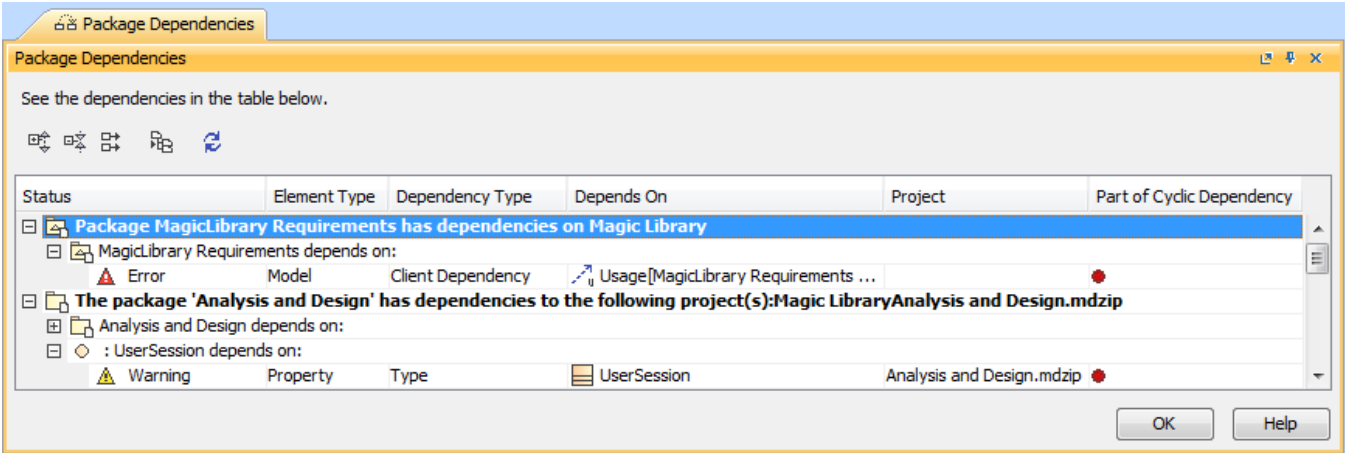


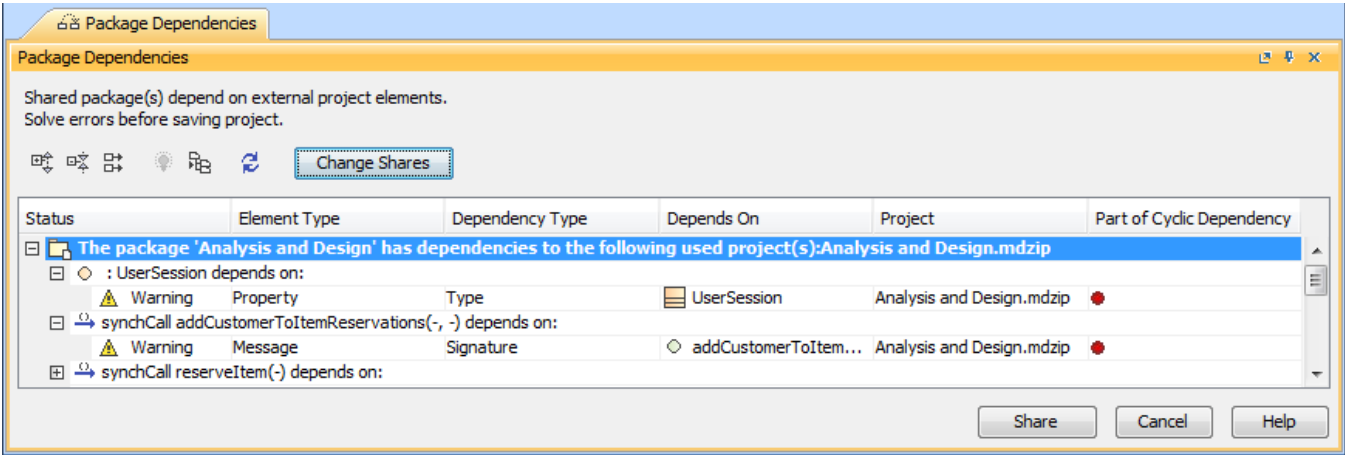
Package Dependencies panel

Before exporting or sharing a package as a independent used project, you need to identify and analyze package dependencies first. The **Package Dependencies** panel will help you with that. The **Package Dependencies** panel is displayed when you select to [share](#) or [export](#) packages.

