is selected.

Creation of the image relies on the actual screen rendering. This avoids less accurate background rendering, but will show some flickering on the screen.

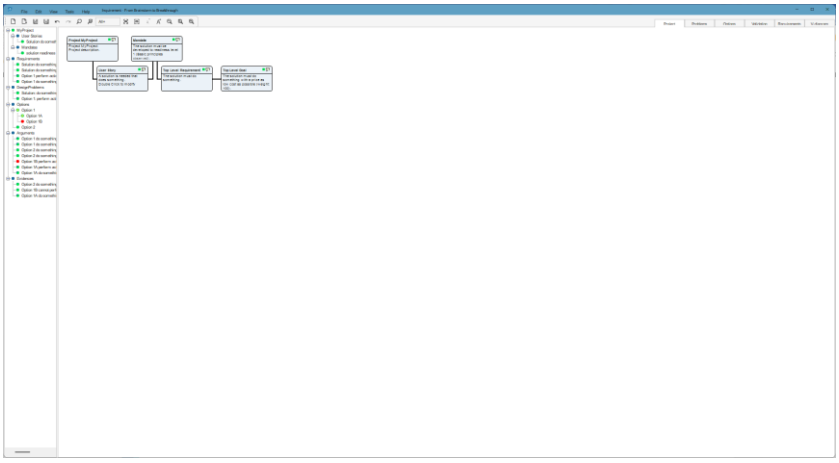
5.11 Take Screen Shot Image

Use this button to take an image of the visible Workspace. Clicking the button dims the screen and shows crosshairs. Press the Left-Mouse-Button to define a corner of the intended image area. Then move the mouse to the second corner of the intended image area. Then let go of the Left-Mouse-Button to finish the selection of the image area.

An image is stored on the clipboard if CTRL is pressed when the Screen Shot Button is clicked. Otherwise, a save menu will appear. Select the folder, provide the image name and select the image format (PNG, JPEG or BMP).

5.12 Decrease Font Size

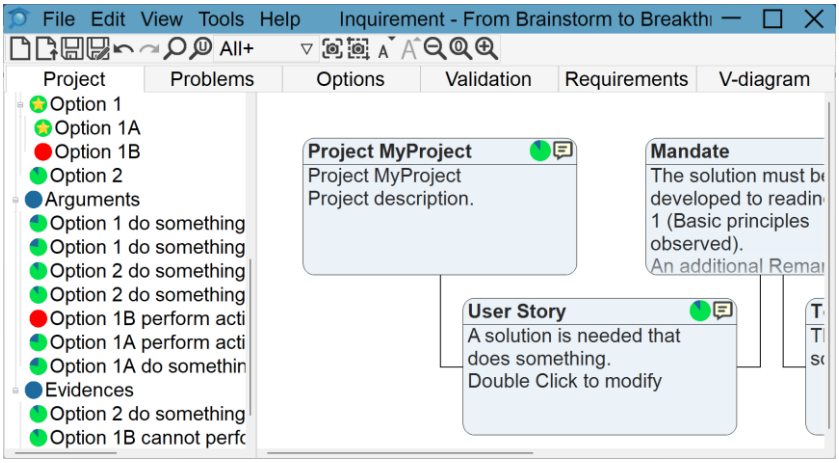
Use this button to decrease the font size of the application. All objects, menus and buttons are scaled with the font size. The image below shows the Inquirement window with minimal size font.



The button can be used to adapt the application appearance to either working on a personal screen or running it on a large screen in a meeting room. Use the Increase Font Size button to increase the font size again if needed. Use the ‘Zoom In’ or ‘Zoom Out’ button to only scale the current Workspace tab page.

5.13 Increase Font Size

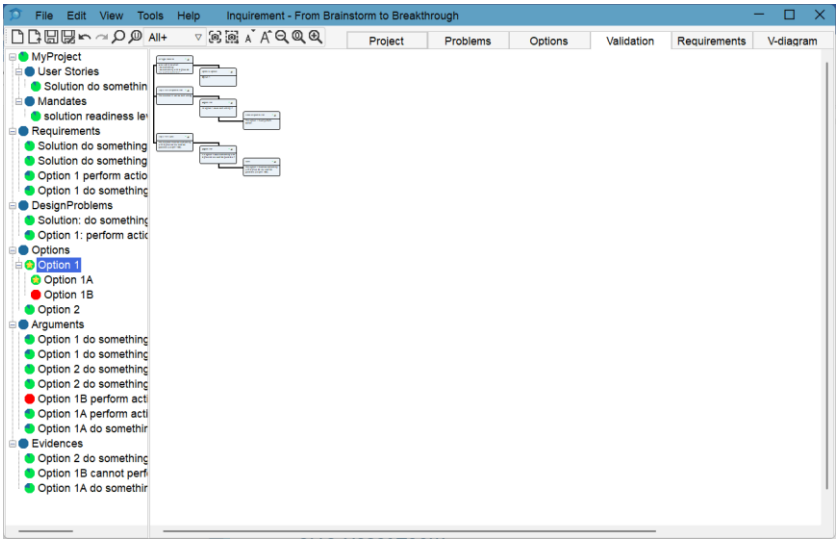
Use this button to increase the font size of the application. All objects, menus and buttons are scaled with the font size. The image below shows the Inquirement window with maximal size font.



The button can be used to adapt the application appearance to either working on a personal screen or running it on a large screen in a meeting room. Use the Decrease Font Size button to decrease the font size again if needed. Use the ‘Zoom In’ or ‘Zoom Out’ button to only scale the current Workspace tab page.

5.14 Zoom Out

Use this button to zoom out on the current Workspace tab page only. The zoom level is stored and switching to another tab page will apply the zoom level stored for that tab page. The image below shows the Inquirement window with minimal zoom.



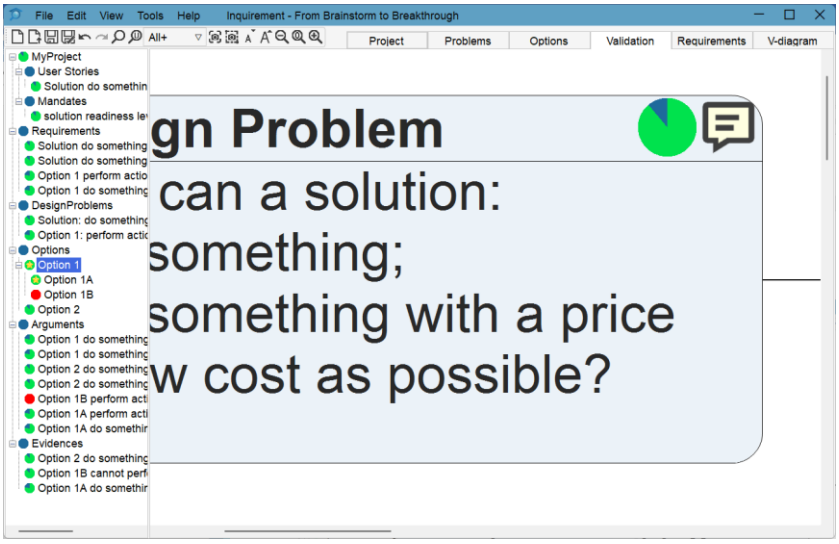
This button will not scale all objects, menus and buttons. Use the ‘Decrease Font Size’ or ’Increase Font Size’ button to scale everything. This Zoom Out button can also be used to modify the resolution before pressing the ‘Image of Current Tab Page’ button. Zooming can also be achieved by pressing CTRL while moving the Mouse Wheel.

5.15 Reset Zoom

Pressing this button will instantly set the zoom level of the current Workspace tab page to 100%.

5.16 Zoom In

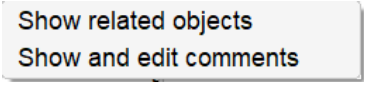
Use this button to zoom in on the current Workspace tab page only. The zoom level is stored and switching to another tab page will apply the zoom level stored for that tab page. The image below shows the Inquirement window at maximal zoom.



This button will not scale all objects, menus and buttons. Use the ‘Decrease Font Size’ or ’Increase Font Size’ button to scale everything. This Zoom In button can also be used to modify the resolution before pressing the ‘Image of Current Tab Page’ button. Zooming can also be achieved by pressing CTRL while moving the Mouse Wheel.

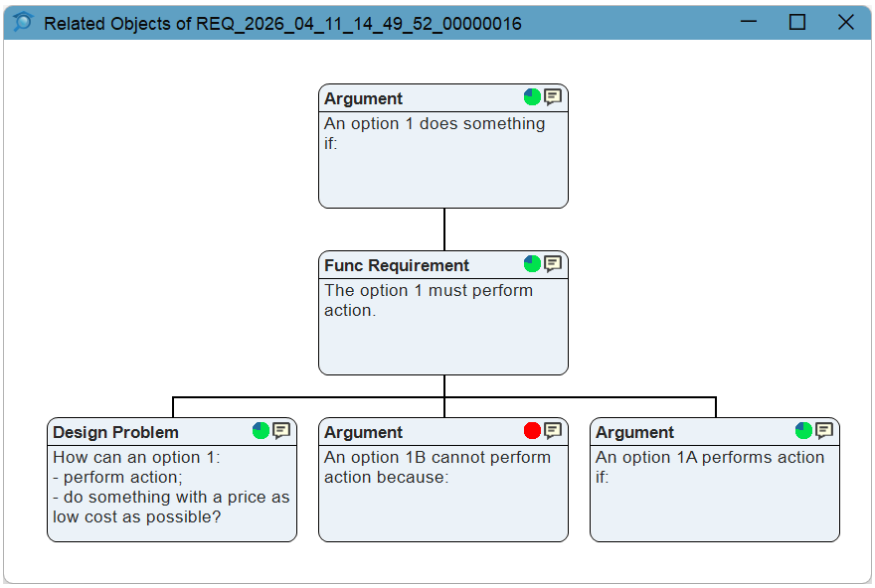
6 Treeview Menu

Pressing Right-Mouse-Button on an item in the Tree View will display the menu shown below.



6.1 Show related objects

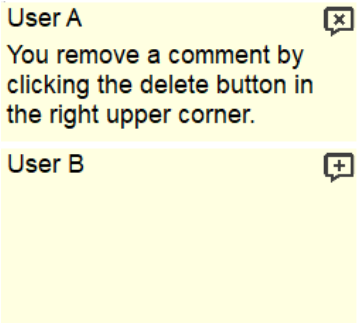
This Treeview Menu item will visualize the related objects of the object as shown below.



See also §7.3.14 where the same function can be called from a Workspace object.

6.2 Show and edit comments

This Treeview Menu item will display the comments that have been added to the object as shown below.



See also §7.3.1 where the same function can be called from a Workspace object. The ‘Edit Menu’ => ‘Find all comments’ function can be used to inspect all objects that have a comment.

7 Objects

7.1 Parameter

Parameters are the objects that can be used to quantify the Requirements or Evidence. Inquirement has a built-in set of Parameters.

Activity	Type	Unit Symbol	Conversion Factor	Delta	Target Text
☑	temperature	C	1	0	as cold as possible
☑		K	1	273.15	colder than
☑		F	0.5556	17.778	at most
					equal to
					at least
					hotter than
					as hot as possible

As shown in the example table above, each Parameter has:

Activity	Whether the Parameter shows up in, for instance, the 'Change the Quantification' pull down menu. Parameters that are in use cannot be deactivated or deleted.
Type	Description of the Parameter type.
Unit Symbol	The symbol for each unit. The first row is the base unit.
Conversion Factor	The factor to convert the unit to the base unit. For instance: ºC = 5/9 x ºF + 17.778, where 5/9 is the conversion factor. The factor is always 1 for the base unit.
Delta	A number that has to be added or subtracted to convert the unit to the base unit. For instance: ºC = 5/9 x ºF + 17.778, where 17.778 is the delta. The factor is always 0 for the base unit.
Target Text	Seven descriptions that correspond to the mathematical equivalent: 'minimal', '<', '<=', '=', '>=', '>', 'maximum'.

The 'Tools Menu' 'Customize Parameters' function can be used to (de)activate, create or modify Parameters.

7.2 Reference

References are the objects that point to a webpage or file. A Reference can be added to all Objects.

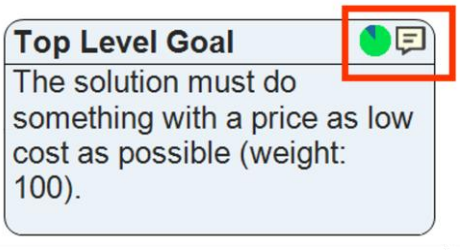
Each Reference has:

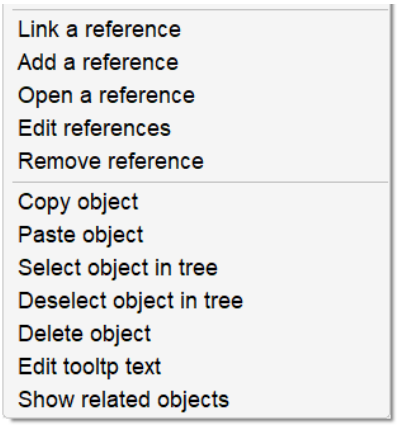
ID	Each Reference receives a unique identifyer number (1, 2, 3, …).
Name	The name of the Reference.
Description	A description of the Reference.
link or URL	Link to a file or URL to a website.
Type	The type of Reference (can be Report, Documents, Web-page, Presentation, Image, Video, 2D CAD file, 3D CAD file, ReqIF file or File).

The 'Tools Menu' 'Project References' function can be used to view or modify References.

7.3 Generic context menu

The following sections discuss the Inquirement object representations and the functions that are available when the Right-Mouse-Button is pressed. The sections in this paragraph discuss the context menu functions that every object has. The meaning and functionality of the two icons in the right upper corner of each object will also be explained.





7.3.1 Show and Edit comments

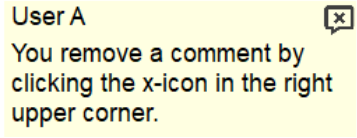
Clicking this icon will show the comments.

The icon has several appearances:

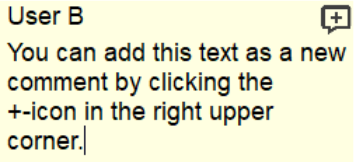
No comments are available. Add a comment by clicking this icon

Comments have been added. Click this icon to show, modify or add comments.

In the comment itself there are the following icons:



Click the x-icon to delete the comment.

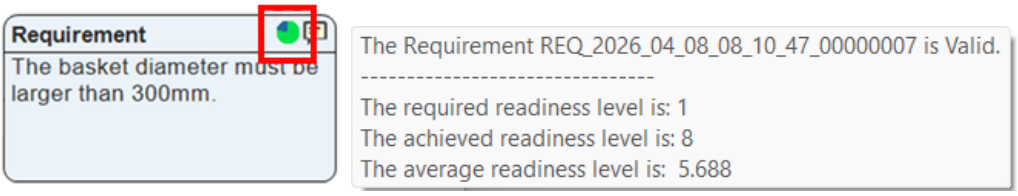


Click the +-icon to add the comment

See also §6.2 where the same function can be called from a Treeview Menu. The ‘Edit Menu’ => ‘Find all comments’ function can be used to inspect all objects that have a comment.

7.3.2 Readiness Level Icon

An icon shows the achieved Readiness Level. Hovering over the icon will show the Readiness Level of the object (see lower right picture). An Option Object will also show the sum of the Parameter values in the Option Readiness Level Icon.



The icon is hidden for Objects that have no children, except for Evidence Objects, because they are supposed to be endpoints. This increases the distinction between objects that have been processed and objects that still need attention.

Examples of the Readiness Level icons meaning is shown below

Optimal solution (valid Option with best score for goals)
Achieved Readiness Level >= Required Readiness Level

Required Readiness Level is 2/9 (indicated by the red pie area)
Achieved Readiness Level is 1/9 (indicated by the green pie area)

Achieved Readiness Level is 5/9 (indicated by the green pie area)
Required Readiness Level is lower than 5/9 (red pie area hidden)

Required Readiness Level is 9/9 and 0/9 Achieved, or
The Option is proven to be invalid (Evidence exist)

7.3.3 Select object in tree

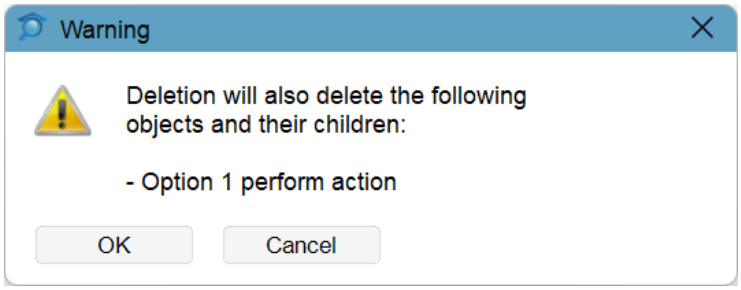
Selecting this context menu item will select the object in the Tree View. A selected object in the Tree View will filter the objects shown in the Workspace tab pages. The Workspace will show objects below the selected object only.

7.3.4 Deselect object in tree

Select this context menu item to de-select the object in the Tree View. This will impact the objects shown in the Workspace tab pages as well.

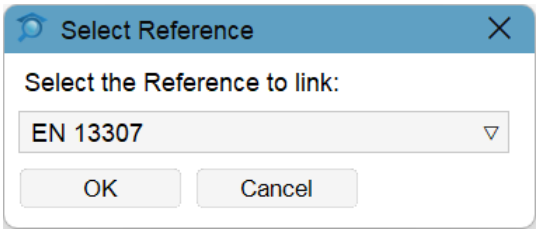
7.3.5 Delete object

All objects, except Project objects can be deleted. All child objects that have this object as single parent will be deleted too. A warning is shown if deletion of the object will cause deletion of more objects as well.



