



Navigating DevOps Model RealTime Models

*Author: Mattias Mohlin
Senior Software Architect
HCL*

NAVIGATING MODEL REALTIME MODELS.....	1
INTRODUCTION.....	3
PROJECT EXPLORER NAVIGATIONS.....	4
NAVIGATION BY TYPING.....	4
FILTER BOX.....	4
NAVIGATE TO RESOURCE ROOT.....	5
SHOW CONTAINER CAPSULE/CLASS.....	6
NAVIGATE TO DIAGRAM.....	6
NAVIGATE TO FILE.....	6
OPEN SOURCE CODE.....	6
INHERITANCE EXPLORER.....	6
VIEW NESTED STATES.....	8
NAVIGATING TO RELATED ELEMENTS.....	9
NAVIGATION HISTORY.....	10
DIAGRAM NAVIGATIONS.....	11
NAVIGATE TO PROJECT EXPLORER.....	11
SHOW CONTAINER CAPSULE/CLASS.....	11
OPEN SUPER AND SUB CLASSES.....	11
Go OUTSIDE/INSIDE.....	11
NAVIGATE TO SUBSTITUTABLE TYPES.....	12
<i>Create Dependencies for a Capsule Part.....</i>	<i>14</i>
NAVIGATE TO SOURCE AND TARGET OF LINES.....	16
OPEN SOURCE CODE.....	16
SHOW TYPE IN PROJECT EXPLORER.....	16
SHOWING RELATED ELEMENTS.....	17
OUTLINE VIEW.....	17
PROPERTIES VIEW NAVIGATIONS.....	19
NAVIGATE TO CONTAINER ELEMENTS.....	19
OPEN TYPE.....	19
NAVIGATE TO C++ CODE SNIPPETS.....	19

NAVIGATE TO REDEFINED OPERATIONS.....	20
NAVIGATING CONNECTORS.....	20
NAVIGATING TO RELATED ELEMENTS.....	20
CODE EDITOR AND CODE VIEW NAVIGATIONS.....	22
NAVIGATE FROM CODE SNIPPET TO MODEL.....	22
NAVIGATE FROM CODE SNIPPET TO GENERATED FILE.....	22
NAVIGATE FROM GENERATED FILE TO MODEL.....	23
NAVIGATE FROM CODE VIEW TO CODE EDITOR.....	23
TRANSFORMATION CONFIGURATION EDITOR NAVIGATIONS.....	25
SHOW IN PROJECT EXPLORER.....	25
NAVIGATE TO INHERITED AND PREREQUISITE TRANSFORMATION CONFIGURATIONS.....	25
OTHER NAVIGATION COMMANDS.....	28
NAVIGATE BY URL.....	28
SEARCH VIEW NAVIGATIONS.....	28
PROBLEM VIEW NAVIGATIONS.....	29
NAVIGATING FROM CODE SENDING AN EVENT TO WHERE IT IS RECEIVED.....	29

This document describes how to use the various navigation commands that are available in DevOps Model RealTime.

The document was last updated for Model RealTime 11.1. All screen shots were captured on the Windows platform.

Introduction

An important capability of a modeling tool is the ability to let users quickly find elements of interest in the model. One way to accomplish this is through search commands. Another is by means of navigation commands. The difference between searching and navigating can sometimes be subtle. However, while search commands usually are general purpose commands for finding elements based on certain criteria, a navigation command is often more specific to a particular usage scenario where you need to find an element from the context of another (selected) element. Search commands usually find many elements and present these using the Search view. Navigation commands usually only find a single element, and present it by selecting it in the Project Explorer and/or in a diagram.

Knowing about and using navigation commands can improve your productivity quite a lot, especially when working with a big model where you may only be familiar with parts of it. This document describes the navigation commands that are provided by Model RealTime and gives some guidelines for when to use each of them.

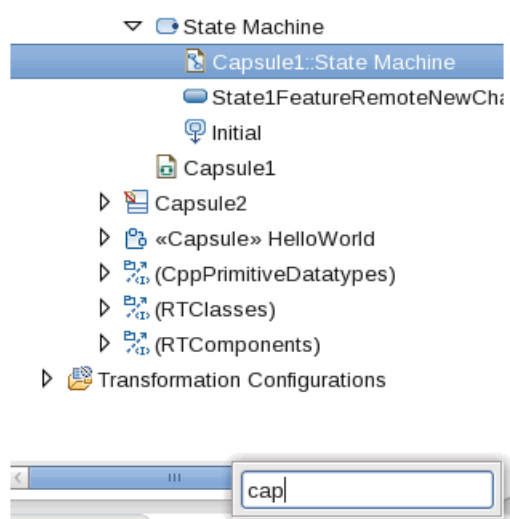
Project Explorer Navigations

Most navigation commands in the Project Explorer are available in the sub context menu called “Navigate” which is available when an element is selected in the Project Explorer. Depending on which kind of element that is selected different navigation commands are available. There are also some navigation commands in the Project Explorer toolbar, for example a navigation history feature, which makes it easy to return to previously selected elements after you have performed a navigation.

Navigation by Typing

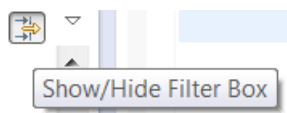
If you know the name of the element you are looking for, just type quickly the first few letters in its name and the Project Explorer selects the next element with a label that matches the typed letters. You can repeat the process until you find the element you are looking for.

