Analyzing dependencies among elements

Before exporting or sharing a package as an independent used project, you need to identify and solve package dependencies first. More specifically, you have to make sure that the package does not depend on any external elements (except other used projects). Furthermore, cyclic dependencies between several used projects are not allowed.

There are three types of dependencies:

- Package dependencies by relationship
- Dependencies by reference
- Diagram dependencies

Related pages

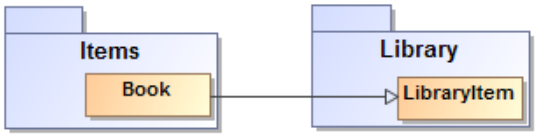
- Identifying Package Dependencies
- Package Dependencies panel
- Unresolved dependencies

Package dependencies by relationship

The used project depends on the external elements which have relationships with the model elements from the used project packages.

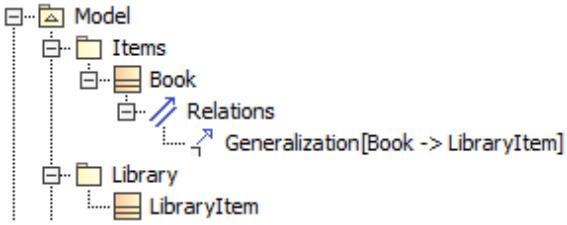
The used project depends on external elements

If a used project's element has a relationship with an external element and that relationship is contained in the used project package, an error message appears when exporting the used package.

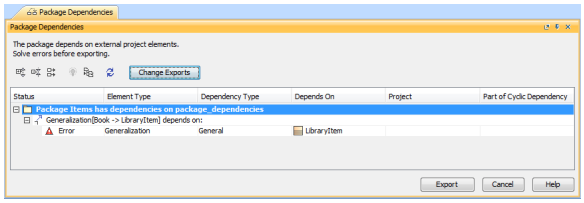


Example of the used project dependency on an external element

Such dependencies on external elements are displayed in the Model Browser:

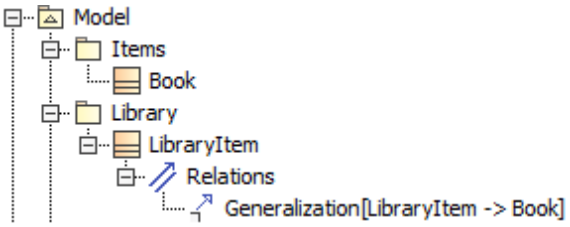


Package depends on an external element



Error in the Package Dependencies panel

In this situation, a modeling tool can suggest moving the relationship into the parent package of this external element. For example, the package *Items* is a parent of a class *Book*, so the relationship can be moved from the *Items* into the package *Library*:

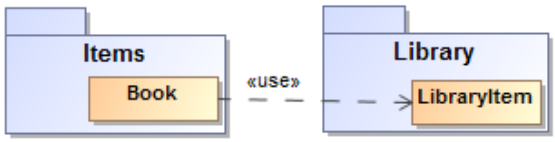


Resolved package dependency on an external element

Some movements can be achieved by clicking **Solve** in the **Package Dependencies** panel.

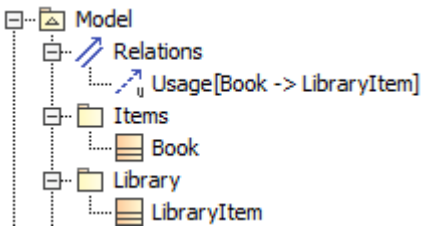
The used project depends on an external element, but can be exported (with warning)

Even though the used project element has a relationship with an external element, that relationship is contained in an external package:



Example of a “legal” used project dependency on an external element

In this case, the dependency on an external element is displayed in the Model Browser:



“Legal” used project dependency on an external element in the Model Browser

The package can be exported as the used project because the relationship is contained in an external package.

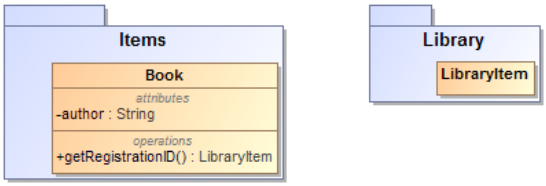
The used project does not depend on an external element

If the used project element has a relationship with an external element it is irrelevant in the context of UML (for instance, the external model uses the used project, but not vice versa) and that relationship is contained in an external model, the package can be exported into an independent used project:

Example of a relationship when the used project does not depend on an external element

Dependencies by reference

The used project depends on the external elements which have references to the model elements from the used project packages.



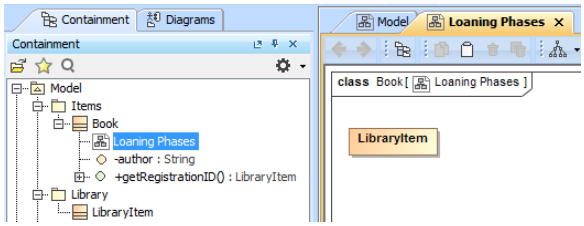
Example of a dependency by reference

In this case, the package *Items* cannot be exported to an independent project because this package has reference to the *Library* package.

Diagram dependencies

The diagram depends on all model elements displayed within it.

If the diagram is contained in the package *Items* and depends on external elements, this package cannot be exported to a separate project.



Example of the relationship when a diagram depends on an external element

For more information about the package dependencies on external elements, see The used project depends on external elements.

In this case, if the diagram is not important to the used project, it can be moved from the used project package into any external package by dragging-and-dropping it within the Model Browser:

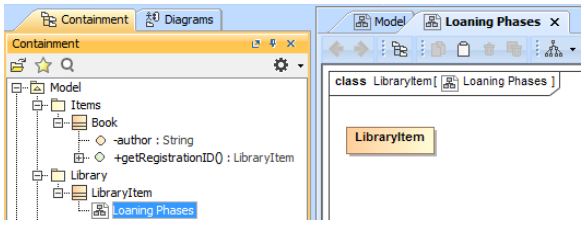


Diagram is moved from the package *Items* to the package *Library*

The package *Items* can now be exported into an independent project.

Teamwork Cloud is an ideal solution for a collaborative work on the same project. For more information about using a modeling tool in the collaborative environment, see Collaborative modeling.