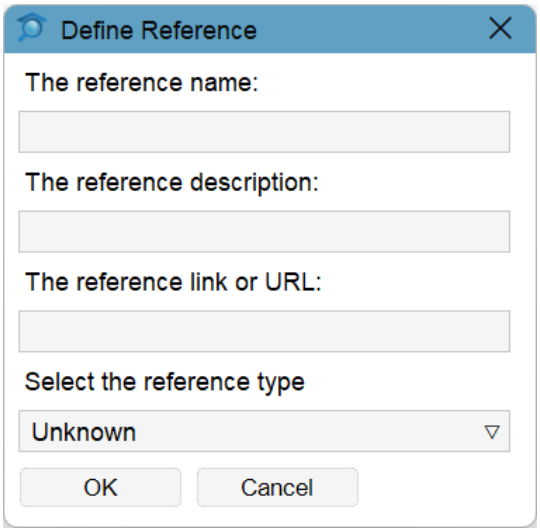
7.3.6 Link a Reference

Selecting this context menu item will link one of the original References from the 'Tools Menu' 'Project References' to be linked to this object. A Window will be shown to select which Reference to link.



7.3.7 Add a Reference

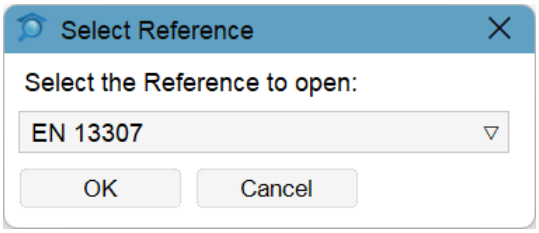
Selecting this context menu item will show a Window where a new Reference can be defined and linked to this object.



The new Reference will be added to the 'Tools Menu' 'Project References' as well.

7.3.8 Open a Reference

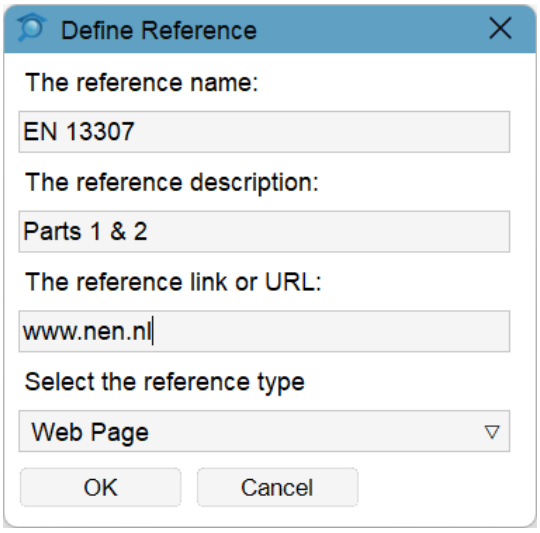
This context menu item is only visible if the object has at least one Reference. Selecting this context menu item will either open the Reference directly or ask which Reference to open in case the object has multiple linked References.



An error message appears if Windows is unable to open the file. The link or URL can be retrieved by using the 'Edit References' context menu and copy the link or URL for further investigation in the file system or internet browser.

7.3.9 Edit References

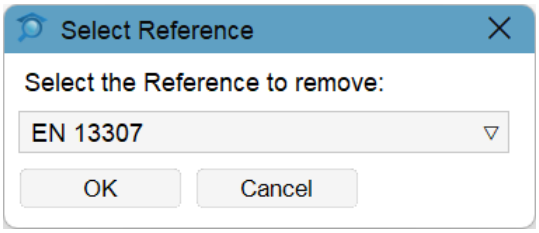
This context menu item is only visible if the object has at least one Reference. Selecting this context menu item will first show a Window where a Reference can be selected. Then the Window below appears that allows to edit the Reference. The Window will appear immediately if only one Reference is linked to the current object.



The Reference will be changed when 'OK' is clicked. Keep in mind that this modifies the Reference itself. All other objects that link to the same Reference will automatically see the modified Reference as well.

7.3.10 Remove Reference

This context menu item is only visible if the object has at least one Reference. Selecting this context menu item will remove a Reference from this object. A menu pops up to select the Reference to remove if this object refers to multiple References.



The Reference is removed immediately if the object has only one Reference. There is no need to select the one to be removed.

The original Reference from the 'Tools Menu' 'Project References' is deleted as well if this is the single object referring to the specific reference.

7.3.11 Copy object

Selecting this context menu item will copy the object Unique Identifier (UID) to the clipboard. This UID on the clipboard can later be used to paste a copy of the object (see below). The copied UID can also be pasted as normal text.

7.3.12 Paste object

Selecting this context menu item will retrieve a Unique Identifier (UID) from the clipboard (if any) and uses the corresponding object as the source for the paste operation.

If the clipboard node has the same type as this object, its values are copied into this object. The exception is when the clipboard object is a Sub-Option Object and this object is an Option Object; in that case, a new Sub-Option Object is created instead.

If the clipboard object values cannot be copied into this node, the function attempts to create it as a new child. Supported relationships include:

1. in a Project Object a User Story Object can be created
2. in a User Story Object a Requirement Object can be created
3. in a Requirement Object a Design Problem Object can be created
4. in a Design Problem Object an Option Object can be created
5. in an Option Object a Sub-Option Object can be created
6. in an Argument Object a Requirement Object can be created
7. in an Argument Object an Evidence Object can be created

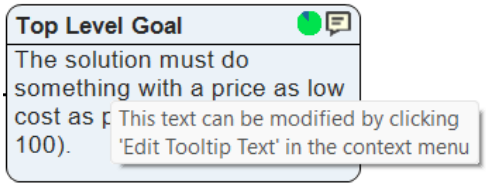
A Project Object or Mandate Object can never be copy-pasted, since only one exists in a Project. Argument Objects cannot be copy-pasted as well, since Argument Objects typically have two parents (a Requirement and an Option) and pasting only can define one of them as a parent.

When a new child is created, the values of the clipboard object are copied to the new child object. Any children of the clipboard object are then recursively copied to the new object, provided that the child only has parents that are known within the pasted tree.

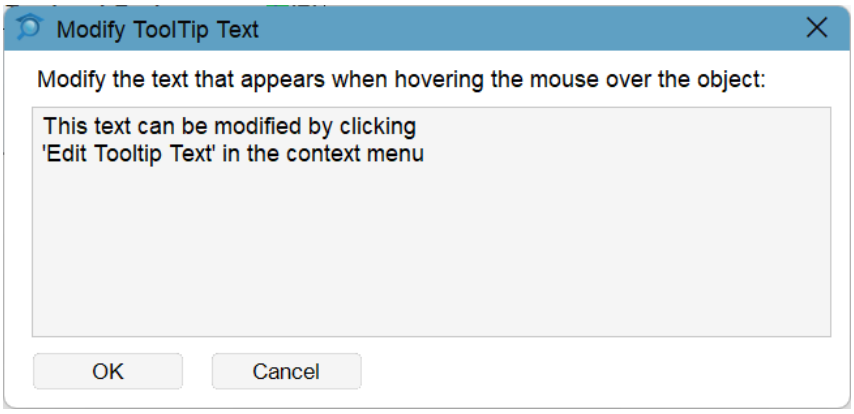
The complete paste operation, including all newly created Objects and affected existing Objects, is added to the history as a single state for later Undo or Redo.

7.3.13 Edit tooltip text

A tooltip text will appear when the mouse hovers over the object.



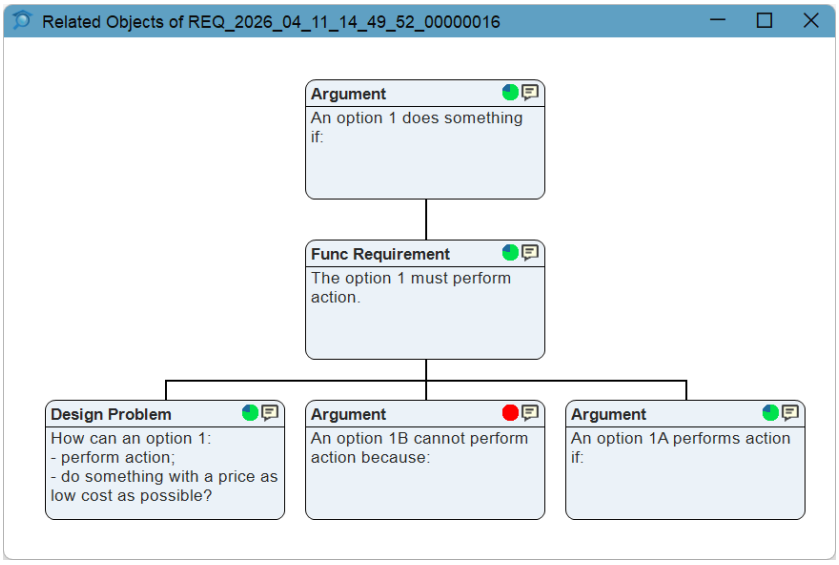
This text can be modified by selecting the ‘Edit tooltip text’ item in the Generic context menu.



Press OK to store the current text as the new tooltip text.

7.3.14 Show related objects

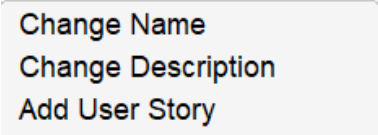
This Generic context menu item will visualize the related objects of the object as shown below.



See also §6.1 where the same function can be called from a Treeview Menu.

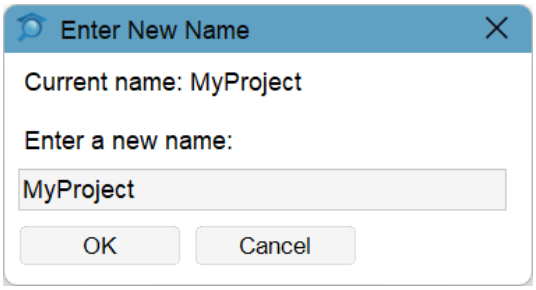
7.4 Project Object

This object represents the Project. It has the ‘Generic context menu’ items plus the following unique menu items.



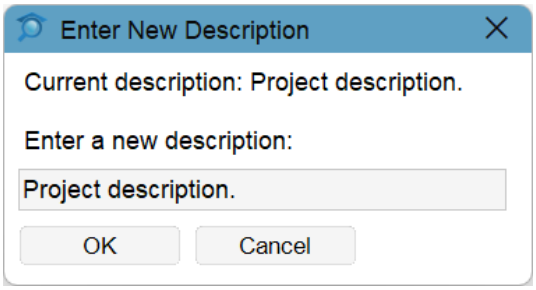
7.4.1 Change Name

This menu item can be used to change the name of the Project.



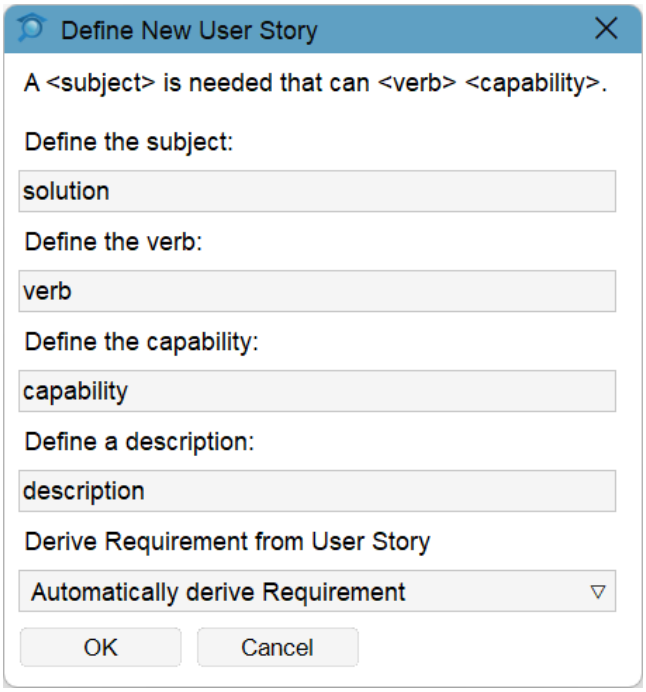
7.4.2 Change Description

This menu item can be used to change the description of the Project.



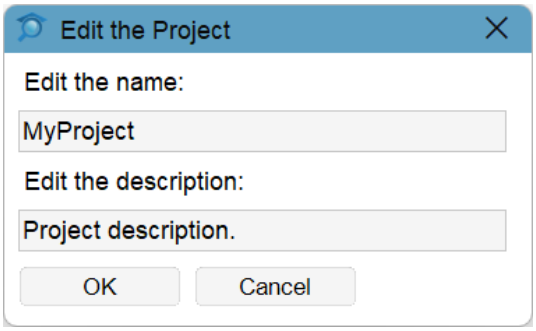
7.4.3 Add User Story

This menu item can be used to add a User Story to the Project. Define the User Story ‘subject’, ‘verb’ and ‘capability’. Also add a description and select whether to automatically derive a Requirement from the User Story or not.



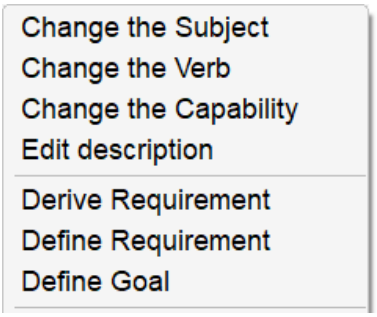
7.4.4 Double Click

Double clicking on the Project object will display a menu where the name and description can be modified.



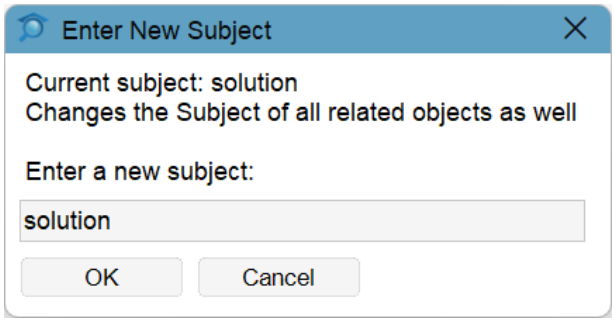
7.5 User Story Object

This object represents a User Story. It has the ‘Generic context menu’ items plus the following unique menu items.



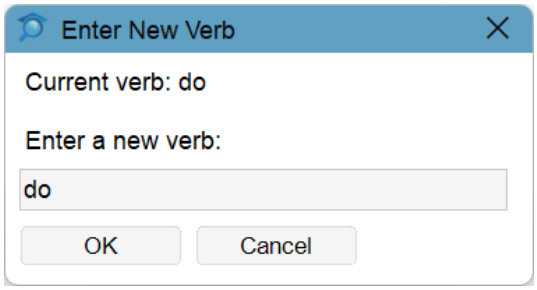
7.5.1 Change the Subject

This menu item can be used to change the subject of the User Story. Changing the subject will also change the subject of all linked objects (see also §2.3 Semantic Relations).



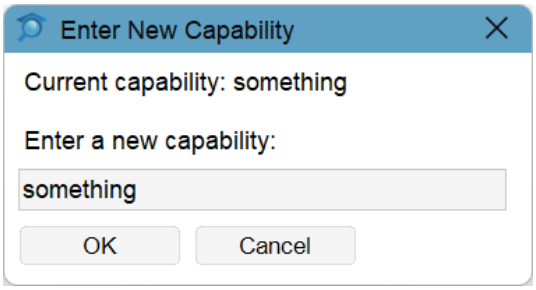
7.5.2 Change the Verb

This menu item can be used to change the verb of the User Story. Changing the verb will also change the verb of all linked objects (see also §2.3 Semantic Relations).



7.5.3 Change the Capability

This menu item can be used to change the capability of the User Story. Changing the capability will also change the capability of all linked objects (see also §2.3 Semantic Relations).



7.5.4 Edit description

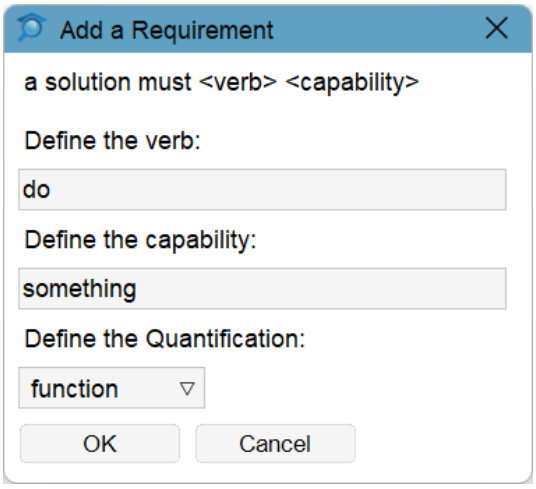
This menu item can be used to change the description of the User Story.

7.5.5 Derive Requirement

This menu item is only visible if there are no Requirement objects derived from this User Story yet. Selecting this creates a Requirement with the same subject, verb and capability at once. More Requirements can be added later using the ‘Define Requirement’ menu item.

7.5.6 Define Requirement

This menu item can be used to define a new Requirement from this User Story. Do not add too many Requirements right at the beginning of the Project. More specific Requirements should appear when more detail is necessary as the Project progresses.

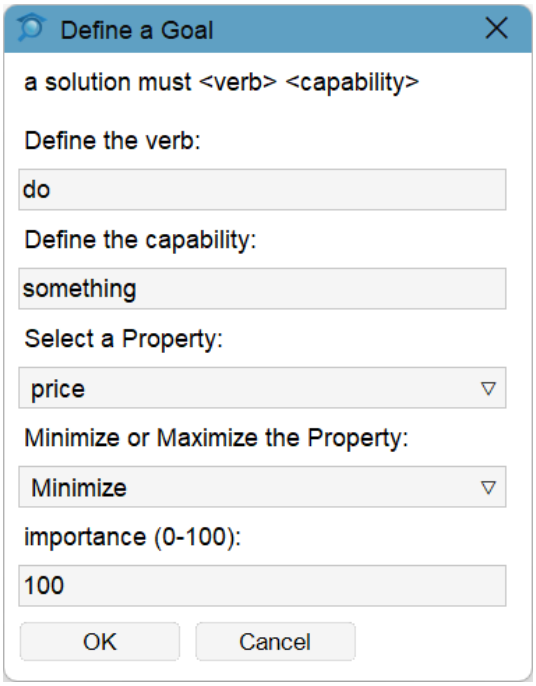


The pull down menu for the Quantification will display all active Parameters. Add a new Parameter in Tools Menu → Customize Parameters if a new Parameter is needed.

The Requirement is Functional when ‘function’ is selected for the Quantification. It can become a Goal, Range or Fixed Value Requirement if a Parameter is selected (see also §7.7 Requirement Object). If a Goal Requirement is defined, its importance/weight will be set to 100. Go to the Requirement object later if this value must be changed.