Note that this navigation is provided by the underlying operating system, so it works slightly differently on Windows and Linux. On Linux a small text box pops up where you can see the letters you have typed, and there you use the up/down arrow keys for moving between different matching elements:

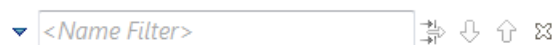


Filter Box

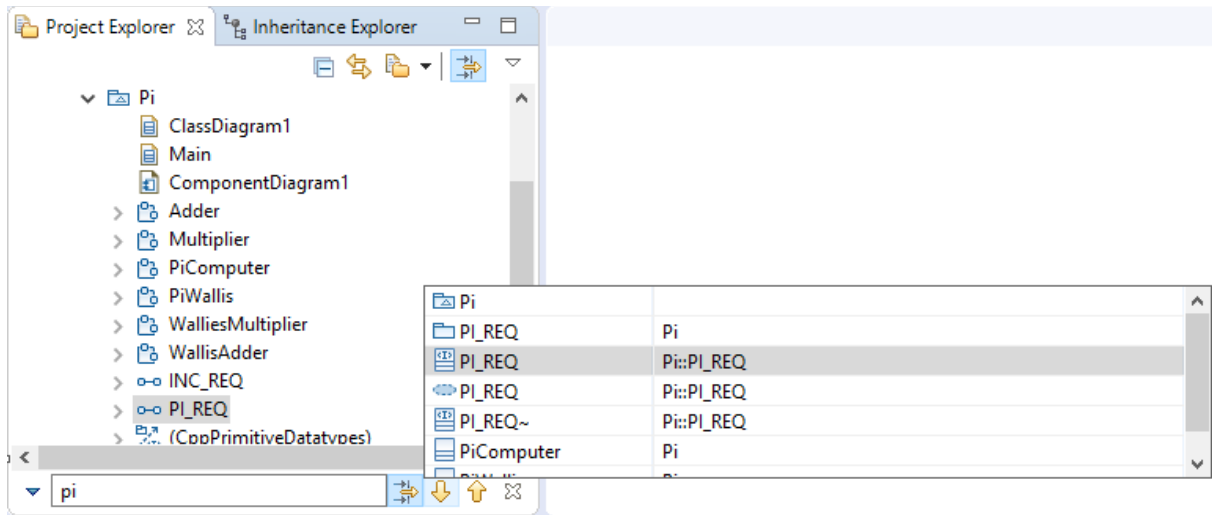
The Project Explorer filter box is useful for navigating to an element when you know approximately how it is named. The filter box is made visible by pressing the “Show/Hide Filter Box” button in the Project Explorer toolbar:



The filter box appears just below the Project Explorer :



Type a few character in the filter box and press Enter. The first matching element will be selected in the Project Explorer. You can use the "Locate next match" and "Locate previous match" buttons to proceed to the next or previous matching element. A tooltip also appears which shows all matches so you can directly navigate to the element you are looking for.



By default at most 50 elements can match the text you type in the filter box. If there are more matches either make the filter text more specific so that it matches fewer elements, or raise the limit by clicking on the triangle to the left of the filter box and select “Max Number of Matches...” in the menu that appears. Having a too high limit can cause poor performance when using the filter box, so usually it's better to instead make the filter text more specific.

Matching of the filter text is done case-insensitively against the names of model elements. It's enough that some part of the name matches the filter text.

Note that elements are found in the order in which they are stored in the model. Depending on how you have sorted the Project Explorer (see *Preferences – Modeling – Views – Project Explorer – Sort by*) this order may be different from the order in which elements are shown in the Project Explorer.

The “Filter Project Explorer content” toggle button to the right of the filter box can be enabled to filter out elements that don't match the filter text. Sometimes this makes it easier to see the matching elements in the Project Explorer.

Navigate to Resource Root

When working with SCM systems such as Git or Clearcase you often need to access the “Team” sub context menu in the Project Explorer, where most SCM commands are found. However, this menu is only available for root elements of a model file, such as a package or a class. The navigation command “Navigate to Resource Root” (available in the “Navigate” sub context menu) can therefore be useful when you want to reach the resource root element quickly from the context of a selected element which is stored in that model file.

For example, assume you are looking at an operation which seems to have been changed by someone in an incorrect way. By invoking “Navigate to Resource Root” you are taken to the

resource root element, for example the containing class. Then you can right-click on it and perform the command *Team – Show in History* to see who has recently modified the file where the class is stored.

There are also several other commands that only are available from the context of resource root elements. Some of them are navigation commands such as [Navigate to File](#).

Show Container Capsule/Class

Elements that are contained in classes or capsules have a navigation command "Show Container Capsule/Class" in the "Navigate" sub context menu. It takes you to the class or capsule that owns the selected element, directly or indirectly.

One use case where this navigation is useful is when you want to copy/paste an attribute or an operation. Select the element and copy it to the clipboard. Then perform the navigation to the container class, and paste the element. This navigation is so common that it has a default key-binding (Ctrl + Shift + C).

Navigate to Diagram

Elements that are displayed on at least one diagram have a navigation command "Navigate to Diagram" (default keybinding is Ctrl + Alt + V) in the "Navigate" sub context menu. It opens the diagram and highlights the symbol or line that corresponds to the element. If the element is shown in more than one diagram, all diagrams are listed in the Search view so you can navigate to any of them, in case the first found diagram was not the one you were looking for.

Navigate to File

By default model files are not shown in the Project Explorer, but there is a filter setting to make them visible (run the "Filters and Customization" command from the Project Explorer view menu and unmark the "UML Model files" filter in the "Pre-set filters" tab). There is a navigation command which allows you to navigate to the model file from a model element. It's located in the "Navigate" sub context menu and is called "To File". If the model file is filtered out from the Project Explorer a dialog appears which lets you turn off the filter.

Open Source Code

Many elements which are translated into source code files have navigation commands in the "Navigate" sub context menu which open these files in an editor. For C++ there are two commands:

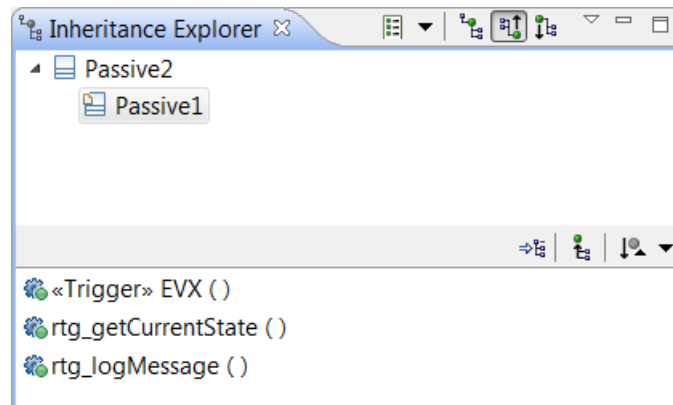
- **Open Source Code Body**
Opens the C++ implementation file (*.cpp) where the element is generated.
- **Open Source Code Header**
Opens the C++ header file (*.h) where the element is generated.

