Understanding differences in local and server project structure

The structure of your model may change when switching back and forth between local and server projects. Usually, this happens when the local project is added to the server, the server project is saved locally, or the server project is updated with local changes.

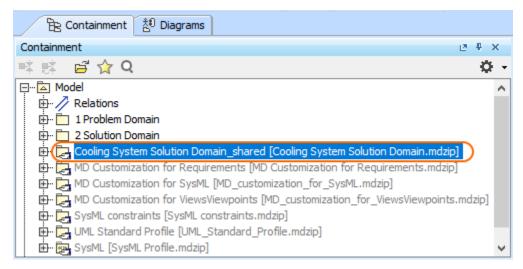
From Teamwork Cloud to local project

• When the Teamwork Cloud project containing used projects is saved locally, packages from used projects are mounted under the model root (

Model) and marked with the 🖾 (shared package) symbol. The original model structure (as it is in the server project) cannot be maintained due to model architecture limitations, i.e., the absence of the dedicated root node for used projects.

Be Containment 참 Diagrams	
Containment	@ # X
喀 縣 🖻 🏠 Q	ф
P. Model	~
En Relations	
🕀 🔁 1 Problem Domain [VehicleCCU Problem.mdzip]	
🕀 🔁 2 Solution Domain [VehicleCCS Solution.mdzip]	
 Problem Domain [VehideCCU Problem.mdzip] 2 Solution Domain [VehideCCS Solution.mdzip] 2 Solution System Solution Domain [Cooling System Solution.mdzip] 	
🖶 📴 2.2 Heating System Solution Domain [Heating System Solution.mdzip]	
🕀 📴 2.3 Control System Solution Domain [Control System Solution.mdzip]	
 2.2 Heating System Solution Domain [Heating System Solution.mdzip] 2.3 Control System Solution Domain [Control System Solution.mdzip] 2.4 Sensors System Solution Domain [Sensors System Solution.mdzip] 	
⊕ 🔁 2.5 UI System Solution Domain [UI System Solution.mdzip]	~

 In the local environment, only shared packages will be visible in the used project. Therefore, if the used project contains elements under the model root, a new package <userProjectName>_shared is created when you save the project to make it possible to share those elements.



From local project to Teamwork Cloud

Mount relationship

• The Mount relationship may be created when the server project containing used projects is saved locally, modified, and then added back to the server.

It is used to recreate the same structure of the Containment tree as it was originally in the local project. This may be important for features that use a defined scope of the model. Only elements from used projects that are mounted under the Model root (Model) or any of its packages are considered part of the model scope (when no used projects are set as the scope).

In general, the Mount relationship enables you to create a visual copy/representation of the chosen package in any place in the model.

B Containment	
Containment	Ψ×
喀 縣 💣 🏠 🔍	ф.,
Model Relations Mount[Model -> 1 Problem Domain] Profile Application[Model -> SysML] Profile Application[¥1]
⊕ □ 1 Black Box ⊕ □ 2 White Box	
Index	

• You can delete **Mount** relationships from your project if, for example, you do not want to have duplicate packages in the Containment tree of your server project.

beeting Mount relationships may affect diagrams, elements or other options that require specifying the scope since packages from used projects will no longer be under the main project model root (they will still be available under **Project Usages**).

Course of the second se

Packages from used projects mounted under the model root or any of its packages via the Used Projects dialog are kept in their original locations by

using Mount relationships	. They are automatically created once	the project is added to the s	erver.		
Project Usages	Update from Local Project	m the local end		=	
The root node Project Usag is visible in the project.	Update your server project fro	in the local one.		$\nabla^{=}$	used project version
			- - -	Sunth	
Containment				WHITE	
📑 🗱 📑 🕁 Q	Output Options				
E Model	Local Project:			_	
🗄 🧪 🧪 Relations	C:\Users\User\Documents\2.3 Control System Solution Domain.mdzip				
🕀 🛅 1 Problem Do					
🕀 🛅 2 Solution Do	Update Branch:				
Project Usages	trunk				
🕀 🔄 Model [MD C		ed Projects 🕕			
🖻 🖻 Model [MD_c					
H Model [MD_c		Update	Cancel	Help	
🖻 🔄 Model [Requ					
	IL constraints.mdzip]				
🖶 🔄 Model [SysM					
🗄 🗄 🖓 Model [UML_	_Standard_Profile.mdzip]				
🛄 💭 Code Engineerin	g Sets				

Symbols used

Symbol	Description
a	Indicates that a package belonging to the used project is shared.
la 🛛	Indicates that a shared package belongs to the opened project.
Ē, ®, \$	Indicates that there is a package/profile/model from the used project in the Teamwork Cloud project.
Black box [Black box.mdzip]	Packages from used projects have the project name they come from written in brackets.

For additional markings that can be displayed in used projects, refer to pages Moving elements to other projects and Working with partially loaded projects.