7.5.7 Define Goal

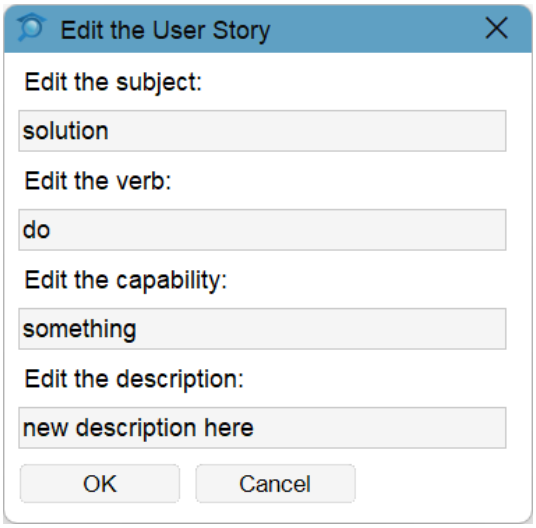
This menu item can be used to define a new Goal Requirement from this User Story. A Goal Requirement can also be defined using the 'Define Requirement' menu item and selecting min or max for the Parameter.



This menu allows direct definition of the importance. The importance (or weight) of the Goal Requirement defines the relative importance if multiple Goals are defined. This allows weighing the Goals with respect to each other when the optimal Option is determined.

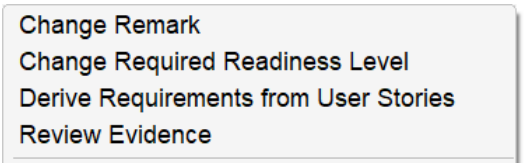
7.5.8 Double Click

Double clicking on the User Story object will show a window where the subject, verb, capability and description can be modified.



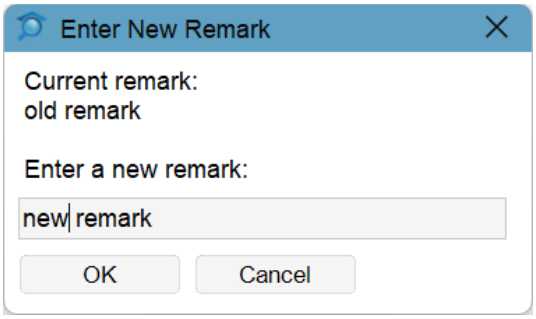
7.6 Mandate Object

This object represents a Mandate. It has the ‘Generic context menu’ items plus the following unique menu items.



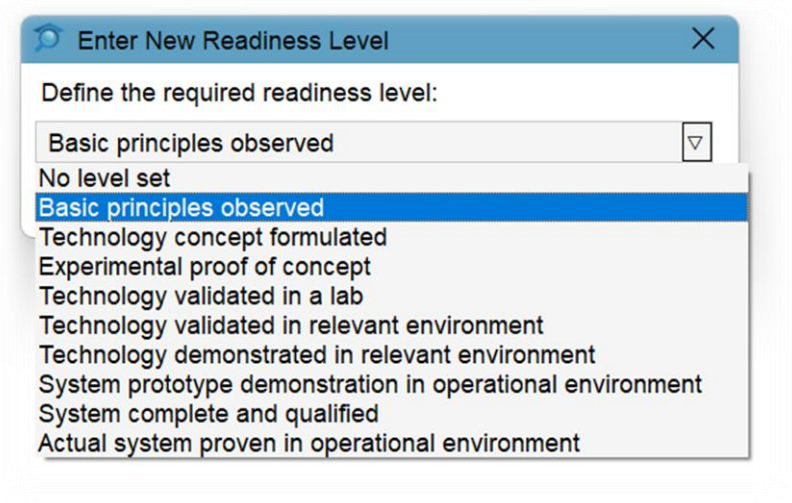
7.6.1 Change Remark

This menu item can be used to change the remark of the Mandate object.



7.6.2 Change Required Readiness Level

This menu item can be used to change the Readiness Level that is required in the Project.



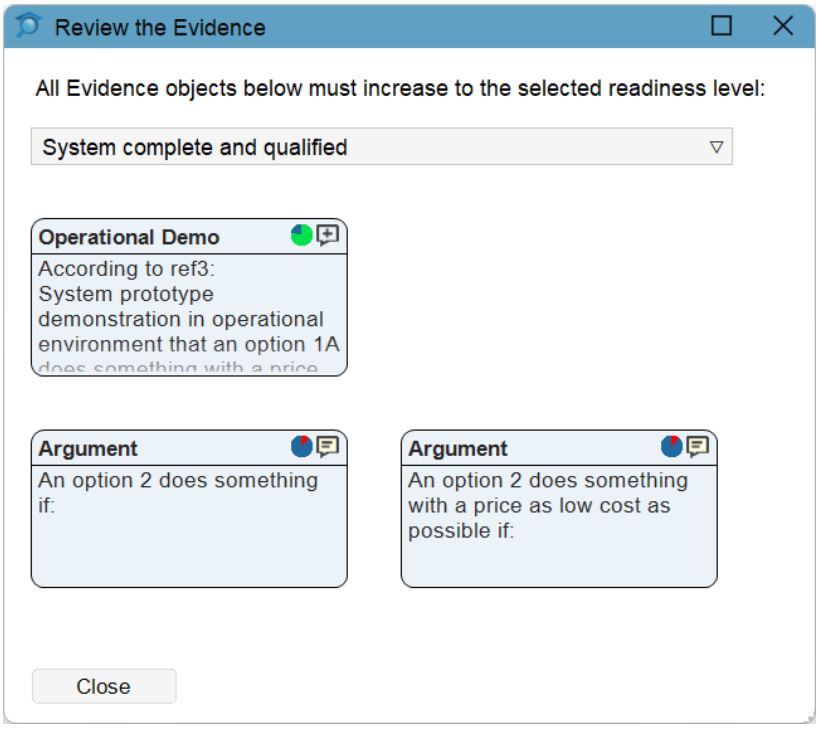
7.6.3 Derive Requirements from User Stories

This menu item can be used to automatically derive Requirements from User Story objects. It is only visible if there are User Story objects that do not have any derived Requirements yet.

7.6.4 Review Evidence

This menu item can be used to check the Evidence that is supporting the Project. It can also be used to quickly find unfinished branches in the Project.

Selecting this menu item displays the window below. Use this menu to investigate the Project Readiness Level status. It shows directly which Evidence is still needed to reach a certain Readiness Level.

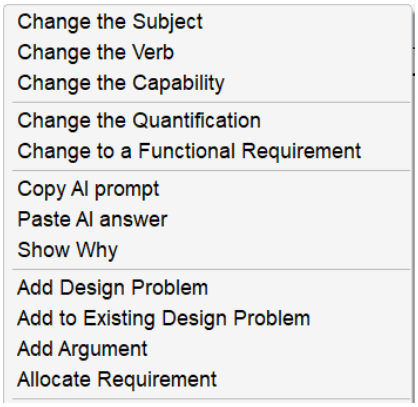


The required Readiness Level can be selected in the pull down menu (initially the Readiness Level required according to the Mandate object is taken). The window shows all Evidence Objects that do not validate the Options to the requested Readiness Level on the first row.

The second row shows all objects that are still invalid and do not have any children. For instance, if an Option does not have any Argument (yet). All objects should finally lead to an Evidence object in order to be considered valid. Any non-Evidence object that does not lead to further objects are unfinished branches in the Project.

7.7 Requirement Object

This object represents a Requirement. It has the ‘Generic context menu’ items plus the following unique menu items.



A Requirement can be defined as:

Functional, when the menu shows: function

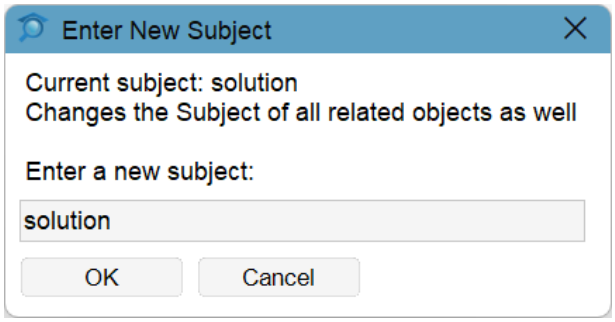
Goal, when the menu shows: length min

Range, when the menu shows: length > 10 mm

Fixed Value, when the menu shows: length = 0 m

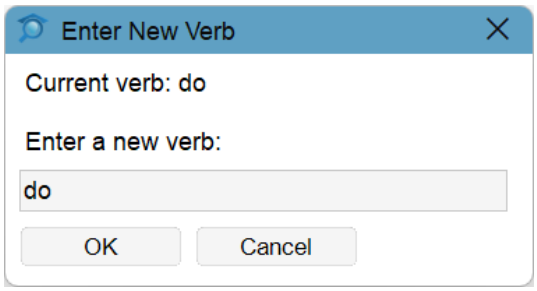
7.7.1 Change the Subject

This menu item can be used to change the subject of the Requirement. Changing the subject will also change the subject of all linked objects (see also §2.3 Semantic Relations).



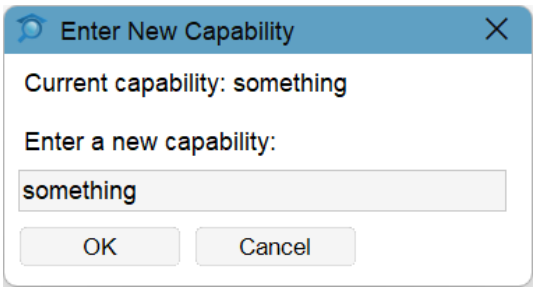
7.7.2 Change the Verb

This menu item can be used to change the verb of the Requirement. Changing the verb will also change the verb of all linked objects (see also §2.3 Semantic Relations).



7.7.3 Change the Capability

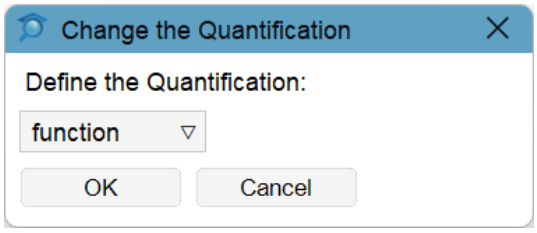
This menu item can be used to change the capability of the Requirement. The capability of all linked objects is changed if the capability is changed (see also §2.3 Semantic Relations).



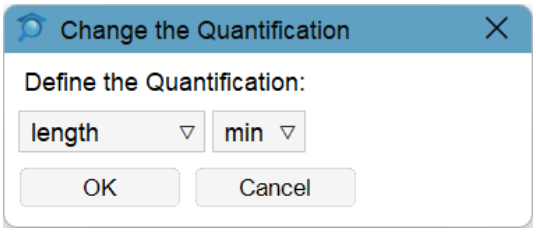
7.7.4 Change the Quantification

This menu item can be used to change the Parameter quantification of the Requirement. The Requirement can be quantified by any Parameter to a:

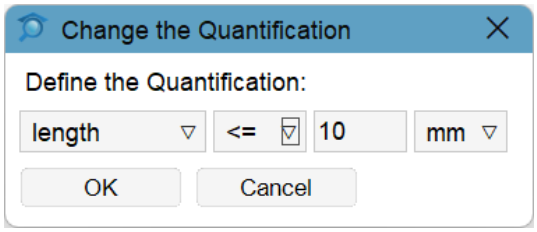
Functional Requirement



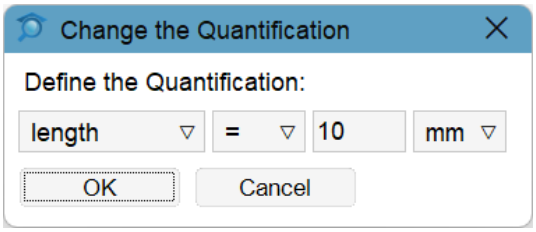
Goal Requirement



Range Requirement

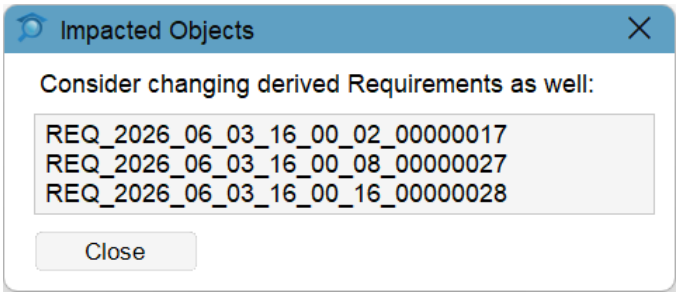


Fixed Value Requirement



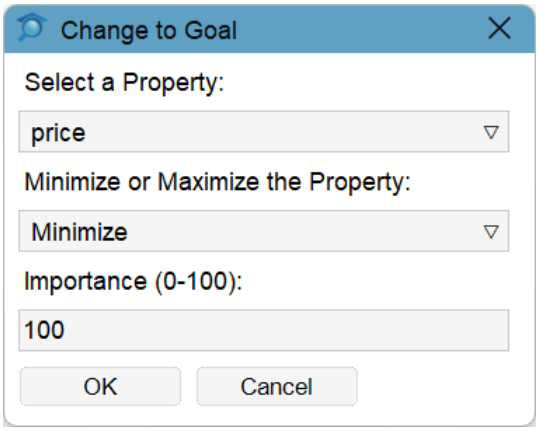
The pull down menu will show all active Parameters. Add or modify Parameters using ‘Tools Menu’ → ’Customize Parameters’.

A Window will appear if there are derived Requirements that might be impacted by this change in Quantification. Only derived Requirements with the same Parameter type are shown.



7.7.5 Change to a Goal

This menu item can be used to change the Requirement into a Goal Requirement.



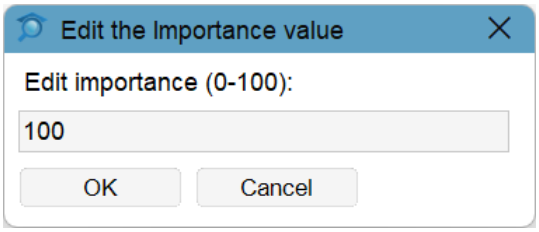
Define the Parameter type and whether to minimize or maximize that value. Also set the 'importance' of the Goal (see 'Edit Importance' below for explanation of 'importance').

7.7.6 Change to a Functional Requirement

This menu item can be used to change a quantified Requirement to a functional Requirement. The menu item will only appear in quantified Requirements.

7.7.7 Edit Importance

A Goal Requirement also has an importance. The importance normally is 100, except when competing Goal Requirements must be weighed against each other.



The 'importance' is used when the optimal Option Object is determined. The score of an Option is the sum of all Parameter values times its 'importance' factors. The 'importance' factor gets a minus sign if the goal is to minimize and positive if the goal is to maximize.

For example, if the goal is to have a high speed (importance 75) for low cost (importance 20).
The score for Option 1 (reaches 10 m/s for €25,-) is 75 x 10 - 20 x 25 = 250
The score for Option 2 (reaches 11 m/s for €30,-) is 75 x 11 - 20 x 30 = 225
Option 1 will be optimal (highest score) with the defined 'importance' factors.

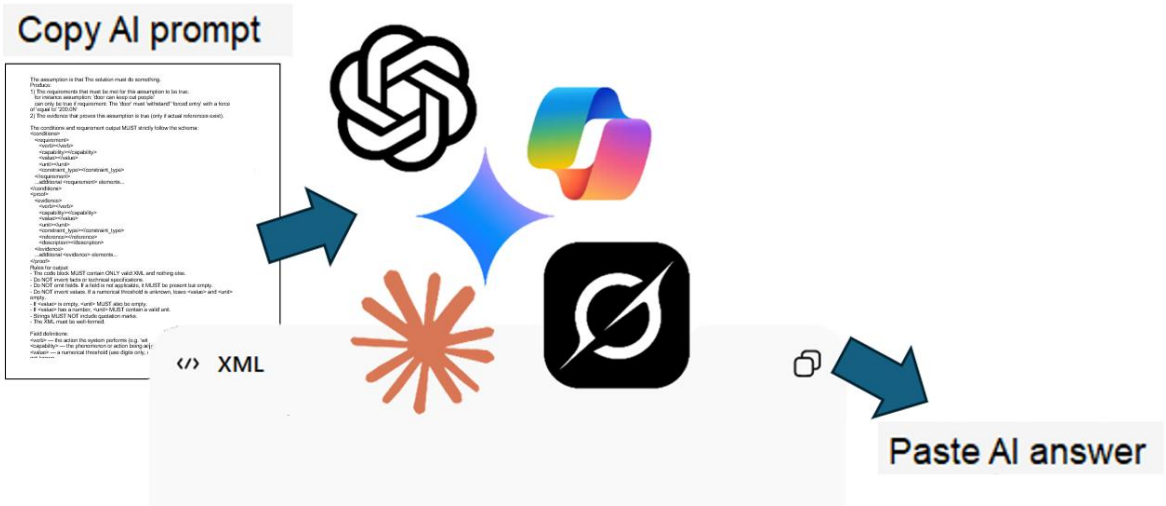
If both 'importance' factors would be 100, the scores would be: 100 x 10 - 100 x 25 = -1500 for Option 1 and 100 x 11 - 100 x 30 = -1900 for Option 2, which has the highest score in this case.

7.7.8 Copy AI prompt

This menu item can be used to create a Large Language Model AI prompt. It will only show up in the Requirements on the left of the Validation tab page.

The prompt will be copied to the clipboard and can be pasted directly into any Large Language Model interface. The prompt will ask how the Requirement can be met by the current Option and what Requirements must be met for the Option to be valid and what Evidence is available to prove that the Option can meet the Requirement.

The prompt requests that the answer is given in a specific format. The answer can then be directly copied and used in ‘Paste AI ’ if the format is properly used by the AI.