These navigation commands are typically available on elements which get translated to a "top level" element in a file, such as a class, capsule, enumeration, artifact or protocol.

Inheritance Explorer

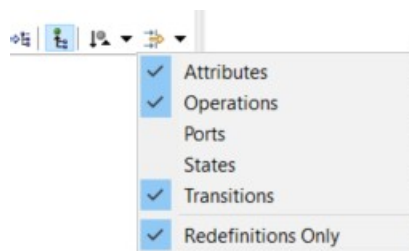
The Inheritance Explorer is a view which provides navigation in inheritance hierarchies. For example you can use it on a class to navigate to its super class or its sub classes. You can also use it on an interface to navigate to classes (or capsules) that implement that interface.

Open the Inheritance Explorer by selecting an element in the Project Explorer and then run the context menu command *Explore Inheritance*. You can also open it without a selection using the command *Window - Show View - Inheritance Explorer*.



Use the toolbar buttons to control if you want to see the super type hierarchy, the sub type hierarchy, or both. There are also toolbar buttons in the lower pane (called the Members pane) which can be used to decide if inherited members should be shown or not, and to control how to sort the member elements in that pane. You navigate to a member element or a type by double-clicking on it.

By default only attributes and operations are shown in the Members pane. Use the right-most toolbar button to control what members to show. For a capsule also ports, states and transitions can be shown, since these also are elements that can be inherited.

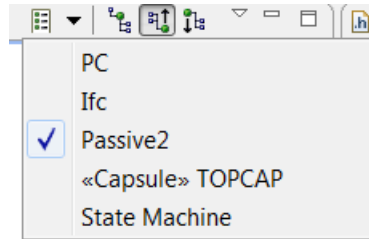


The filter "Redefinitions Only" is useful when you are interested in how a class or capsule has modified an inherited class or capsule by redefining elements. For example, for a capsule this can give you a good overview of all redefined states and transitions there are in its state machine.

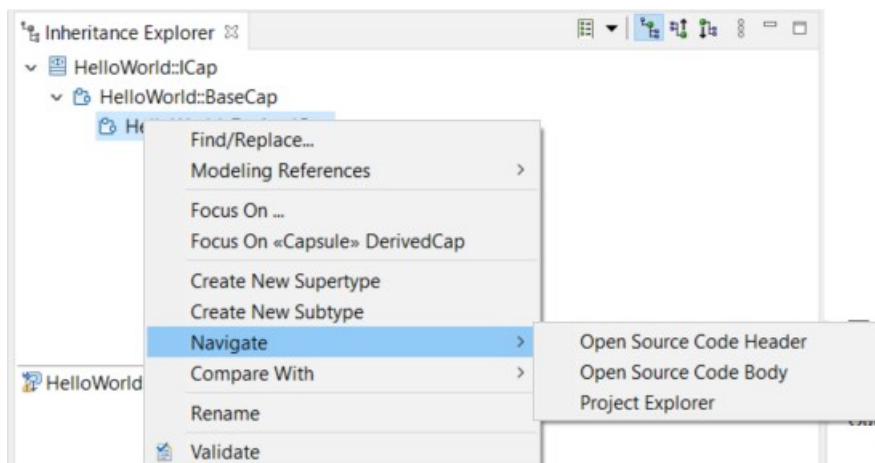
When the Inheritance Explorer opens it first focuses on the element that was selected in the Project Explorer, if any. You can later change the focus element by right-clicking on a type shown in the inheritance tree and run the command "Focus On". This also allows you to focus on a completely unrelated type, without having to first switch back to the Project Explorer. It's

also possible to switch focus element by dragging an element from the Project Explorer and dropping it onto the Inheritance Explorer.

The Inheritance Explorer keeps a list of previously selected focus elements. You can easily go back to any of these focus elements from the drop down menu under this button:



There are many other useful commands in the toolbars and context menus of the Inheritance Explorer. For example, you can navigate from an element shown in the Inheritance Explorer to the corresponding generated C++ files:



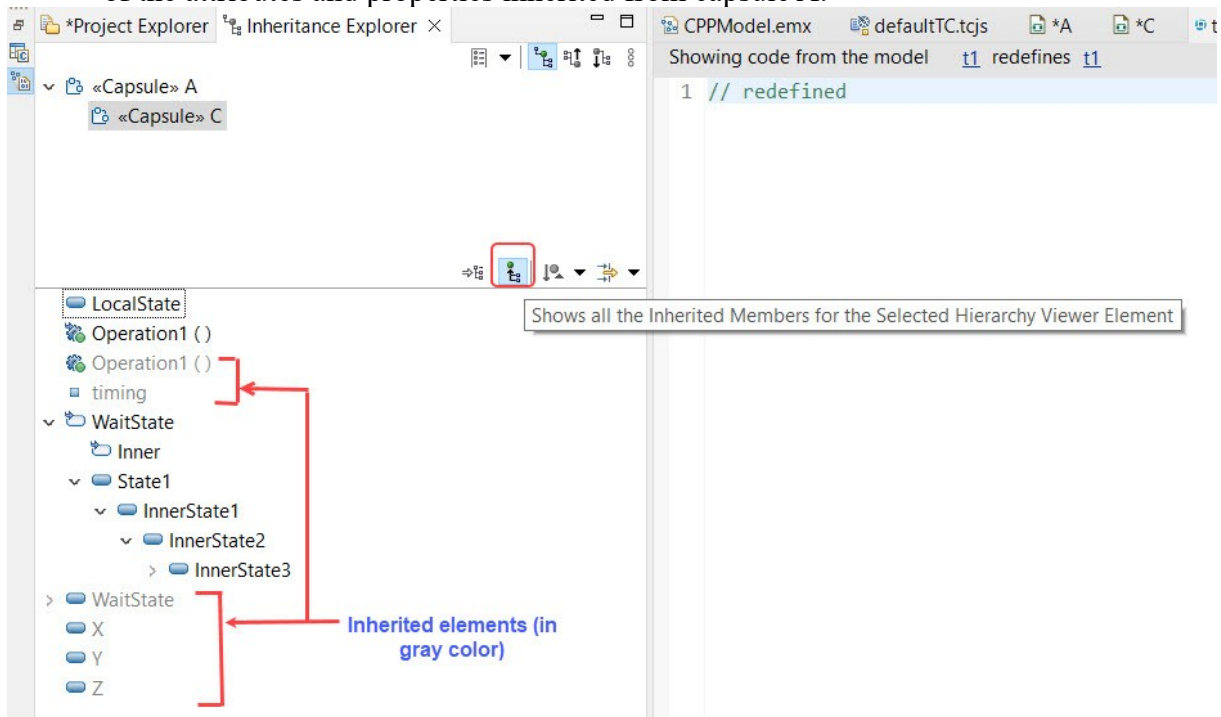
View Nested States

The *Inheritance Explorer* window also allows you to view nested states and transition entries. To gain a deeper understanding of this feature, consider a scenario where capsule C inherits attributes and properties from capsule A.