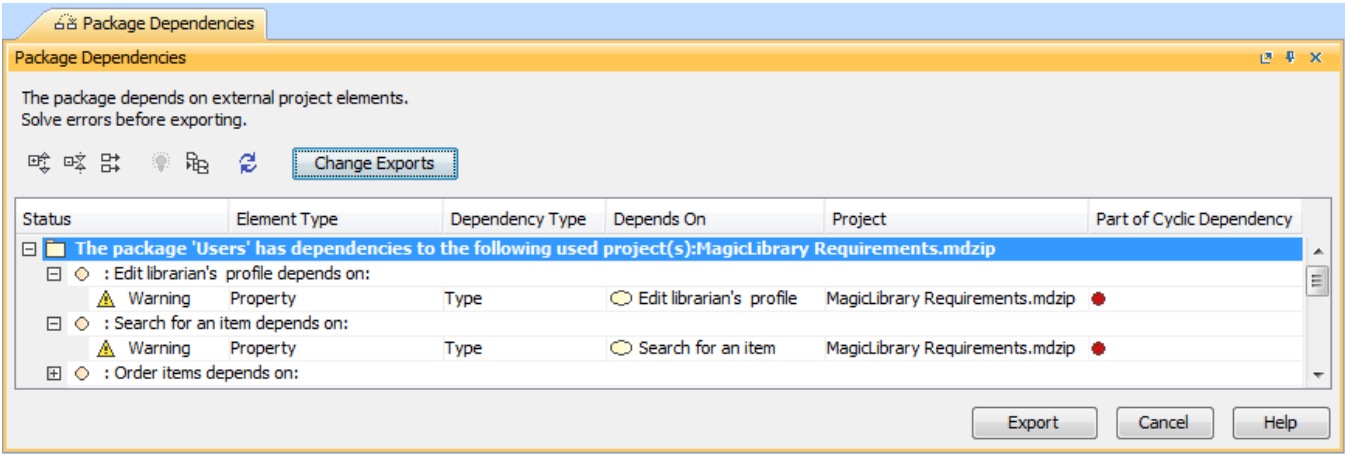
Before hiding a package, you can identify and analyze package usages and usages of all elements in the package first. The **Package Usage** window will help you with that. The **Package Usage** panel is displayed when you choose to check usages of elements in the selected packages [for hiding](#).



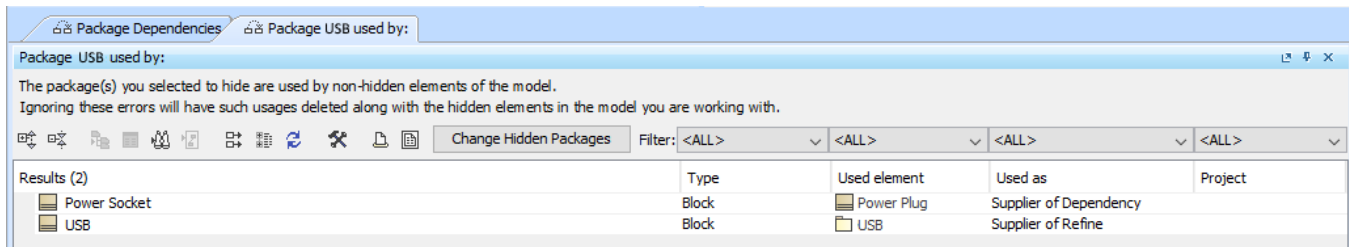
Package Dependencies panel



Package Dependencies panel when checking dependencies on package sharing



Package Dependencies panel when checking dependencies on package exporting



Package Usage search window when checking usages on package hiding

The **Package Dependencies** panel has a table which shows the list of dependencies and buttons for managing data displayed in this table.

Button	Description
Expand All Tree Branches	Expands all nodes in the package dependencies tree.
Collapse All Tree Branches	Collapses all nodes in the package dependencies tree.
Display the Full Path of the elements	Displays the element full path next to the element name.
Solve	The button is enabled, when a dependency with Error status is selected in the table and a solution for the problem can be found. Clicking the button opens the dialog for choosing the solution for a specific dependency problem. This button is only visible when exporting or sharing a package.
Select in Containment Tree	Shows the selected element in the Browser. The button is available, when a dependency is selected in the table.
Refresh	Performs dependency analysis and refreshes the dependency table with the new analysis results.
Change Shares	Opens the Shared Packages dialog wherein you can select another packages to be shared. This button is visible only when sharing a package.
Change Exports	Opens the Export Package to New Project dialog wherein you can select another packages to be exported. This button is visible only when exporting a package.
Share	Closes the Package Dependencies panel and makes the package shared. The button is available, when the Check Dependencies on Package Export/Sharing environment option is set to <i>Allow dependencies</i> (to open the Environment Options dialog, select Options > Environment). This button is visible only when sharing a package.
Export	Closes the Package Dependencies panel and opens the Save as/ Commit Settings dialog for saving/committing the package as separate project. The button is available, when the Check Dependencies on Package Export/Sharing environment option is set to <i>Allow dependencies</i> (to open the Environment Options dialog, select Options > Environment). This button is visible only when exporting a package.

Column	Description
Status	Shows severity of the element dependency problem. The status can be Error , Warning , or Info . Dependencies that have Error status: <ul style="list-style-type: none"> used project depends on the project Dependencies that have Warning status: <ul style="list-style-type: none"> cyclic used project depends on other used projects Dependencies that have Info status: <ul style="list-style-type: none"> project element depends on elements from shared packages (shared packages can belong both to the project and external project)

Element Type	Displays element type.
Dependency Type	Displays dependency type.
Depends On	Displays the model element, on which the package/model element is dependent.
Project	Displays the name of the used project file that owns the model element, on which the package/model element depends.
Part of Cycling Dependency	Indicates that the element is a part of the <i>cycling dependency</i> *.



*** Cyclic dependency**

There is a chain of dependencies such that $A \rightarrow M(1), M(1) \rightarrow M(2), M(2) \rightarrow M(3), \dots, M(X) \rightarrow M(A)$, where:

- A is an element from the used project M(A)
- M(1..X) are other used projects
- [Identifying Package Dependencies](#)
- A M(x) is element A dependency on used project M(x)
- [Analyzing Dependencies](#)
- [Unresolved dependencies](#)
- M(y) M(x) is a dependency of at least one element in used project M(y) on used project M(x),

Related pages

then this chain is called a cyclic dependency and every atomic dependency in this chain is considered as part of cyclic dependency.