7.7.9 Paste AI answer

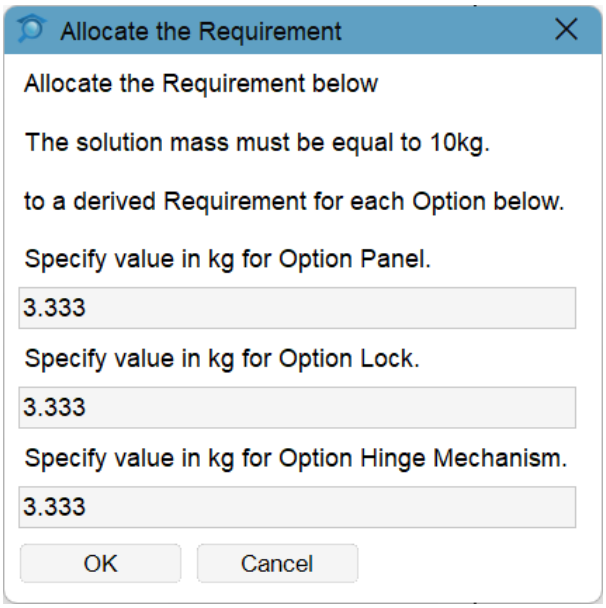
This menu item can be used to process a Large Language Model AI answer. It will only show up in the Requirements on the left of the Validation tab page.

The original prompt (created in ‘Copy AI prompt’) requests that the answer is given in a specific format. Make sure to copy the AI answer to the clipboard before using this menu item. Requirement Objects and/or Evidence Objects will be created if the format is properly used by the AI. Finally, inspect the result and make sure to carefully redact objects that were created.

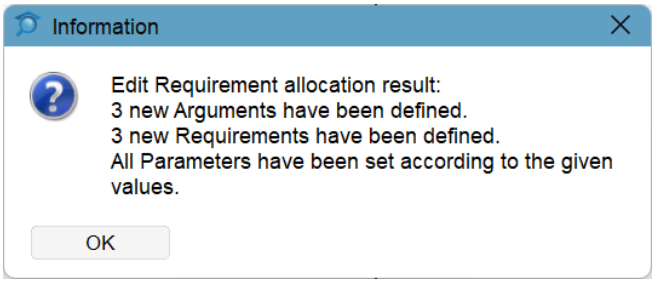
7.7.10 Allocate Requirement

This menu item can be used to allocate the quantitative value of the Requirement. It will only show up in Requirement objects on the left of the Validation tab page. The function works both to create new or modify existing Arguments and Requirements.

Selecting the function results in the window below. In this case the Quantitative Requirement value is distributed evenly over three Sub-Option Objects, because there were no derived Quantitative Requirement of that type yet.

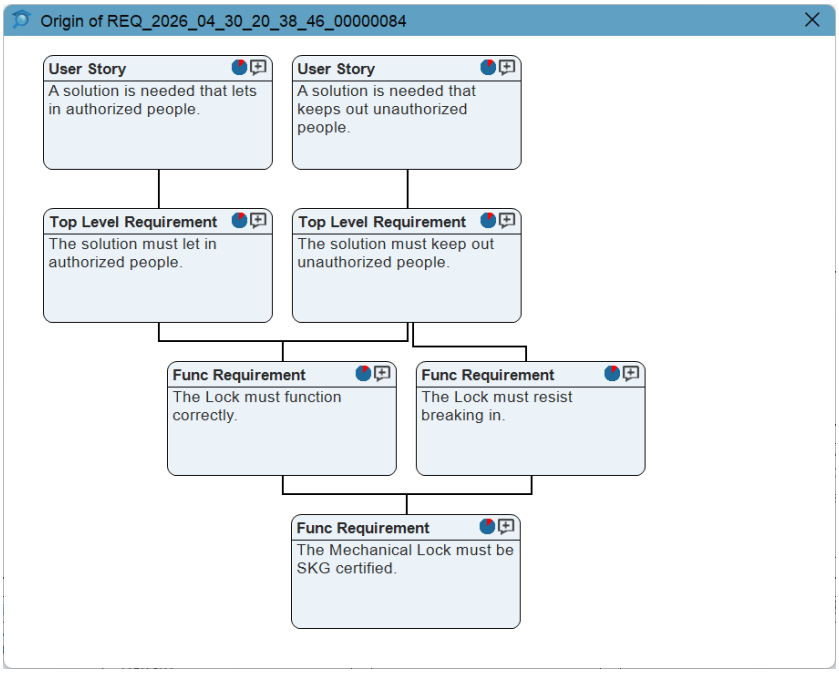


The value of existing derived Quantitative Requirements of the same type are displayed if they already exist. A message will show how many objects are created or modified.



7.7.11 Show Why

This menu item can be used to show why this Requirement exists. A window opens that displays the User Story objects and all Requirement objects that led to this Requirement (where 'show why' was selected).



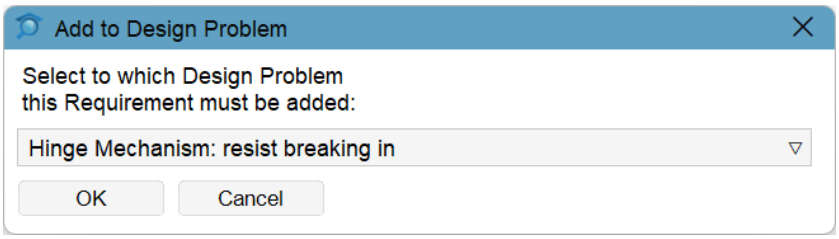
'Show why' can be called on each Requirement Object. The above example illustrates that a SKG certified Lock is assumed to validate that the Lock functions correctly and can resist breaking in. These Requirements in turn validate that 'the solution' can keep out or let people in. Those Requirements are derived from the User Story objects.

7.7.12 Add Design Problem

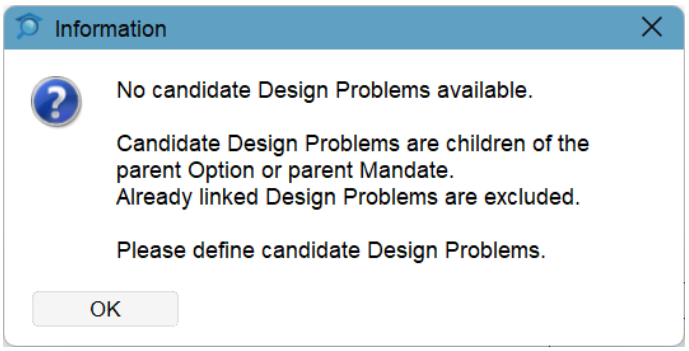
This menu item can be used to create a new Design Problem Object. This Requirement Object is input for the new Design Problem Object.

7.7.13 Add to Existing Design Problem

This menu item can be used to add this Requirement Object to the input of an existing Design Problem Object. The Requirement Object can only be added to Design Problem Objects that have the same parent Option or Mandate. This basically means that the candidate Design Problem Objects have the same Subject or have the same Subject of a Sub-Option Object of the same Option.

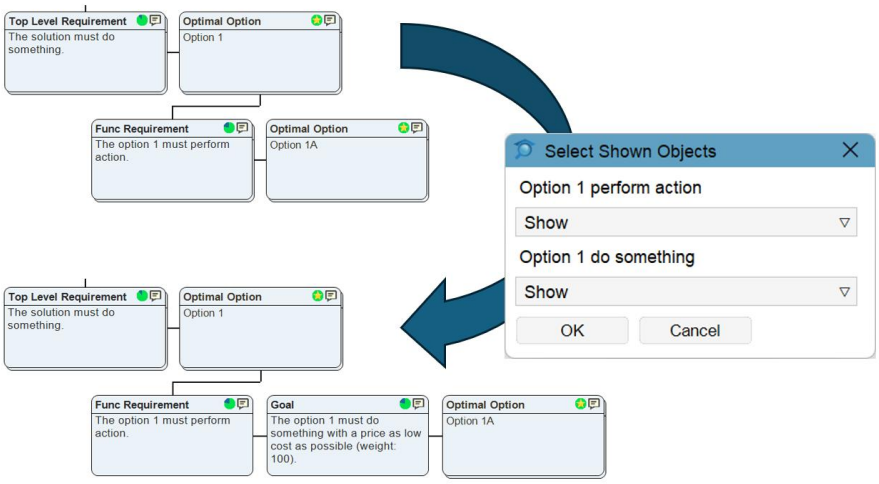


A window will open with a pull-down menu to select to which Design Problem this Requirement must be added.



7.7.14 Hide or Show Nodes

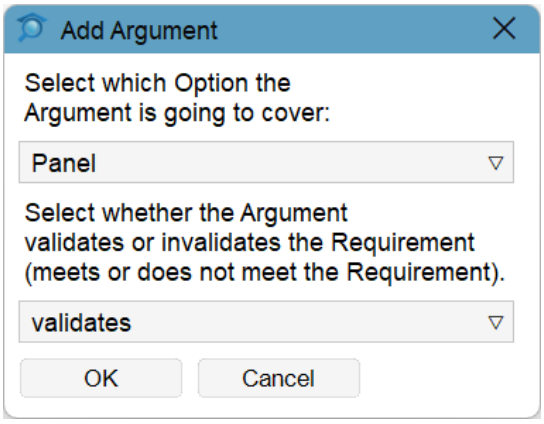
This menu item can be used to decide which Requirement Objects are shown in the V-diagram tab page. At least one Requirement Object is shown. The menu allows to show a different, more important, Requirement Object.



Select which Requirement Objects to Show or Hide in the window displayed above. It is also possible to show multiple Requirement Objects if that is preferred.

7.7.15 Add Argument

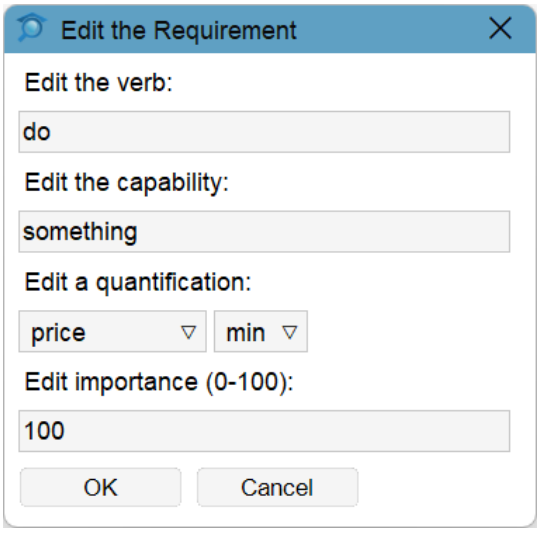
This menu item can be used to add an Argument Object and is only available if no Argument Object is created yet. An Argument Object is created directly in the Validation tab page with an Option without Sub-Option Object. In other cases, the window below will pop up.



Select the Option that the new Argument Object is going to validate (meets the Requirement) or invalidate (does not meet the Requirement). The Argument Object is created when OK is clicked.

7.7.16 Double Click

Double clicking on the Requirement object will show a window where the verb, capability and quantification can be modified.

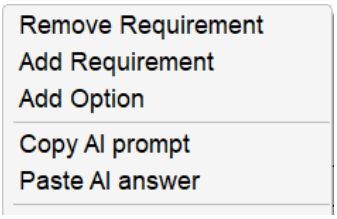


The example above also shows the ‘importance’ field, which is visible if the Requirement is a Goal Requirement. See Edit Importance in §7.7.7 for more information on ‘importance’.

A Window will appear if there are derived Requirements that might be impacted by a change in Quantification. Only derived Requirements with the same Parameter type are shown (see also ‘Change the Quantification’).

7.8 Design Problem Object

This object represents a Design Problem. It has the ‘Generic context menu’ items plus the following unique menu items.

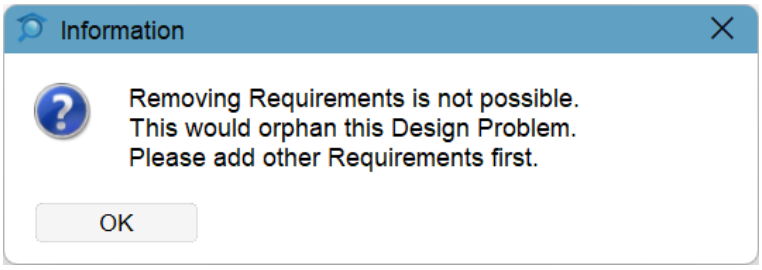


7.8.1 Remove Requirement

This menu item can be used to remove one of the input Requirements. The to be removed Requirement Object can be selected in the window that is shown below.

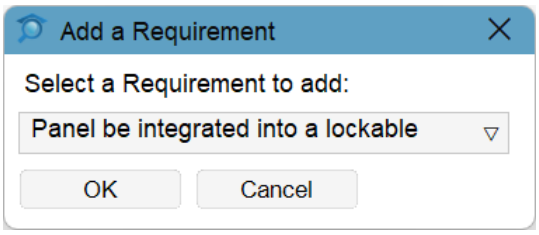
Removing an input Requirement will delete any linked child Arguments as well. The entire branch will be deleted below the Design Problem Object.

The window below highlights that nothing can be removed if the Design Problem Object only has one input Requirement. Use ‘Delete object’ if you insist that the object must be deleted.

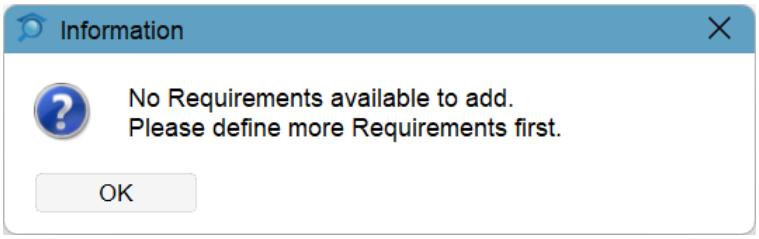


7.8.2 Add Requirement

This menu item can be used to add an input Requirement. The Requirement Objects that can be added can be selected in the window below.

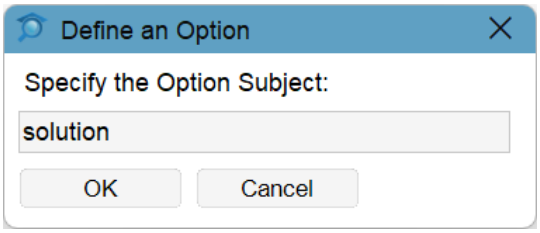


Only Requirements that exist under the same Option can be added. This includes Requirements that exist under other Sub-Option Objects of the same Option Object. It also includes related Integration Requirements. The window is shown if no further Requirements are available to be added.



7.8.3 Add Option

This menu item can be used to add an Option that can be a solution for the Design Problem.

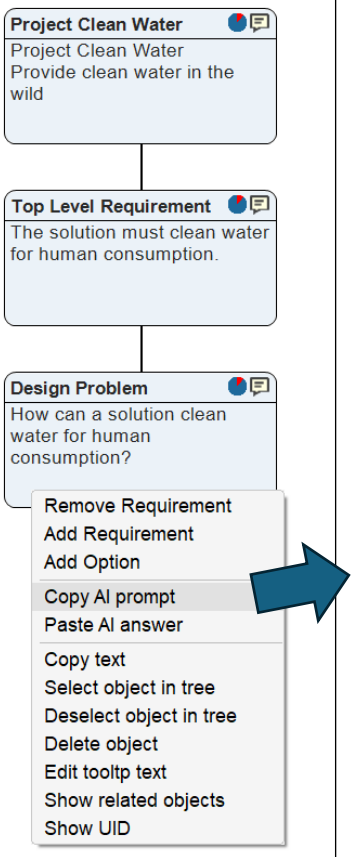


Use the window to specify the Subject of the Option Object.

7.8.4 Copy AI prompt

This menu item can be used to create a prompt for any Large Language Model (LLM) based on this Design Problem. The prompt will be copied on the clipboard. The prompt can be pasted into any LLM interface. The LLM is instructed to provide the answer in a specific format, so Option objects can be defined by using the 'Paste AI answer' menu item.

An example of a prompt is shown below.



How can a solution clean water for human consumption?

Your task:
1) Try to come up with high-level principles or categories FIRST.
2) Try to come up with concrete solutions ONLY IF no high-level principles or categories can be found.
3) Principles, categories or concrete solutions must be expressed as <solution> elements.

Output format:
Your answer MUST consist of exactly one code block containing XML with the following structure:
<solutions>
  <solution>
    <subject></subject>
    <description></description>
  </solution>
  ...additional solution elements...
</solutions>

Rules for output:
- The XML code block MUST contain ONLY valid XML and nothing else.
- Do NOT omit fields.
- If a field is not applicable, it MUST be present but EMPTY.
- Do NOT invent facts or technical specifications.
- NO commentary, explanation, or reasoning outside the XML.
- <subject> must be a short label(1–3 words) describing the principle, category, or solution.
- <description> must explain the principles, categories or concrete solutions clearly.

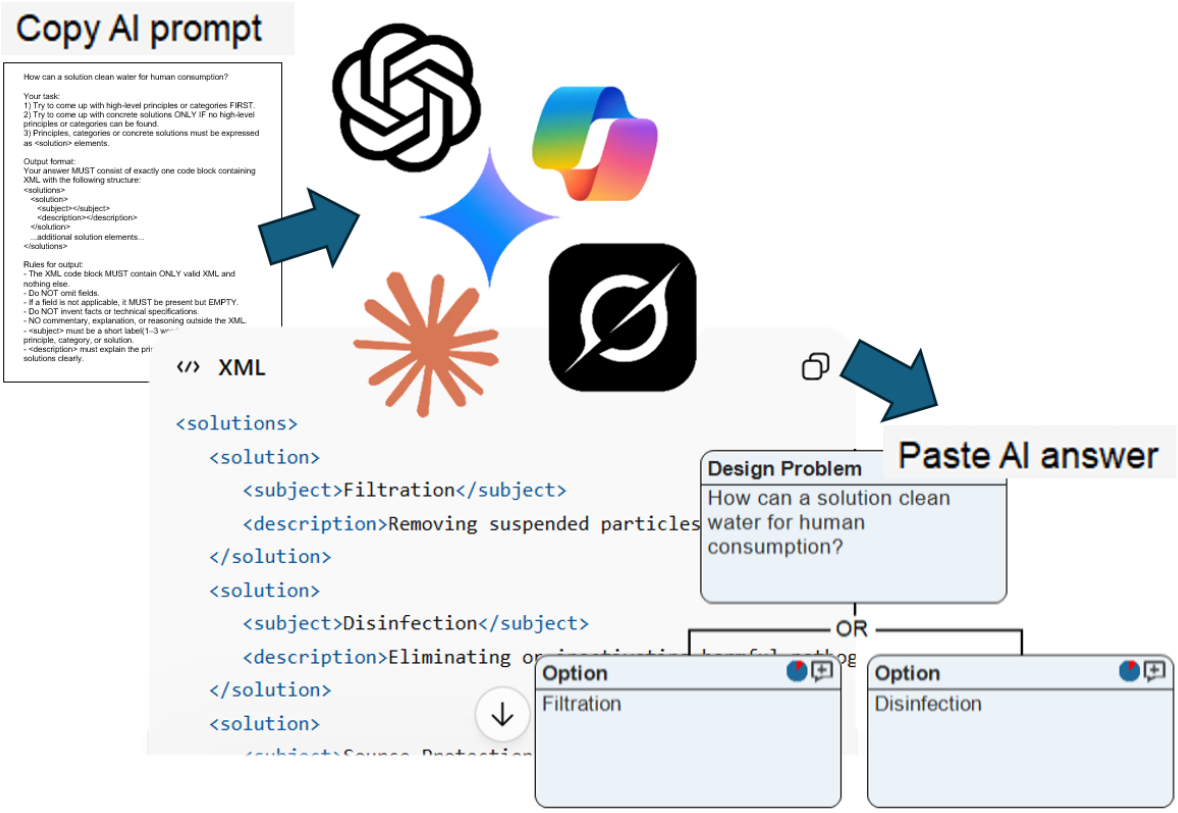
Field definitions:
<subject> — a very short description of the principle, category or concrete solution, preferably one word.
<description> — a description of how this principle, category or concrete solution helps solve the problem.