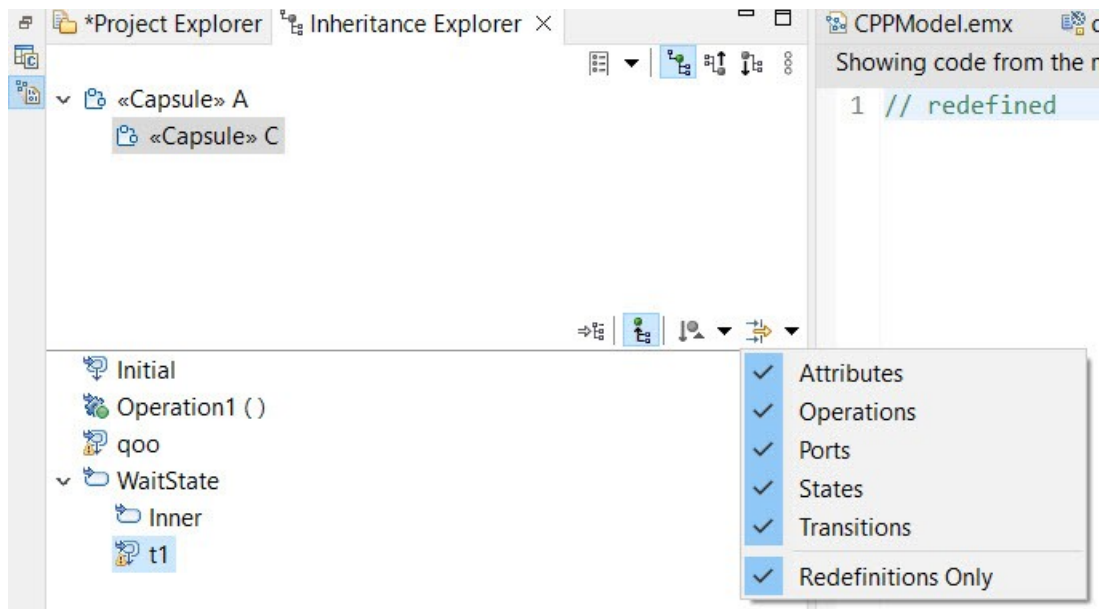
To utilize this functionality:

1. **Select the Inherited Capsule:** Begin by selecting capsule C from the *Project Explorer* to view its nested states and transitions within the *Inheritance Explorer*.
2. **Display Inherited Members:** Click the "Shows all the Inherited Members for the Selected Hierarchy Viewer Element" button. This action will enable you to see all the in-

herited members within the Inheritance Explorer, providing a comprehensive overview of the attributes and properties inherited from capsule A.



You can also use the option *Redefinitions Only* to easily view the states and transitions of an inherited state machine that have been redefined. The snapshot below depicts it better.



Navigating to Related Elements

An element in a model is usually related to several other elements, and it may be useful to be able to navigate to those related elements. For example, an attribute references a type, and you

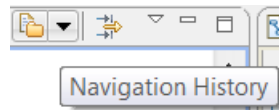
often need to navigate to that type when the attribute is selected. These navigation commands are available in the “Navigate” sub context menu.

Here are some examples of commands for navigating to related elements in the Project Explorer:

- Navigating to the type of an attribute, parameter or port
- Navigating to the source state of a transition
- Navigating to the event, port, data class or protocol of a trigger
- Navigating to the target element of a relationship such as a dependency or a generalization. (In UML this element is called the supplier of the relationship and the navigation command is therefore called “Show Supplier”)
- Navigating to association ends from a bidirectional association (shown under the special “Associations” virtual folder). These commands are called “Show End1” and “Show End2” respectively.
- Navigating from redefined or excluded states, transitions or ports to the corresponding element in the super class or capsule.

Navigation History

The Project Explorer keeps track of all navigation commands that take place. Whenever its selected element changes as a result of running a navigation command, the previously selected element is added to the navigation history. You can access the elements in the navigation history by pressing the button in the Project Explorer toolbar:



If you want to return to the previously selected element after running a navigation command you can simply press this button once. You can press it many times in order to move back to an element further back in the navigation history. You can also view all elements in the navigation history by pressing the triangle next to the button. This allows you to directly navigate to any of these elements.

The menu that appears when you press the triangle also contains a command "Add Selection to History". This allows you to manually add elements to the navigation history even without running a navigation command. This can be useful as a lightweight bookmark feature where you can add elements that you want to be able to return to easily in the future.

If the list of elements in the navigation history becomes too long you can use the command "Clear History" to remove all elements from the history.

If you prefer to access the navigation history using the keyboard you can open the preferences at *General – Keys* and define keyboard shortcuts for the commands “Navigation History Go Back” and “Navigation History Go Forward”. Then you can use these keyboard shortcuts for quickly moving backwards or forwards in the navigation history.

Diagram Navigations

Navigation commands in diagram editors are usually located in the “Navigate” sub context menu of symbols, lines or the diagram itself. There is also a special Outline view which helps navigating around within a diagram.

Navigate to Project Explorer

From each symbol or line in a diagram you can find out which model element it represents by running the context menu command *Navigate – Project Explorer*. This command can also be used on diagrams themselves by right-clicking in the diagram background, in order to navigate to the diagram in the Project Explorer. Some diagrams (for example state charts) have a frame that encloses the diagram content, and if you perform the navigation on that frame, you will instead find the model element that owns the diagram (for example a state machine).

Show Container Capsule/Class

This command is available in the “Navigate” sub context menu. It locates the model element of the selected symbol, line or diagram and then works the same as [Show Container Capsule/Class](#) in the Project Explorer.

Open Super and Sub Classes

When working in composite structure or state chart diagrams of capsules or classes that are inherited it's often useful to be able to navigate the inheritance hierarchy from these diagrams. To navigate to the super class run the command *Navigate – Open Superclass* and to navigate to the sub class run the command *Navigate – Open Subclass*. These commands are only available if the capsule or class has a super or subclass. If there are many subclasses a dialog will let you choose which one to navigate to.

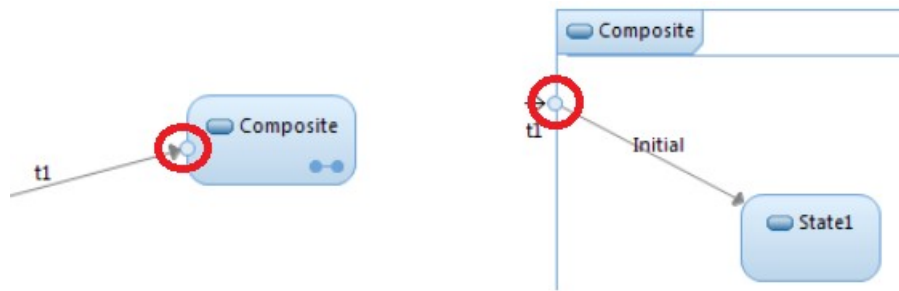
This navigation command opens the current diagram as it looks in the super or subclass. This for example means that you can quickly find out which elements in a super class state chart diagram which subclasses have redefined. If you have a particular symbol or line selected when running the command, that same symbol or line gets selected in the diagram that opens.

Go Outside/Inside

These commands are available in the “Navigate” sub context menu and can be used on part symbols in a composite structure diagram and on state symbols in state chart diagrams. They are used for navigating in the composite hierarchies of capsule parts and states. Use “Go Inside” (default keybinding is F3) to navigate to the state chart diagram of a composite state, or the composite structure diagram of a capsule that types a capsule part. In the same way, use “Go Outside” to navigate in the other direction. Note that for composite structures it's possible to have multiple capsule parts that are typed by the same capsule. Therefore a dialog may appear in this case to let you choose which capsule part to navigate to.

When using these commands for navigating in a hierarchical state machine it's useful to select an entry or exit connection point on a state. Performing “Go Outside” with a connection point selected on the state chart diagram frame will open the enclosing state chart diagram and select the same connection point on the state symbol. This makes it easier to follow a compound transition path out to the connected triggered transition. In the same way it's possible to select

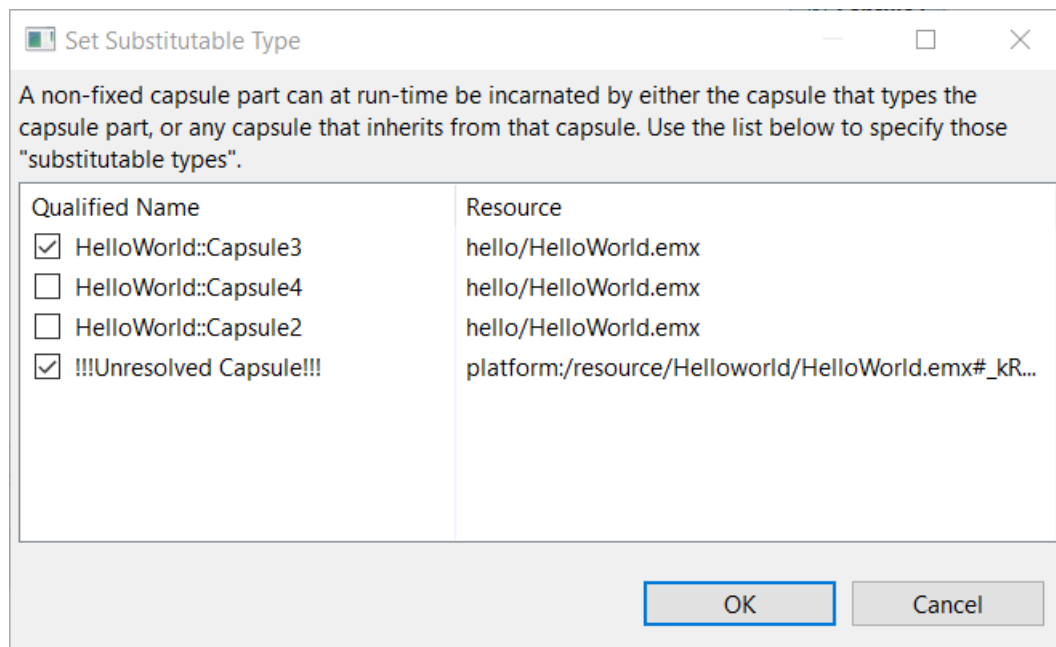
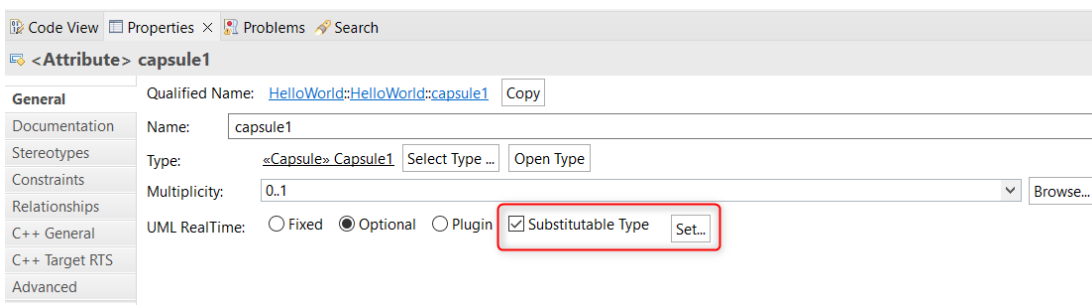
a connection point on a state symbol and do “Go Inside” to have the same connection point selected on the state chart diagram frame of the composite state.