Before generating output, validate internally that all formatting and schema rules are satisfied.

7.8.5 Paste AI answer

This menu item can be used to create Option objects from a specifically formatted text on the clipboard. Previously mentioned text was generated by a Large Language Model (LLM) using the prompt from the ‘Copy AI prompt’ menu item.

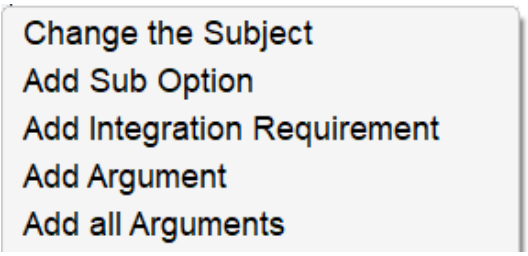
An example of this process is shown below.



Sometimes new Requirement Objects need new Parameter types to quantify them properly. Therefore pasting an AI answer can also create new Parameter types to enable the proper quantification of the new Requirement Objects.

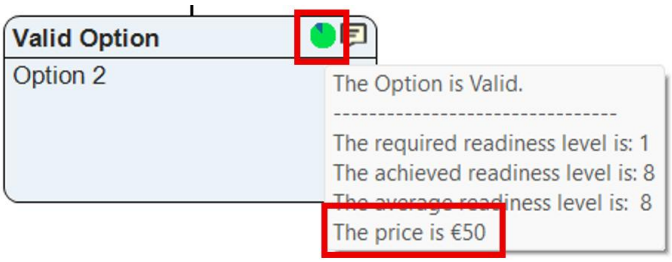
7.9 Option Object

This object represents an Option. It has the ‘Generic context menu’ items plus the following unique menu items.



7.9.1 Option Readiness Level Icon

Options can be Invalid, Valid or Optimal (see also §2.2.7). The tooltip text of the Readiness Level Icon of the Option will also show the sum of all Parameter values from lower level Evidence Objects. These summed Parameter values define which Option is optimal if there is a Goal Requirement defined in the Project (otherwise the Option with the highest Readiness Level will be optimal).

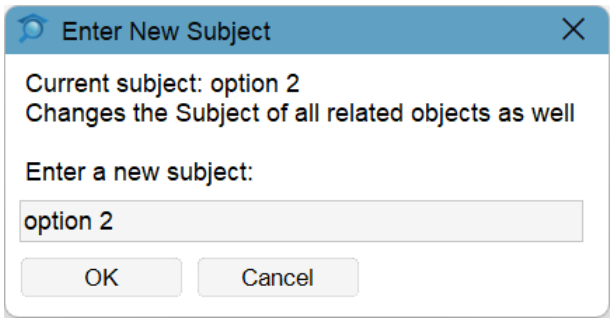


Create a new custom Parameter in ‘Customize Parameters’ in case multiple Parameters of the same type must be recorded separately. Two specific derivative ‘force’ Parameters must be created if, for instance, a spoiler must create a certain ‘down-force’ while the ‘drag-force’ must remain below a certain value. Having ‘down-force’ and ‘drag-force’ as separate ‘forces’ allows the Inquirement summation functionality to keep track of them, since having just ‘force’ would sum all ‘force’ Parameter values and break the logic.

Be aware that having two or more Evidence Objects with the same Parameter type for one Option will show the sum of both Parameter values in the Option Object.

7.9.2 Change the Subject

This menu item can be used to change the name or Subject of the Option. Changing the subject will also change the subject of all linked objects (see also §2.3 Semantic Relations).

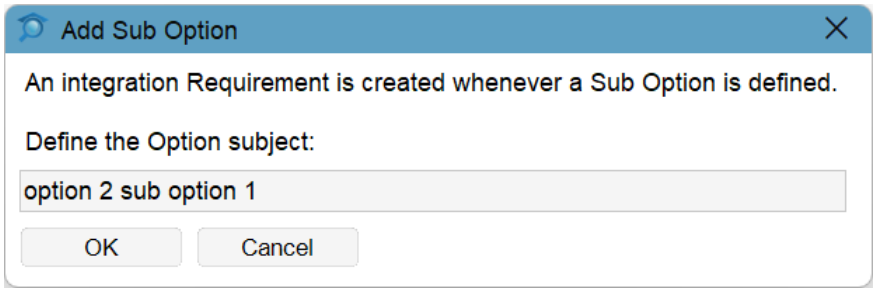


7.9.3 Add Integration Requirement

This menu item can be used to add an Integration Requirement to the Option. The menu item is only visible if an Integration Requirement is lacking (see also §2.2.5).

7.9.4 Add Sub Option

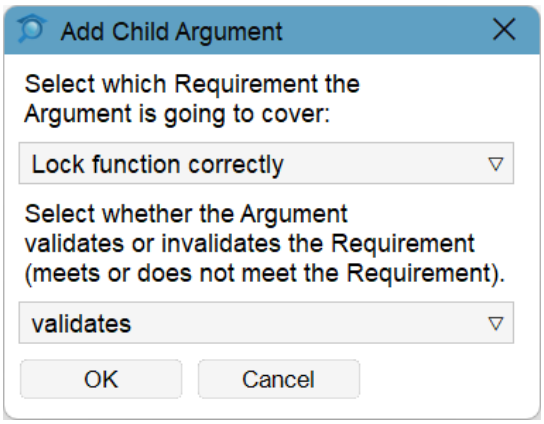
This menu item can be used to create a Sub-Option Object to an Option. An Integration Requirement is automatically created when a Sub-Option Object is created.



Use the window to define the name (Subject) of the Sub-Option Object.

7.9.5 Add Argument

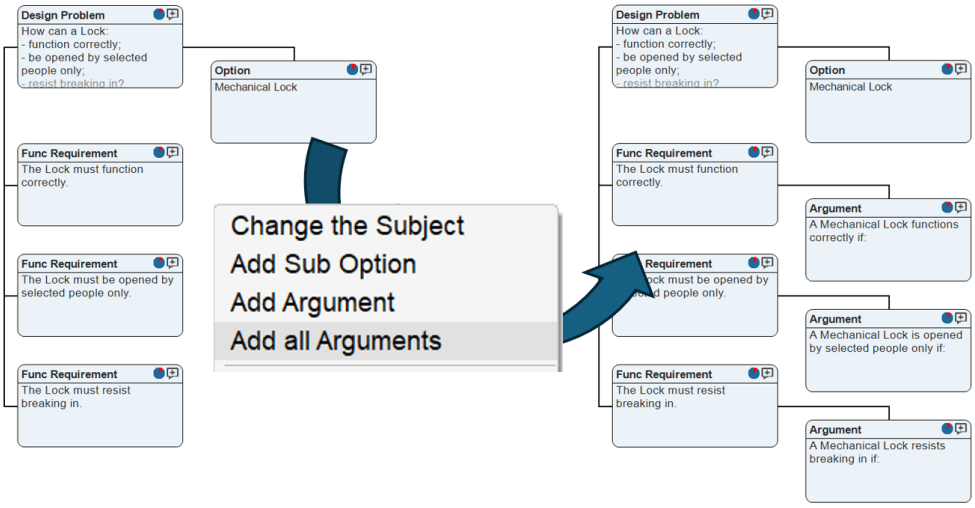
This menu item can be used to add an Argument Object and is only available in the Validation tab page if not all possible Argument Objects are created yet.



Select the Requirement that the new Argument Object is going to cover. Also select whether the Argument will validate (meets) or invalidate (does not meet) the Requirement. The Argument Object is created in the Validation tab page when OK is clicked.

7.9.6 Add all Arguments

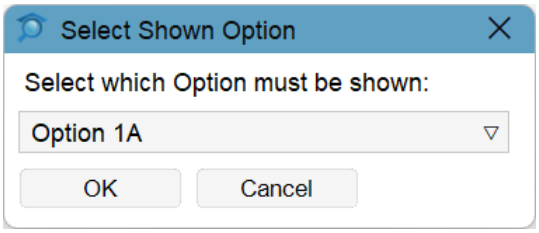
This menu item can be used to add all Argument Objects at once. The menu item is only available in the Validation tab page if not all possible Argument Objects are created yet.



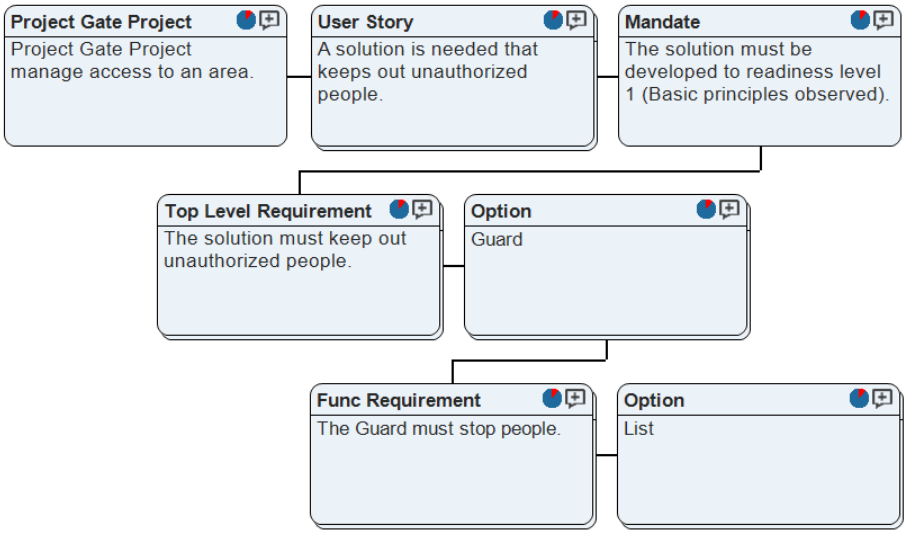
Use this function to quickly create all Argument Objects at once.

7.9.7 Hide or Show Nodes

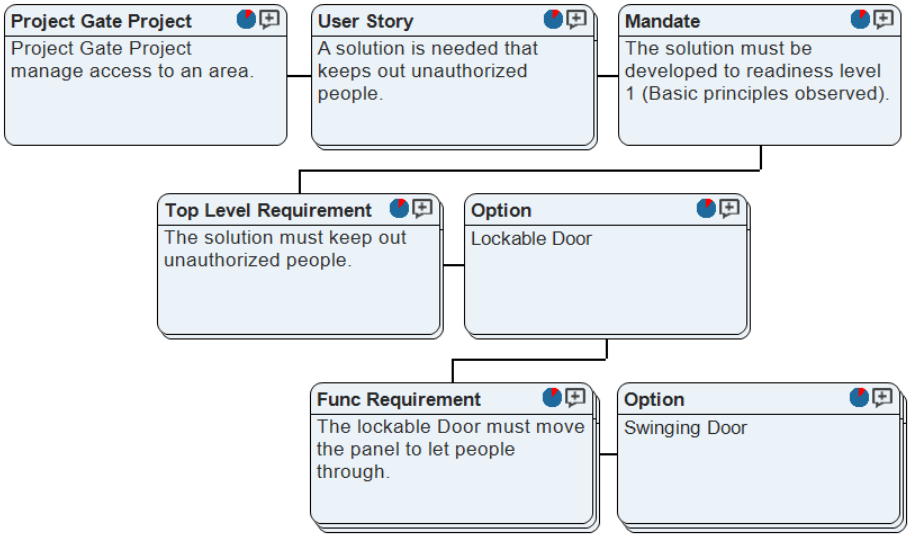
This menu item can be used to select which Option is shown in the V-diagram tab page. Use the drop down menu in the window to select the Option that must be shown. This menu item is only visible in V-diagram tab page.



Rows below the Option will adapt to the selected Option Object. The example below illustrates a V-diagram tab page where the Option 'Guard' is selected in the second row.

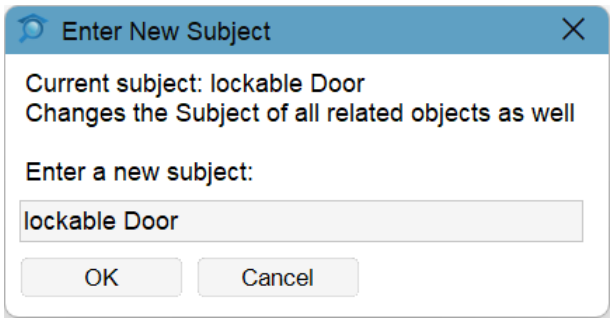


The Option 'List' on the lower row is replaced by 'Swinging Door' if the Option 'Lockable Door' is selected instead of 'guard'. 'List' belongs to 'Guard' and is irrelevant for a 'Lockable Door'.



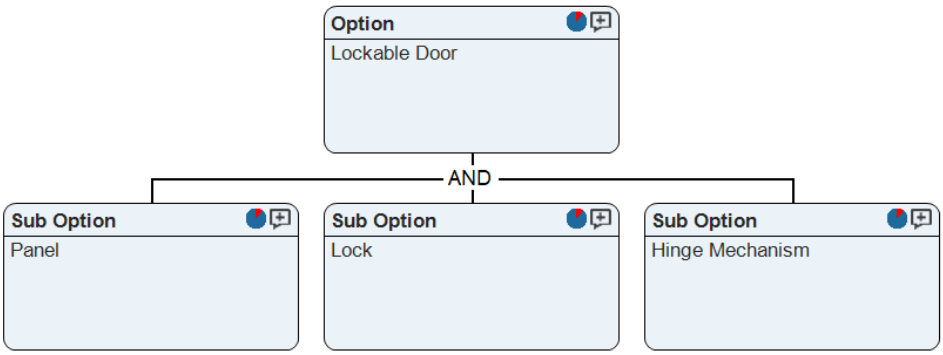
7.9.8 Double Click

Double clicking on the Option object will show a window where the name or Subject can be modified.

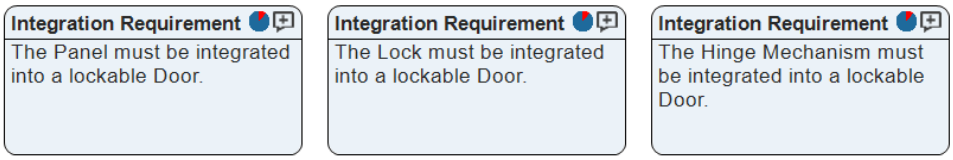


7.10 Sub-Option Object

This object is similar to an Option object, as they have the same menu items. A Sub-Option is intended to define a functional breakdown of the Option.



The example above shows how a Lockable Door is broken down into the 'Panel', 'Lock' and 'Hinge Mechanism' Sub-Options. Integration Requirements are automatically created whenever an Option is broken down into Sub-Options. The Integration Requirements below are automatically derived from the example above.

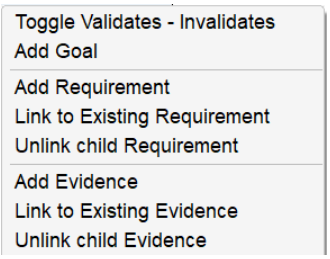


The Top-Option can only be valid if all Integration Requirements are valid as well, hence the AND in the lines connecting the Sub-Options to the Option. It is possible to delete Integration Requirements. They can be created again in the Option Object 'Add Integration Requirement' menu item.

The Integration Requirements can be solved the same way that normal Requirements are solved. For instance, possible options for the middle Requirement in the example above are screwing, nailing or bonding the lock in the door.

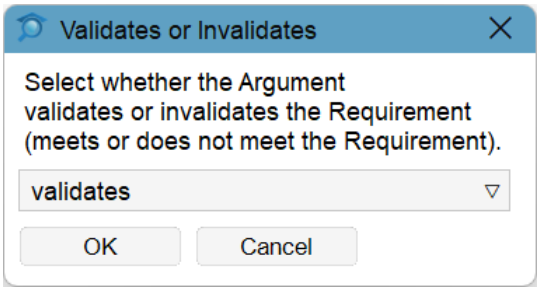
7.11 Argument Object

This object represents an Argument. It has the ‘Generic context menu’ items plus the following unique menu items.



7.11.1 Toggle Validates – Invalidates

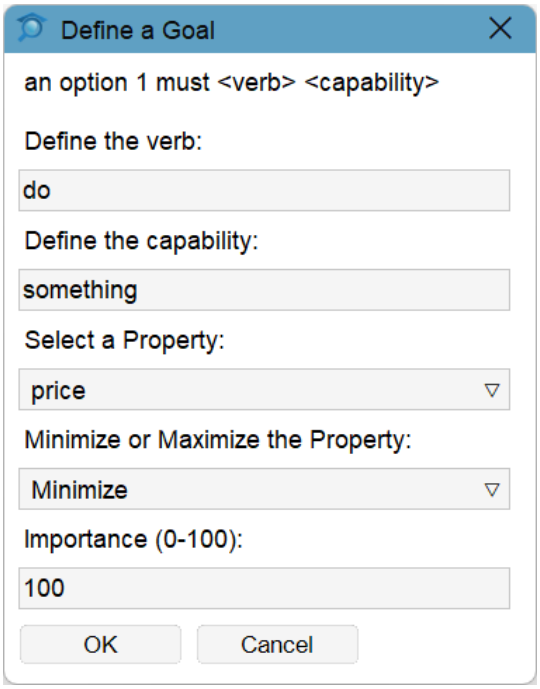
This menu item can be used to set the Argument to ‘validates’ or ‘invalidates’. All Argument Objects ‘validate’ the Option Object by default.