Navigate to Substitutable Types

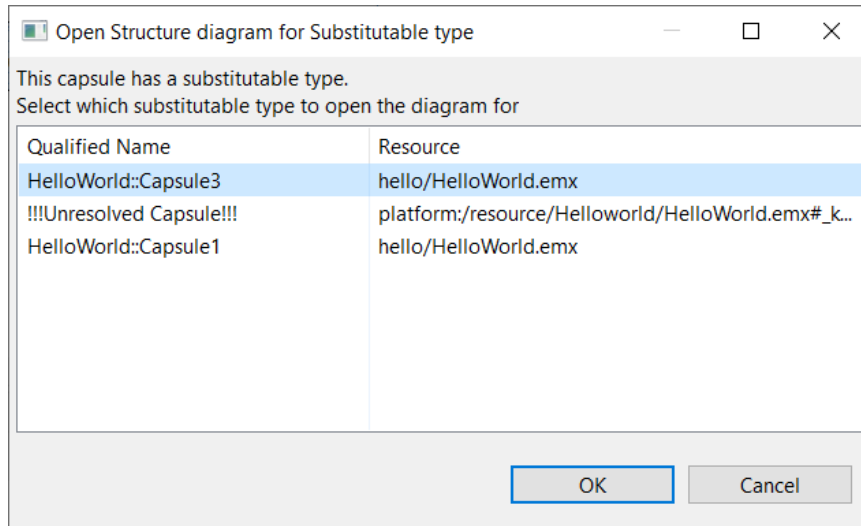
An optional capsule part will be incarnated during runtime using a specialized capsule that inherits from the abstract capsule. From a technical perspective, any concrete capsule inherited from the abstract capsule can be incarnated into the capsule part. However, in practice, it is usually known that only some concrete capsules will be incarnated. Model RealTime, as a modeling tool, provides a way to specify these specialized capsules.

This can be done by when the “Substitutable type” property of a capsule part is checked. Then the “Set..” button can be used to open the “Set Substitutable type” dialog which will list out all the specialized capsules that inherit from the abstract capsule. A checkbox is provided alongside each subcapsule which can be checked to specify that subcapsule will be used by the capsule part at run-time.



When the specialized capsules are set and we right-click a capsule part, and select either the “Open State Machine Diagram” or “Open Structure Diagram” options from the context menu, the “Open diagram for Substitutable type” dialog will open and list these specialized capsules.

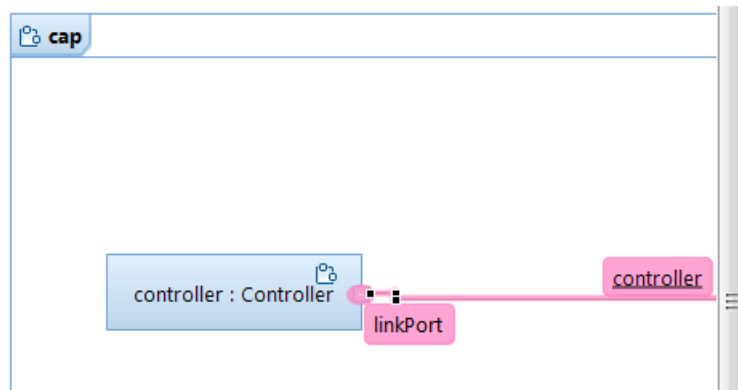
Select a concrete capsule from the list of the available capsules. Click “OK” and this opens the requested diagram for the concrete capsule.



Navigate to Source and Target of Lines

For any line that is selected in a diagram you can navigate to the symbols to which the line is connected. This can be useful if the diagram is large and you cannot see the source and target symbols at the same time without zooming out. The commands are called “Select Source” and “Select Target” and are available in the “Navigate” sub context menu.

Another way to perform the same navigation is to enable the preference *RealTime Development – Diagrams – Show label highlight even if the label is not visible*. This causes hyperlinks to appear when a line is selected, if the source or target symbol is not shown within the visible part of the diagram. Simply click such a hyperlink to scroll the diagram so that the remote end symbol becomes visible in the diagram. The picture below shows an example of a selected connector line, where the “controller” port is not visible in the diagram. Its label highlight therefore contains a hyperlink which can be clicked in order to scroll the diagram so the “controller” port becomes visible.



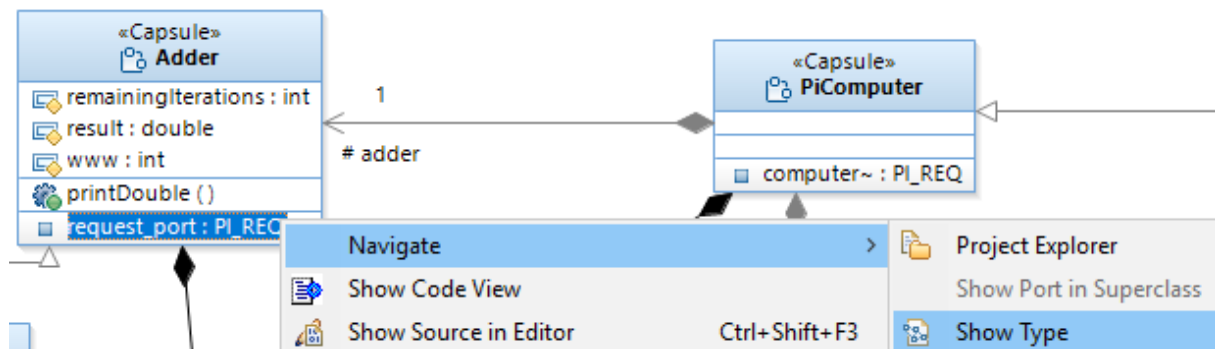
See also [Navigating Connectors](#) for another way to navigate within composite structure diagrams.

Open Source Code

Commands for opening the source code are available in the “Navigate” sub context menu. It locates the model element of the selected symbol, line or diagram and then works the same as the Project Explorer navigation commands described in [Open Source Code](#).

Show Type in Project Explorer

When looking at typed elements in a diagram, for example a port in a composite structure diagram or an attribute in a class diagram, you may want to navigate to the type of the element. This is done by running the context menu command *Navigate – Show Type*.



Showing Related Elements

Navigation commands similar to those described in [Show Type in Project Explorer](#) are also available for some other symbols and lines, where the corresponding model element references one or many other model elements which may be useful to navigate to. Here are some examples of such navigations:

- Navigating from a lifeline symbol in a sequence diagram to the property that is represented by the lifeline. The command is *Navigate – Show Property in Project Explorer*.
- Navigating from a message line in a sequence diagram to the operation or signal which it represents. The command is *Navigate – Show Referenced Element in Project Explorer*.

- Navigating from a partition symbol in an activity diagram to the element it represents. The command is *Navigate – Show Referenced Element in Project Explorer*.

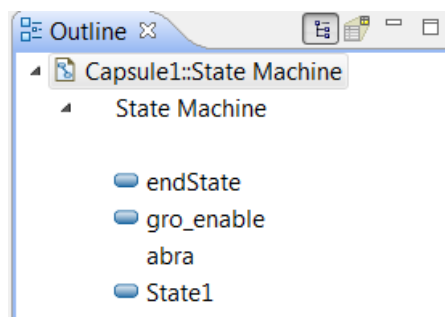
Other examples of navigation to related elements, which often also are available in the diagrams, can be found in [Navigating to Related Elements](#).

Outline View

The Outline view serves two purposes:

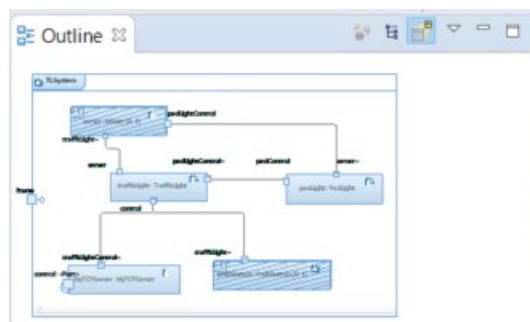
1. It gives a graphical overview of a diagram.
2. It enables quick navigation to different parts of a diagram.

You open the Outline view by running *Window – Show View – Outline*. It will update automatically when you open a diagram editor.



Note that a current limitation is that the icons for many kinds of elements are not shown in the outline view. For these you will only see the name of the element (which may be empty).

Use the toolbar buttons to toggle the Outline view between showing this outline and a graphical overview of the diagram.



Properties View Navigations

The Properties view shows properties for the selected element, but sometimes also information about elements that are related to the selected element. In those cases it's often possible to navigate to those related elements. The navigation commands are usually provided in the form of hyperlinks or buttons. By default the result of performing a navigation in the Properties view is to select the element that was navigated to in the Project Explorer. Remember to pin the Properties view before performing a navigation if you don't want it to switch to showing the properties of the element that was navigated to. If you instead prefer that such navigation commands should open a Property dialog showing the properties for the element, you can set the preference *Modeling – General – Navigation – Default navigation destination* to “Property Dialog”.

Navigate to Container Elements

The “Qualified Name” property on the General tab shows the qualified name of the selected element. Each name in the qualified name is a hyperlink which navigates to the container element with that name.



Open Type

For typed elements such as attributes, parameters and ports it is possible to navigate to the type of the element by clicking the “Open Type” button.



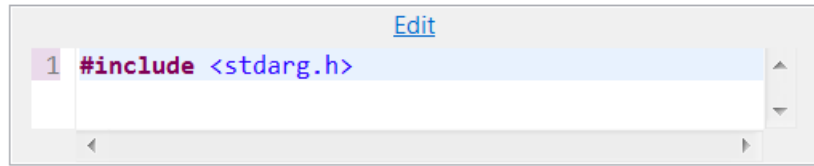
This button can be pressed even if the type reference has not yet been set-up for the element, or when the reference is unbound, which happens for example if the type has been removed from the model, or is located in a project which is currently not present in the workspace. In this case a direct navigation to the type element is not possible, and Model RealTime will instead start to search for the type. The search takes place primarily in the models that are present in the workspace. If it is not found there, you will be prompted if you also want to search for the type in generated C++ projects. This is useful in case the type refers to an external C++ type which is not present in the model. Finally, if the type cannot be found in generated C++ projects neither, a dialog appears which shows the URI of the missing type. This URI is helpful in case the reference is unbound because you forgot to import the project which contains the type into the workspace. Learn more about model element URIs in [Navigate by URI](#).

Navigate to C++ Code Snippets

All code snippets that are extensions to UML are stored in special properties on the model elements and can be viewed using the Properties view in the “C++ General” and/or the “C++ Target RTS” tabs. You cannot edit the code snippets there, but you can navigate to the code

snippet in order to show it in the Code Editor where you can edit it. To perform this navigation click on the "Edit" hyperlink above the code snippet:

Header Preface:



```
1 #include <stdarg.h>
```

Navigate to Redefined Operations

For an operation that redefines another operation in a super class the Properties view shows a hyperlink to that operation so that you can easily navigate to it.

Redefined Operation: [CPPModel::Passive1::rtg_getCurrentState](#)

As you can see there are hyperlinks also to the container elements of the operation so that you can navigate to these too.

Navigating Connectors

For a connector the Properties view shows information about the connector ends in the General tab. For each connector end its referenced port, (capsule) part and protocol are shown in a table:

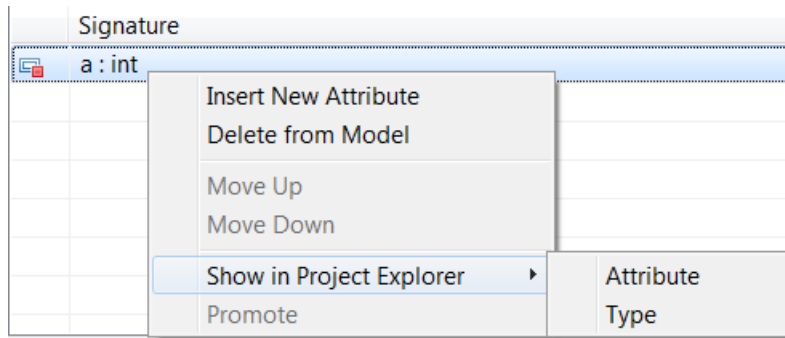
	Connector End	Connector End	
Ports	myport	myport	
Parts	capsule2	capsule1	
Protocols	p	p	

Use the hyperlinks to navigate to these elements.