Set the Argument Object to ‘invalidates’ if linked Evidence Objects prove that the Option cannot be valid. Toggling can also be done by Double Clicking the Argument Object.

7.11.2 Add Goal

This menu item can be used to add a Goal Requirement to the Argument.

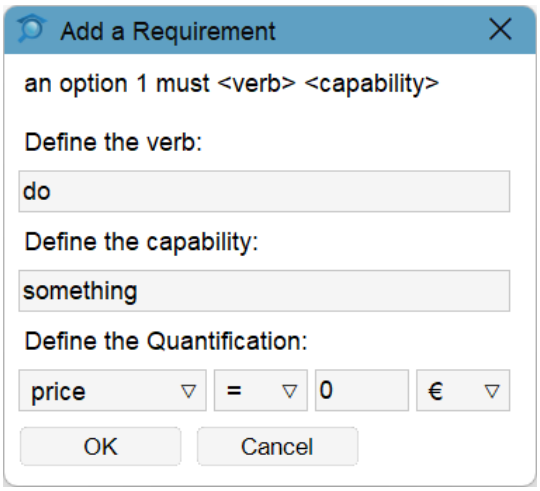


Define the Verb, Capability and Parameter properties. Also define whether the goal should Minimize or Maximize the Parameter property and define the 'Importance' (see also Edit Importance).

7.11.3 Add Requirement

This menu item can be used to add a Requirement to the Argument. The Requirement will function as a condition for the Argument to be true.

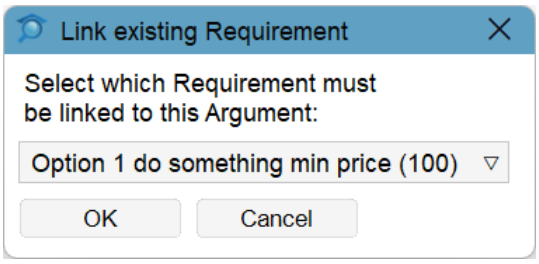
Identified risks related to the Option can be captured as Requirements that the risk must not happen. The Option will never be valid until a valid corrective action is defined.



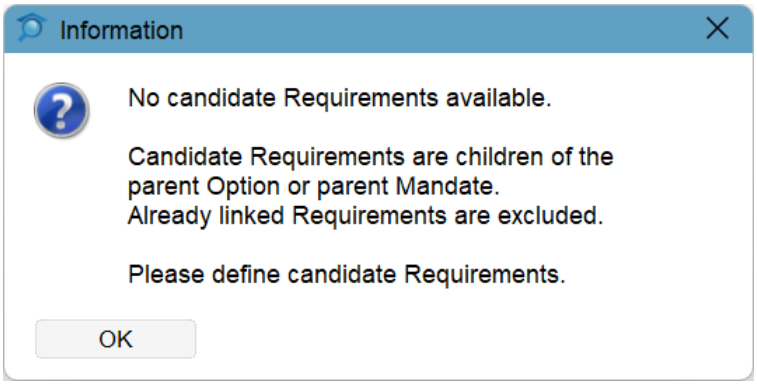
Define the Verb, Capability and Quantification Parameter. Notice that a Goal is defined if 'min' or 'max' is selected in the second pull-down (with the '=').

7.11.4 Link to Existing Requirement

This menu item can be used to link the Argument to an existing Requirement. The Requirement will function as a condition for the Argument to be true.



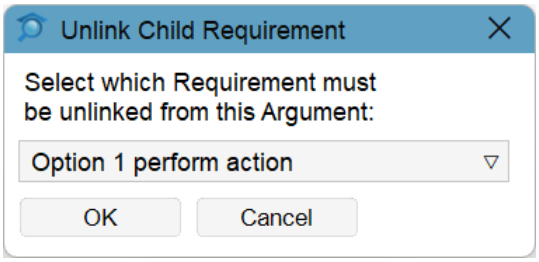
Choose one of the available Requirements in the pull-down menu. Only Requirements that are directly below the same Option can be chosen. This includes Requirements below different Sub-Option Objects of a top Option Object. It also includes related Integration Requirements.



The window above appears if no candidate Requirements are available.

7.11.5 Unlink child Requirement

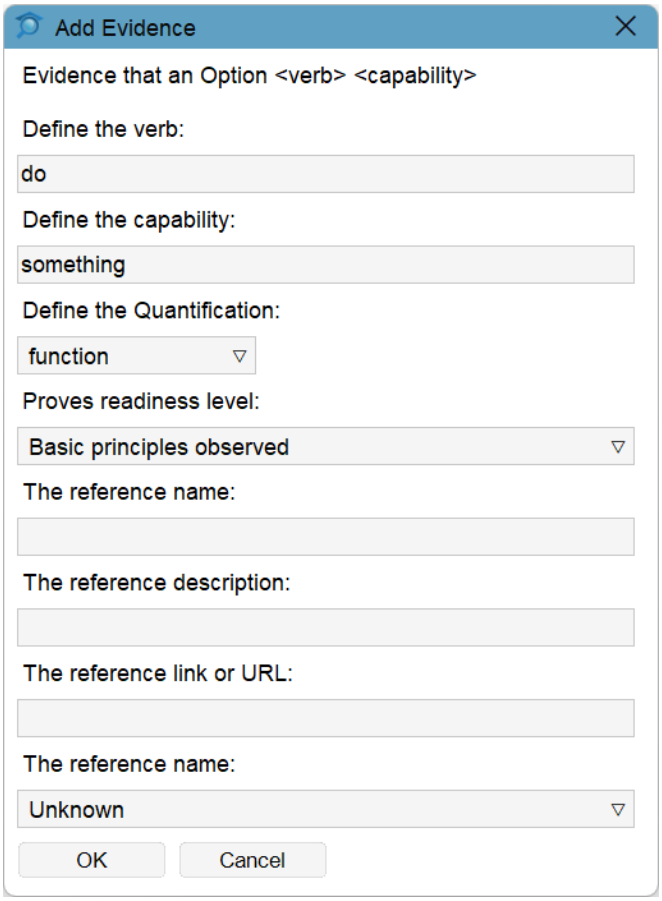
This menu item can be used to unlink the Argument from linked Requirements. The Requirement will no longer function as a condition for the Argument to be true.



The menu item is only visible if there are Requirement Objects related to this Argument Object.

7.11.6 Add Evidence

This menu item can be used to add an Evidence Object to the Argument. The Evidence Object will prove that the Argument is valid (or disprove if the Argument is set to 'invalidates' using 'Toggle Validates – Invalidates').

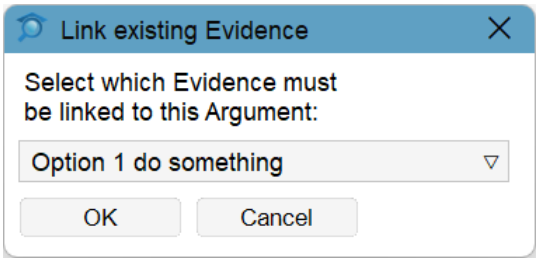


Define the Verb, Capability and Quantification Parameter. Also define up to what Readiness Level the Evidence proves the Argument.

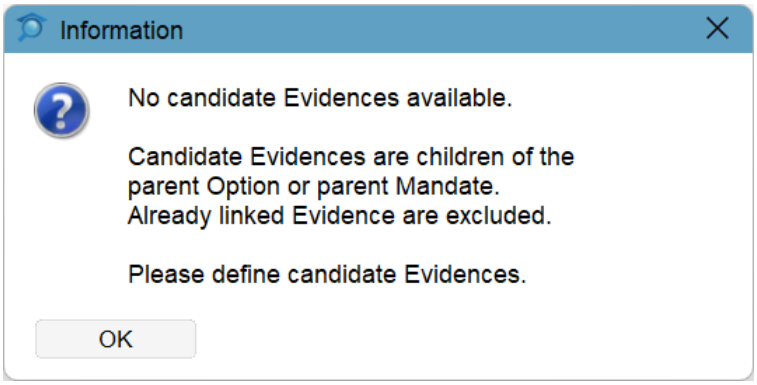
The window also has fields to define the Reference.

7.11.7 Link to Existing Evidence

This menu item can be used to link the Argument to an existing Evidence Object. The Evidence Object will prove that the Argument is valid.



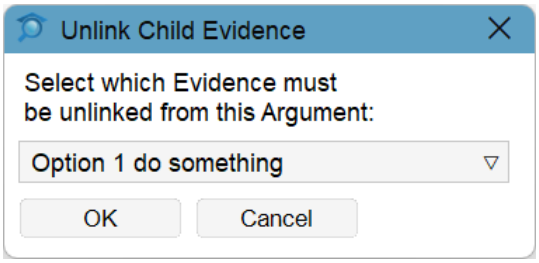
Choose one of the available Evidence Objects in the pull-down menu. Only Evidence Objects that are directly below the same Option can be chosen. This includes Requirements below different Sub-Option Objects of a top Option Object.



The window above appears if no candidate Evidence Objects are available.

7.11.8 Unlink child Evidence

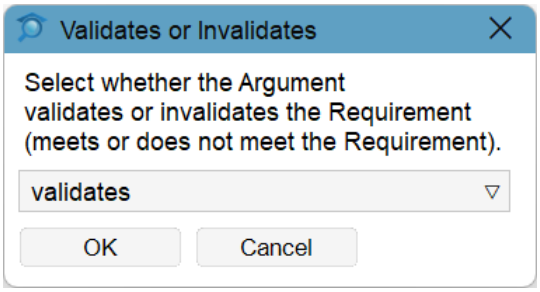
This menu item can be used to unlink the Argument from linked Evidence Objects. The Evidence Object will no longer prove that the Argument is valid.



The menu item is only visible if there are Evidence Objects related to this Argument Object.

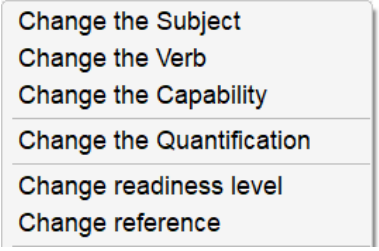
7.11.9 Double Click

Double clicking on the Argument Object will show a menu to set the Argument to ‘validates’ or ‘invalidates’ (see also §7.11.1).



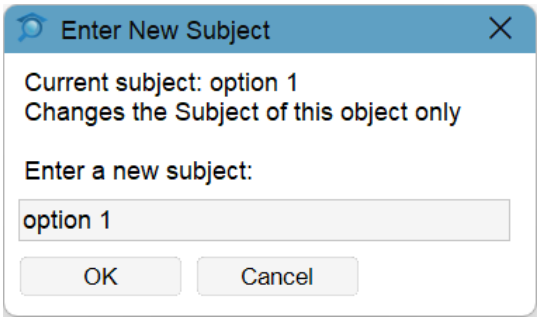
7.12 Evidence Object

This object represents an Evidence. It has the ‘Generic context menu’ items plus the following unique menu items.



7.12.1 Change the Subject

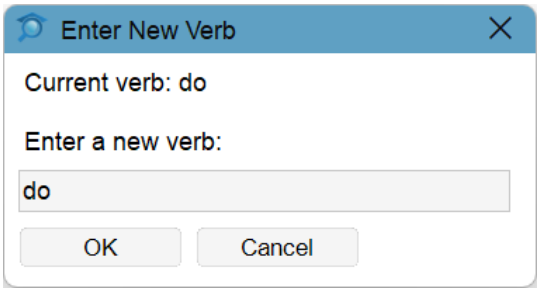
This menu item can be used to change the Subject of the Evidence Object.



Changing the Subject will not change the subject of other objects. The Evidence is a separate piece of evidence.

7.12.2 Change the Verb

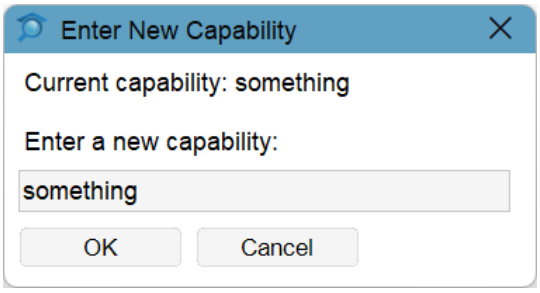
This menu item can be used to change the Verb of the Evidence Object.



Changing the Verb will not change the verb of other objects. The Evidence is a separate piece of evidence.

7.12.3 Change the Capability

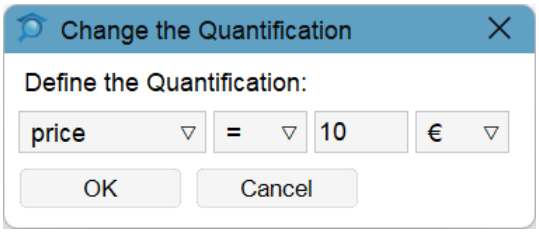
This menu item can be used to change the Capability of the Evidence Object.



Changing the Capability will not change the capability of other objects. The Evidence is a separate piece of evidence.

7.12.4 Change the Quantification

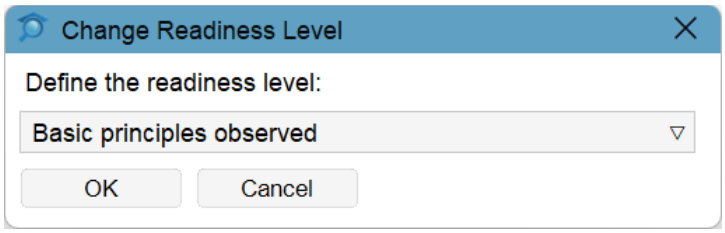
This menu item can be used to change the Parameter Quantification of the Evidence Object.



See also §7.7.4 Change the Quantification of the Requirement Object.

7.12.5 Change the Readiness Level

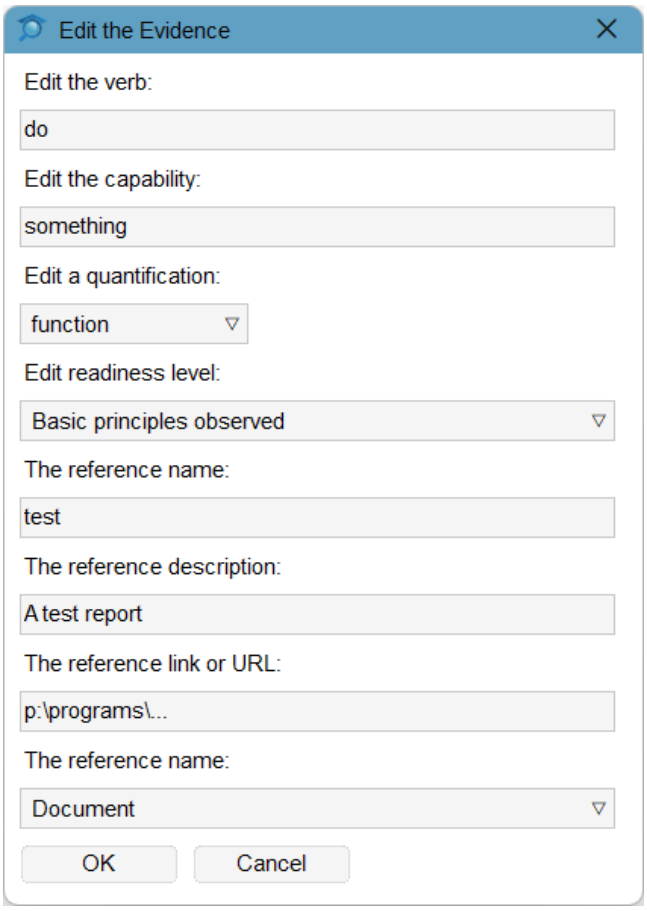
This menu item can be used to change the Readiness Level of the Evidence Object.



This states to what maturity or risk level the Evidence is proven.

7.12.6 Double Click

Double clicking on the Evidence Object will show a window where the Subject, Verb, Capability, Quantification, Readiness Level and Reference can be modified at once.



8 Inquirement files

A Project is saved using the ‘Save’ or ‘Save As’ File Menu items or pressing the ‘Save’ or ‘Save As’ button in the Toolbar. The Project is saved as an xml file, which provides maximum openness and allows inspection. It is not recommended to edit this file directly.

8.1 Default Possible Parameter Activity

```
<PPA>
  <par_type>length</par_type>
  <unit_is_active>true or false; whether the unit is active</unit_is_active>
  <unit_is_active>true or false for each unit</unit_is_active>
</PPA>
```

8.2 Custom Possible Parameter

```
<PPN>
  <par_type>new Parameter type name</par_type>
  <par_target_text>text if ‘as small as possible’</par_target_text>
  <par_target_text>text if ‘lower than’</par_target_text>
  <par_target_text>text if ‘lower than or equal’</par_target_text>
  <par_target_text>text if ‘equal to’</par_target_text>
  <par_target_text>text if ‘larger than or equal’</par_target_text>
  <par_target_text>text if ‘larger than’</par_target_text>
  <par_target_text>text if ‘as large as possible’</par_target_text>
  <PPU>
    <par_unit_symbol>base unit symbol</par_unit_symbol>
    <par_factor>base unit factor as real, must be 1.0</par_factor>
    <par_delta>base delta factor as real, must be 0.0</par_delta>
    <unit_is_active>true or false; whether the unit is active</unit_is_active>
  </PPU>
  <PPU>
    <par_unit_symbol>additional unit symbol</par_unit_symbol>
    <par_factor>additional unit factor as real</par_factor>
    <par_delta>additional delta factor as real</par_delta>
    <unit_is_active>true or false; whether the unit is active</unit_is_active>
  </PPU>
</PPN>
```

8.3 Project Object

```
<PRJ>
  <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
  <subject>define Subject here</subject>
  <ref_id>unique ID of an existing Reference in the Project</ref_id>
  <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
  <child>add child UID here. Add as many lines like this as there are children</child>
  <comments>
    <comment>
      <author>User A</author>
      <content>comment content here (see also §7.3.13)</content>
    </comment>
  </comments>
  <name>Project name (see also §7.4.1) </name>
  <description>Project description (see also 7.4.2)</description>
  <sett_user_name>User Name according to Windows settings</sett_user_name>
  <Inquirement_version>1.0.0.0</Inquirement_version>
  <sett_font_size>14 (edit font size buttons §5.12 and §5.13)</sett_font_size>
  <sett_zoom_prj>Project tab page zoom (§5.14, §5.15 and §5.16)</sett_zoom_prj>
  <sett_zoom_prb>Problems tab page zoom (§5.14, §5.15 and §5.16)</sett_zoom_prb>
  <sett_zoom_opt>Options tab page zoom (§5.14, §5.15 and §5.16)</sett_zoom_opt>
  <sett_zoom_val>Validation tab page zoom (§5.14, §5.15 and §5.16)</sett_zoom_val>
  <sett_zoom_req>Requirements tab page zoom (§5.14, …)</sett_zoom_req>
  <sett_zoom_vdi>V-diagram tab page zoom (§5.14, …)</sett_zoom_vdi>
  <sett_show>All+ | Show Integration Requirements (see §5.9)</sett_show>
  <sett_prj_color>Project Object color as RXXXGYYYBZZZ</sett_prj_color>
  <sett_usy_color>User Story color as RXXXGYYYBZZZ</sett_usy_color>
  <sett_man_color>Mandate  color as RXXXGYYYBZZZ</sett_man_color>
  <sett_dpr_color>Design Problem color as RXXXGYYYBZZZ</sett_dpr_color>
  <sett_req_color>Requirement  color as RXXXGYYYBZZZ</sett_req_color>
  <sett_opt_inv_color>invalid Option color as RXXXGYYYBZZZ</sett_opt_inv_color>
  <sett_opt_val_color>valid Option color as RXXXGYYYBZZZ</sett_opt_val_color>
  <sett_opt_opt_color>optimal Option color as RXXXGYYYBZZZ</sett_opt_opt_color>
  <sett_arg_color>Argument color as RXXXGYYYBZZZ</sett_arg_color>
  <sett_evd_color>Evidence color as RXXXGYYYBZZZ</sett_evd_color>
  <sett_par_color>Parameter color as RXXXGYYYBZZZ</sett_par_color>
  <sett_prj_txt_color>Project text color as RXXXGYYYBZZZ</sett_prj_txt_color>
  <sett_usy_txt_color>User Story text as RXXXGYYYBZZZ</sett_usy_txt_color>
  <sett_man_txt_color>Mandate text as RXXXGYYYBZZZ</sett_man_txt_color>
  <sett_dpr_txt_color>Design Problem text as RXXXGYYYBZZZ</sett_dpr_txt_color>
  <sett_req_txt_color>Requirement text as RXXXGYYYBZZZ</sett_req_txt_color>
  <sett_opt_opt_txt_color> invalid Option text RXXXGYYYBZZZ</sett_opt_opt_txt_color>
  <sett_opt_val_txt_color> valid Option text RXXXGYYYBZZZ</sett_opt_val_txt_color>
  <sett_opt_inv_txt_color> optimal Option text RXXXGYYYBZZZ</sett_opt_inv_txt_color>
  <sett_arg_txt_color>Argument text as RXXXGYYYBZZZ</sett_arg_txt_color>
  <sett_evd_txt_color>Evidence text as RXXXGYYYBZZZ</sett_evd_txt_color>
  <sett_par_txt_color>Parameter text as RXXXGYYYBZZZ</sett_par_txt_color>
  <sett_panel_color>panel color as RXXXGYYYBZZZ</sett_panel_color>
  <sett_panel_sel_color>selected panel RXXXGYYYBZZZ</sett_panel_sel_color>
  <sett_text_color>text color as RXXXGYYYBZZZ</sett_text_color>
  <sett_text_sel_color>selected text color as RXXXGYYYBZZZ</sett_text_sel_color>
  <sett_box_connector_color>line color as RXXXGYYYBZZZ</sett_box_connector_color>
  <sett_box_border_color>object border RXXXGYYYBZZZ</sett_box_border_color>
  <sett_button_color>button color RXXXGYYYBZZZ</sett_button_color>
  <sett_btn_border_color>button border color RXXXGYYYBZZZ</sett_btn_border_color>
```

```
  <PRJ_Node_Expanded>true</PRJ_Node_Expanded>
  <REQ_Node_Expanded>false</REQ_Node_Expanded>
  <DPR_Node_Expanded>false</DPR_Node_Expanded>
  <OPT_Node_Expanded>true</OPT_Node_Expanded>
  <ARG_Node_Expanded>false</ARG_Node_Expanded>
  <EVD_Node_Expanded>false</EVD_Node_Expanded>
  <REF>
    <ref_id>1</ref_id>
    <ref_name>EN 13307</ref_name>
    <ref_description>Parts 1 & 2</ref_description>
    <ref_link>www.nen.nl</ref_link>
    <ref_type>Unknown</ref_type>
  </REF>
    Further References…
</PRJ>
```


8.4 User Story Object


```
<USY>
  <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
  <subject>define Subject here</subject>
  <ref_id>unique ID of an existing Reference in the Project</ref_id>
  <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
  <parent> add parent UID here. Add as many lines like this as there are parents </parent>
  <child>add child UID here. Add as many lines like this as there are children</child>
  <comments>
    <comment>
      <author>User A</author>
      <content>comment content here (see also §7.3.13)</content>
    </comment>
  </comments>
  <verb>define the Verb here (see also §2.3.2)</verb>
  <capability>define the Capability here (see also §2.3.3)</capability>
  <description>description here (see also §7.5.1)</description>
</USY>
```


8.5 Mandate Object


```
<MAN>
  <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
  <subject>define Subject here</subject>
  <ref_id>unique ID of an existing Reference in the Project</ref_id>
  <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
  <parent> add parent UID here. Add as many lines like this as there are parents </parent>
  <child>add child UID here. Add as many lines like this as there are children</child>
  <comments>
    <comment>
      <author>User A</author>
      <content>comment content here (see also §7.3.13)</content>
    </comment>
  </comments>
  <reqd_readiness_lvl>integer for the Readiness Level (see §2.4)</reqd_readiness_lvl>
  <remark>An additional Remark can be added here (see §7.6.1)</remark>
</MAN>
```

8.6 Design Problem

```
<DPR>
   <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
   <subject>define Subject here</subject>
   <ref_id>unique ID of an existing Reference in the Project</ref_id>
   <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
   <parent> add parent UID here. Add as many lines like this as there are parents </parent>
   <child>add child UID here. Add as many lines like this as there are children</child>
   <comments>
      <comment>
         <author>User A</author>
         <content>comment content here (see also §7.3.13)</content>
      </comment>
   </comments>
</DPR>
```

8.7 Requirement

```
<REQ>
   <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
   <subject>define Subject here</subject>
   <ref_id>unique ID of an existing Reference in the Project</ref_id>
   <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
   <parent> add parent UID here. Add as many lines like this as there are parents </parent>
   <child>add child UID here. Add as many lines like this as there are children</child>
   <comments>
      <comment>
         <author>User A</author>
         <content>comment content here (see also §7.3.13)</content>
      </comment>
   </comments>
   <verb>define the Verb here (see also §2.3.2)</verb>
   <capability>define the Capability here (see also §2.3.3)</capability>
   <par_type>name of the Parameter type</par_type>
   <par_unit_symbol>symbol of the Parameter type</par_unit_symbol>
   <par_value>value of the Parameter as real</par_value>
   <par_target_int>0:‘minimal’, 1:‘<’, 2:‘<=’, 3:‘=’, 4:‘>=’, 5:‘>’, 6:‘maximum’</par_target_int>
   <par_weight>importance of the Parameter (0-100, see also §7.7.7)</par_weight>
   <par_target_ref_only>Parameter Target Text for reference only</par_target_ref_only>
</REQ>
```

8.8 Option

```
<OPT>
   <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
   <subject>define Subject here</subject>
   <ref_id>unique ID of an existing Reference in the Project</ref_id>
   <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
   <parent> add parent UID here. Add as many lines like this as there are parents </parent>
   <child>add child UID here. Add as many lines like this as there are children</child>
   <comments>
      <comment>
         <author>User A</author>
         <content>comment content here</content>
      </comment>
   </comments>
</OPT>
```

8.9 Argument

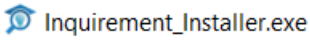
```
<ARG>
   <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
   <subject>define Subject here</subject>
   <ref_id>unique ID of an existing Reference in the Project</ref_id>
   <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
   <parent> add parent UID here. Add as many lines like this as there are parents </parent>
   <child>add child UID here. Add as many lines like this as there are children</child>
   <comments>
      <comment>
         <author>User A</author>
         <content>comment content here (see also §7.3.13)</content>
      </comment>
   </comments>
   <validates>true or false (see also §7.11.1)</validates>
</ARG>
```

8.10 Evidence

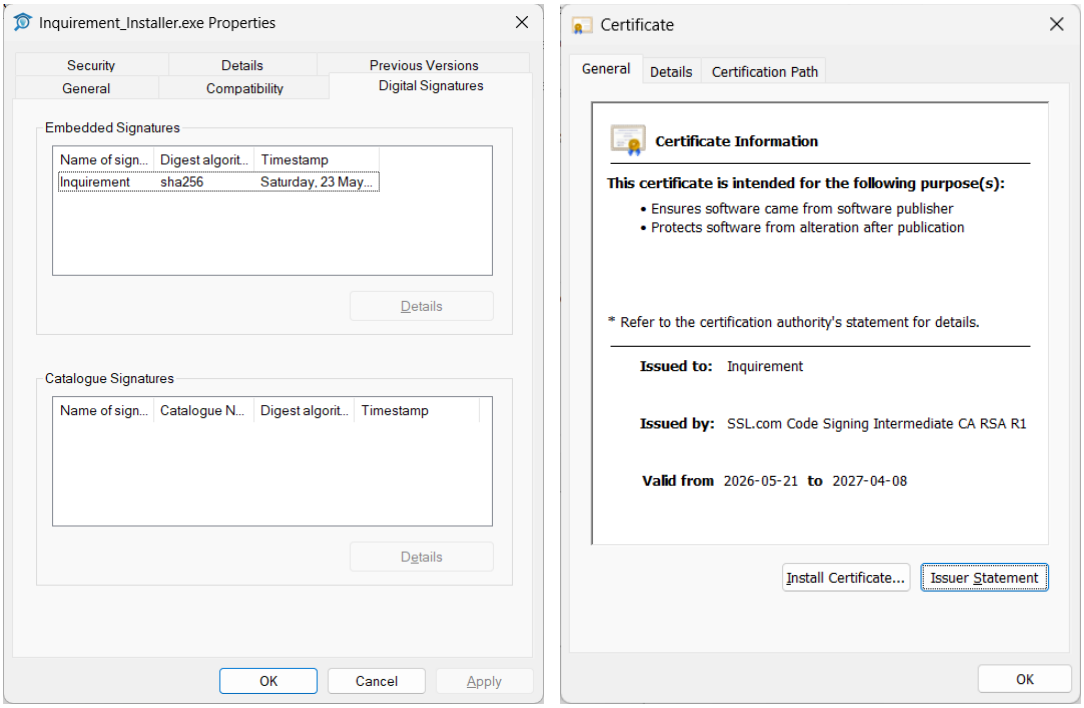
```
<EVD>
   <uid>UID of the object (see also §4.2.5 and §7.3.11)</uid>
   <subject>define Subject here</subject>
   <ref_id>unique ID of an existing Reference in the Project</ref_id>
   <tooltip_txt>any tooltip text here (see also §7.3.13)</tooltip_txt>
   <parent> add parent UID here. Add as many lines like this as there are parents </parent>
   <comments>
      <comment>
         <author>User A</author>
         <content>comment content here (see also §7.3.13)</content>
      </comment>
   </comments>
   <verb>define the Verb here (see also §2.3.2)</verb>
   <capability>define the Capability here (see also §2.3.3)</capability>
   <par_type>name of the Parameter type</par_type>
   <par_unit_symbol>symbol of the Parameter type</par_unit_symbol>
   <par_value>value of the Parameter as real</par_value>
   <par_target_int>0:‘minimal’, 1:‘<’, 2:‘<=’, 3:‘=’, 4:‘>=’, 5:‘>’, 6:‘maximum’</par_target_int>
   <par_weight>importance of the Parameter (0-100, see also §7.7.7)</par_weight>
   <par_target_ref_only>Parameter Target Text for reference only</par_target_ref_only>
   <achieved_readiness_lvl>integer for the Readiness Level</achieved_readiness_lvl>
</EVD>
```

9 Installing Inquirement

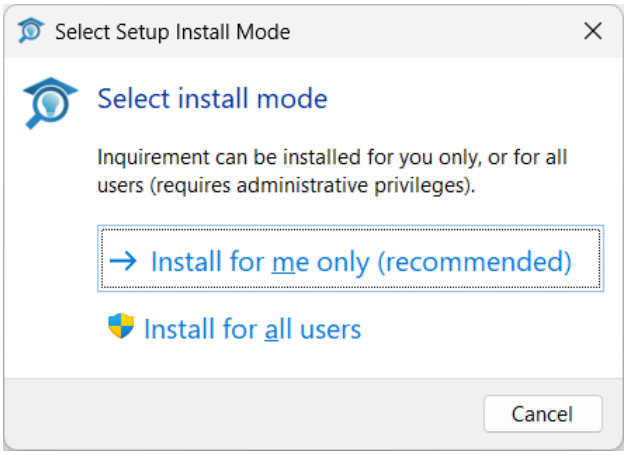
Download the Inquirement installer ‘Inquirement_Installer.exe’.



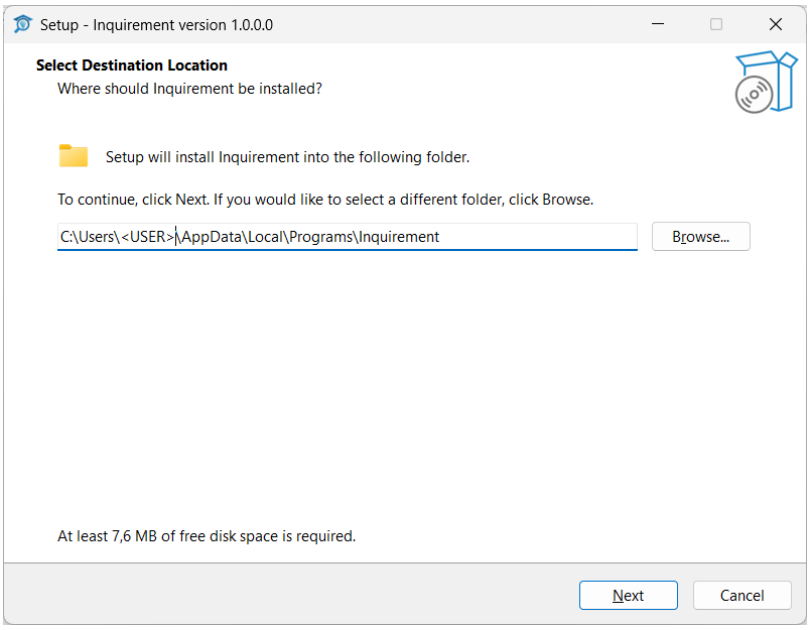
Check the signature of the installer (Right-Mouse-Button -> Properties -> Digital Signatures).



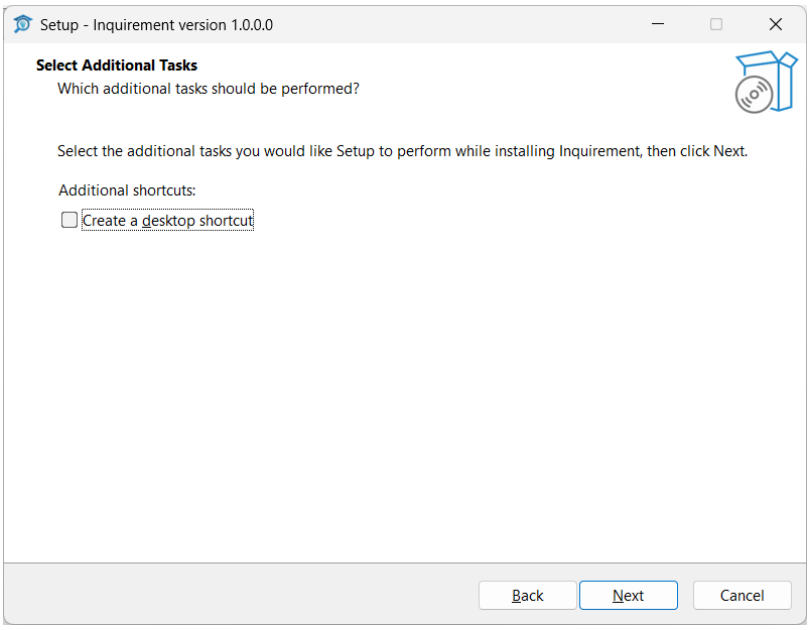
Run the installer. Select ‘Install for me only’.



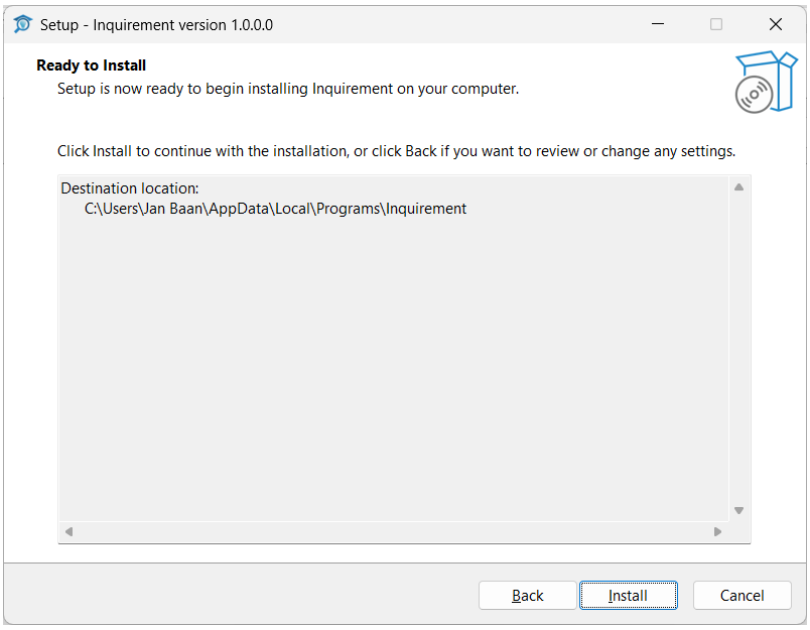
Select where to install Inquirement. Click ‘Next’.



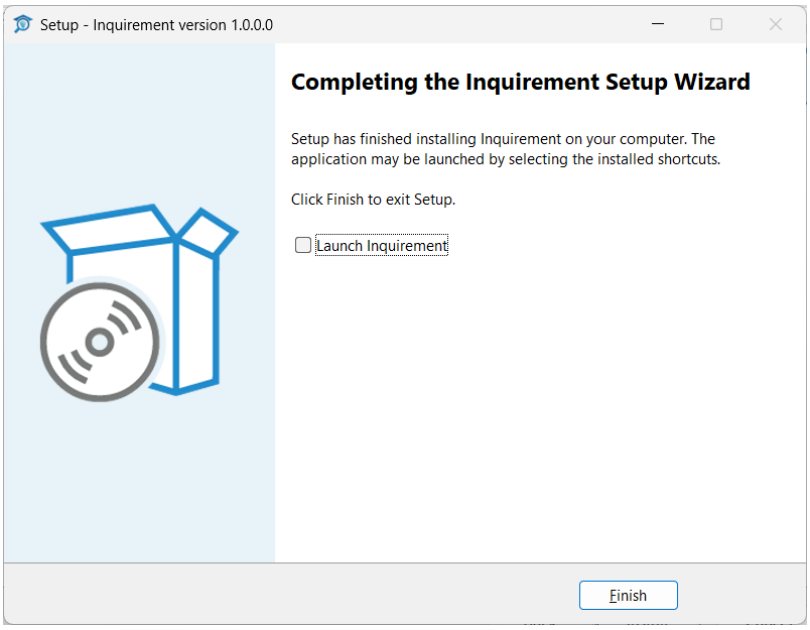
Choose whether to create a shortcut on the desktop. Click ‘Next’.



Click ‘Install’.



Select whether to launch Inquirement and click ‘Finish’.

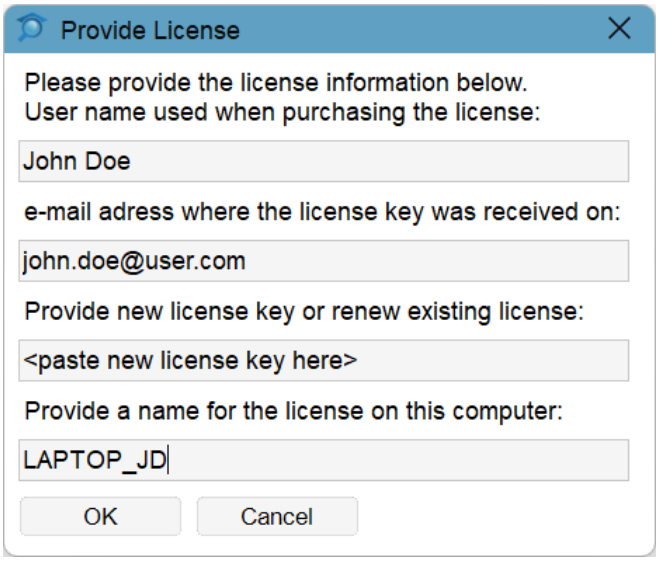


A file has been created in a new folder:

C:\Users\<USER>\AppData\Roaming\Inquirement\Inquirement_config.xml

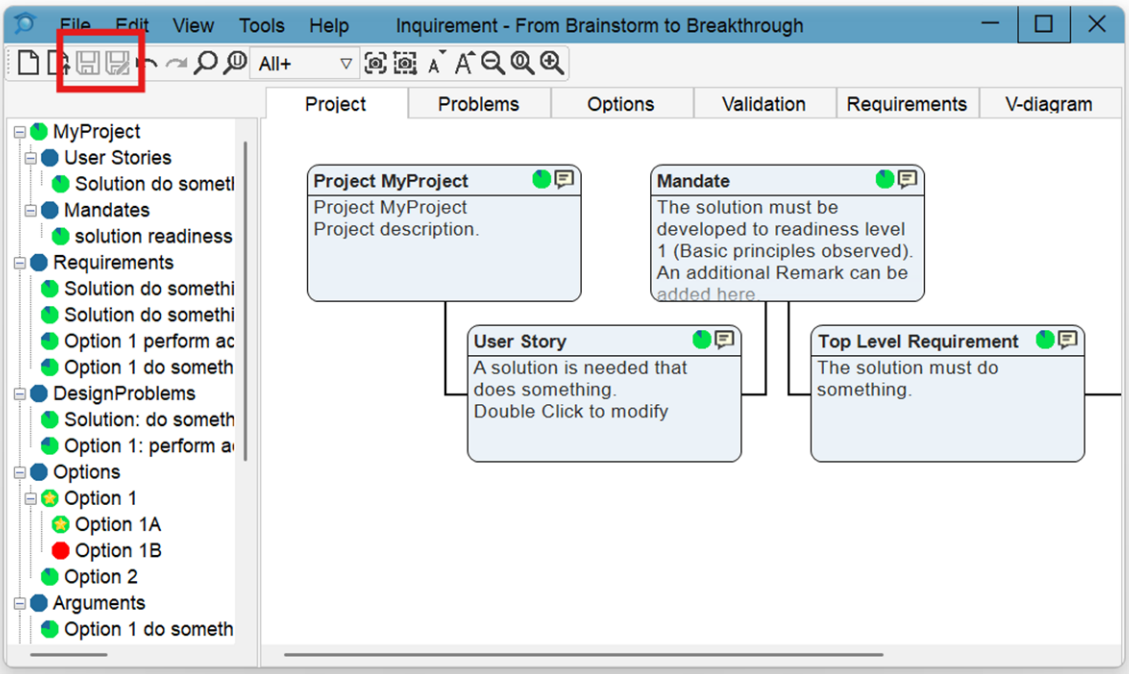
This file contains appearance settings and license information.

The window below appears if Inquirement is started for the first time or if ‘Launch Inquirement’ was selected.



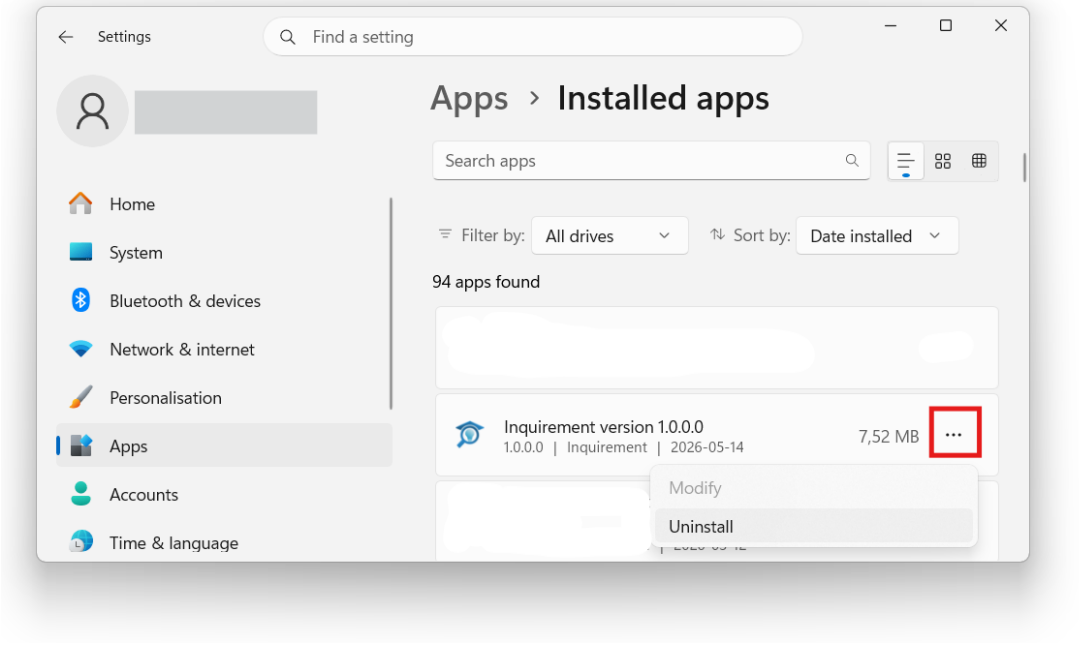
Purchase a license according to the instructions on the Inquirement website and follow the instruction in §4.4.5 ‘Renew License’ on how to include the license information.

Inquirement will start up without a license if ‘Cancel’ is pressed. All options and the full functionality is available, except the ‘Save’ and ‘Save As’ functions.

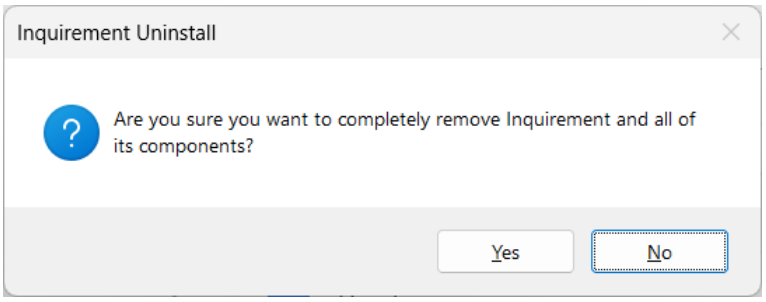


10 Uninstall Inquirement

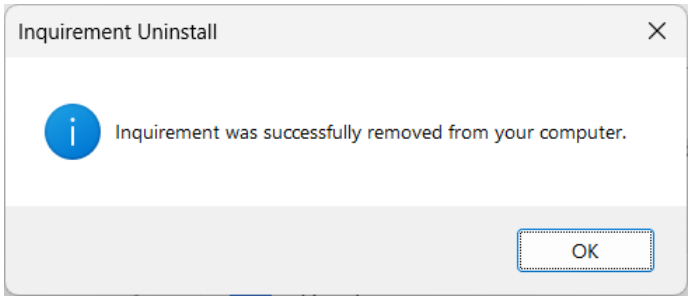
An uninstaller is available. Find it in Settings -> Apps -> Installed Apps. Press the three dots '⋯' on the right and select 'Uninstall'.



Press 'Yes' when asked to confirm the removal of Inquirement.



A confirmation will be shown if the uninstall was successful.



C:\Users\<USER>\AppData\Roaming\Inquirement has been removed.

11 Versions

Version	Date	Description
V1.0.0.0	2026-05-23	Initial version
V1.0.1.0	2026-06-03	- Added file locking functionality. - Hide Parameter Value and Unit if it represents a Goal Requirement in the input Window. - Solved Validation tab page error when more than three Sub-Option Objects with Argument Objects exist. - Solved that File Menu -> Open causes loss of license information. - Solved that changing a (Goal Requirement) Parameter does not update which Option is optimal. Also the pop-up to inform which Requirements need change is added to the 'Change the Quantification' and 'Double Click' menu items.
V1.0.2.0	2026-06-05	- Improved process to Open locked files read-only. - Requirement Object -> Allocate Requirement now only handles children of Option and Requirement Object itself. - Argument Object -> Add Requirement process now also takes target type, value and unit into account. - Solved rare case that Object Filter causes error. - No Readiness Level Icon in Objects that have no children (except for Evidence Objects) to increase distinction.
V1.0.3.0	2026-07-08	- Inquirement attempts to save all data if an unexpected error occurs. - Differentiate between create and validate URL for the LemonSqueezy license check.
V1.0.4.0	2026-08-02	- Removed explain objects example project from opening at start-up. - Improved paste behaviour of text boxes - Now always gets the current Windows username. It is essential to get this because it is used to encrypt and decrypt license data. - The file menu now shows the recently opened and saved files.
v 1.0.5.0	2026-08-12	- Does not ask to save a new project if no modifications have been done yet. - Solved error with parameter value after changing the value using double-click. - Removed description from inputs to be given to a new Evidence. The field input ended up in the tooltip text. - Open file and create new Project now switches automatically to the Project tab-page. - Links between Design Problems and Requirements do not allow mixing of Integration Requirements (either due to sub-Options or due to multiple Design Problems) and normal Requirements anymore. - Solved issue with update after adding or removing a Requirement to a Design Problem. - Only the Options in the Tree View are expanded by default. This is more convenient for large datasets.

Version	Date	Description
v 1.0.6.0	2026-08-14	- Determination of the optimal Option now looks for Goals in the Top Level Requirements if there are no Goals in the Design Problem. - Solved issue where image save dialogue was still visible in the image when making a screenshot. - Solved issue with opening door example project. Some potential Design Problems were not found and were therefore refused as child Design Problems for a Requirement. - Solved issue with unlinked subject in some cases (especially when opening a file and when doing undo or redo). - Show why window sizes better around the objects. Especially if scrolling is needed.
v 1.0.7.0	2026-08-15	- Inquirement will open the folder of the most recent file. - The expand state of the Main Tree View is now stored and reapplied in case of an update and when the Project is saved. - Example projects have been transferred to C++ code and tests functions have been added for them
v 1.0.8.0	2026-08-20	- Extended the Gate Example Project - Integration Requirements can now be mixed with non-Integration Requirements as input for Design Problems - Integration Requirements must only be managed by the Inquirement program. They can no longer be changed manually. Context menu items and double click functionality are removed for Integration Requirements. - Options are added above Integration Requirements on the Requirements tab page. This makes sure that the Requirements levels and Argument levels do not mix, since without the added Option the Integration Requirement would be a direct child of another Requirement. - Added Find all comments' menu in the 'Edit Menu'. - Improved the Form for Show or Hide of Requirements in the V-diagram tab page. - Any custom Parameters are deleted when creating a new file. - Selecting another Option in the Validation tab page now resets the scroll position (it goes to the upper right corner of the tab page).

Version	Date	Description
v 1.1.0.0	2026-09-23	- Added functionality to add references to each object - Added functionality to copy paste a tree of objects. The 'show uid' functionality has been removed since 'copy object' copies the UID to the clipboard which basically does the same. - The AI answer is now capable of adding new Parameter Types and units if the quantification of a new Requirement cannot be expressed by existing Parameter types. - A menu item is added that shows the existing References in the Project. - The default Parameter changed from price to functional. - Solved issue with undo/redo of added custom units. - Small improvements to the ReqIF export functionality. - Comboboxes now accept paste of text from the clipboard through CTRL-V - Comboboxes now limit their size to maximum 11 lines. A scrollbar appears if more lines are available. - Made the application more efficient by changing a list (c_deep) to an unordered_map (nd_map) in the code.
v 1.1.1.0	2026-10-03	- Additional improvements on the ReqIF export functionality - Added SysML export functionality

12 Privacy policy

Below is a summary of our privacy policy. Please visit our website for the complete policy.

At Inquirement, we believe your thoughts, requirements, and know-how belong to you.

Your privacy and confidentiality is important to us. The Inquirement application does not send any information through any network connection except in the following cases:

- If you open or save a file on a network location. This uses the standard Windows SaveFileDialog function.
- The Inquirement application will try to open www.inquirement.com if you click the website link.
- The Inquirement application will try to send an e-mail to info@inquirement.com if you click the e-mail link.
- The Inquirement application will regularly verify the license key at Lemon Squeezy. Please visit https://www.lemonsqueezy.com/privacy for their privacy policy.

Below is a summary of our privacy policy. Please go to our website for more information on the Inquirement privacy policy.

Privacy Policy: The "Local-First" Commitment

Our software is designed around a simple principle: we never need to see your project data in order for the product to work.

---

1. The "No-Cloud" Guarantee

Inquirement is a desktop application, not a SaaS platform. We do not host or store your project data on our servers.

Local Storage: All requirements, nodes, relationships, and project files are stored locally on your own computer or storage device.

No Access to Your Data: Because your project data never reaches our servers, we cannot read, scan, analyze, or sell it.

Backups: You are responsible for managing your own backups. You can use services such as OneDrive, Dropbox, iCloud, external drives, or any backup solution of your choice.

---

2. Data We Do Collect (The "Bare Minimum")

To operate our business and provide support, we only collect the minimum amount of information necessary.

Identity & Billing Information

We collect your name and email address through our payment provider, Lemon Squeezy, so we can:

- Send your license key
- Process receipts and invoices
- Provide customer support

License Validation

When you activate the software, Inquirement performs a one-time license validation request to verify your key.

You may install Inquirement on as many devices as you need. Hardware fingerprinting or permanent device tracking is not required.

Support Communications

If you contact us for support, we may retain your correspondence to help resolve future issues and improve customer service.

Marketing Communications

If you opt in, we may send occasional marketing or promotional emails about product updates, features, or announcements.

You can unsubscribe from marketing emails at any time.

Newsletter

If you subscribe to the Inquirement newsletter, we use your name and email address to send news, updates, and product information.

Feedback Requests

We may contact users to request feedback about their experience using Inquirement.

---

3. Third-Party Processors

We intentionally keep our technology stack small to protect your privacy.

Merchant of Record

Lemon Squeezy handles payments, taxes, Value Added Tax (VAT) processing, and billing operations.

Their services are governed by their own privacy and General Data Protection Regulation (GDPR) compliance policies.

Analytics

Inquirement does not include any in-app tracking or analytics.

We do not collect usage telemetry, behavioural analytics or project content from within the application.

---

4. Your Rights Under GDPR

As a privacy-first and EU-friendly product, we respect your rights under the General Data Protection Regulation (GDPR).

Right to Access & Erasure

You may request:

- A copy of the personal information we hold about you
- Deletion of your personal information

In practice, this is typically limited to your email address, billing records, and support correspondence.

Data Portability

Your project data remains stored locally on your machine in XML format.

You are free to move, export, back up, or transfer your files at any time without requiring our permission or involvement.

---

5. Contact

If you have questions about this Privacy Policy or your personal data, please contact:

info@inquirement.com

13 License Agreement

The Inquirement application uses the standard license agreement from our Merchant of Record Lemon Squeezy. Please visit lemonsqueezy or Inquirement for more details.

14 How to report a bug

Please report a bug by sending an e-mail to info@inquirement.com.

15 Known Issues

There are no known issues at the moment. Please send any issues you encounter when using Inquirement to info@inquirement.com.

Notice:

The Windows username is used to encrypt and decrypt Inquirement license data. This means that license data has to be provided again if the Windows user name changes.

16 Inquirement community

We are working on setting up a community for Inquirement users. Please check our website for the latest updates.