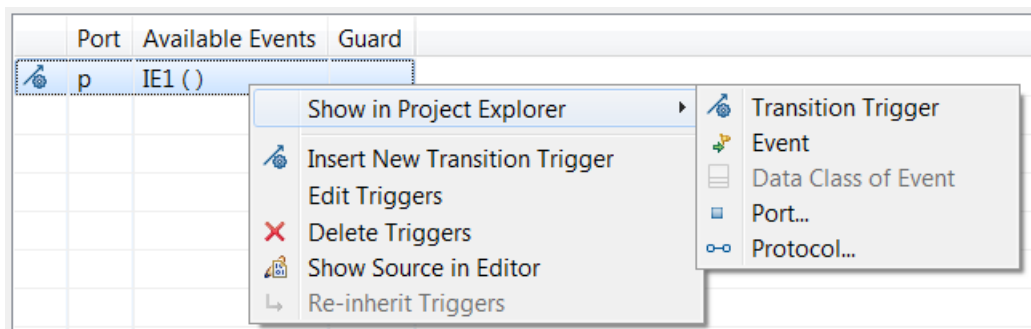
If you select the connector in a composite structure diagram the navigations to the ports and parts will be highlighted in the diagram. This is useful when working with big and cluttered composite structure diagrams where you may not be able to see both ends of a connector at the same time. Using these navigation commands can then be easier than trying to follow the connector line in the diagram when scrolling it.

Navigating to Related Elements

For some elements the Properties view shows lists of contained and/or related elements. For example, for a class its attributes are shown in the Attributes tab and its operations are shown in the Operations tab. You can navigate to the elements shown in such lists by right-clicking on them and running the command "Select in Project Explorer". In some cases there are other related elements which also can be useful to navigate to. For example, for an attribute navigation to the type of the attribute is also supported.



Another example is a transition trigger for which you may want to navigate to either the trigger itself, the protocol event that it references, the data class of its parameter (if any), the port it references or the protocol by which the port is typed. In this case all these navigation commands are available in a sub context menu called "Show in Project Explorer". For example:

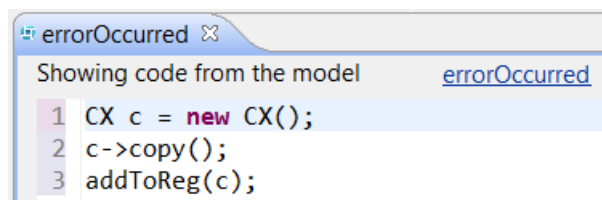


Code Editor and Code View Navigations

When working in the Code Editor or Code View there are several navigation possibilities for navigating within the source code and from the source code to the model. Navigations within the source code are provided by Eclipse CDT and are documented in the [CDT Documentation](#). These navigations are therefore only available once you have loaded generated source code using the "Load Generated Source" context menu command (by default it happens automatically when you click in the code). All other navigations are provided by Model RealTime and are described in this chapter.

Navigate from Code Snippet to Model

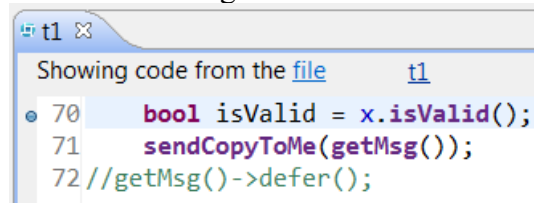
If you have many code editors open you may lose track of to which model element each of them belongs. In the tooltip of the Code Editor title you can see the fully qualified name of the model element, and to navigate to it you can press the hyperlink that is present just above the code snippet.



This hyperlink is available both in the Code View and Code Editor and is useful whenever you need to go to the model context of a code snippet. For example, if you are editing the body of an operation and realize that you need to add another parameter to the operation, using this hyperlink takes you to the operation in the Project Explorer where you can add the new parameter.

Navigate from Code Snippet to Generated File

Sometimes you may want to look at a code snippet in its full code context to look at surrounding code (generated, or from other code snippets). You can navigate from a code snippet to its location in the generated C++ file by clicking the hyperlink which appears in the top-left corner of the Code View or Code Editor once generated source code has been loaded.



The CDT editor opens and shows the code snippet in the generated C++ file. The same line is selected as was selected in the code snippet.

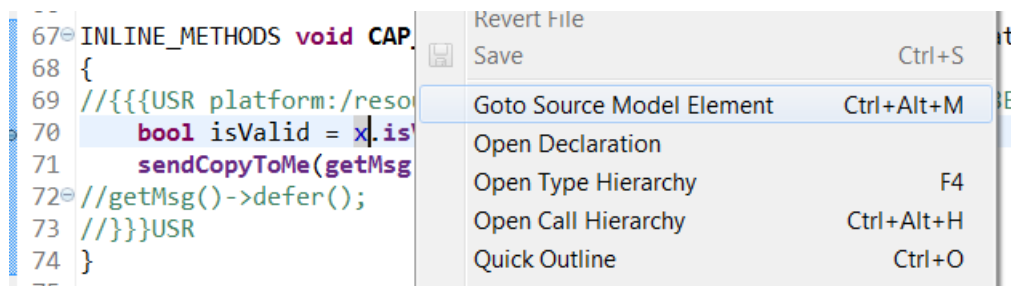
```

64 //}}}USR
65 }
66
67 INLINE_METHODS void CAP_Actor::transition2_t1( const void * rtdata, Timing::Base * rtport
68 {
69 //{{{USR platform:/resource/TimeoutDeferal/CppModel.emx#_GyeR8Kt8EeSbp9RtMa_nrg
70     bool isValid = x.isValid();
71     sendCopyToMe(getMsg());
72 //getMsg()->defer();
73 //}}}USR
74 }
75
76 INLINE_CHAINS void CAP_Actor::chain1_Initial( void )
77 {
78     rtgChainBegin( 1, "Initial" );

```

Navigate from Generated File to Model

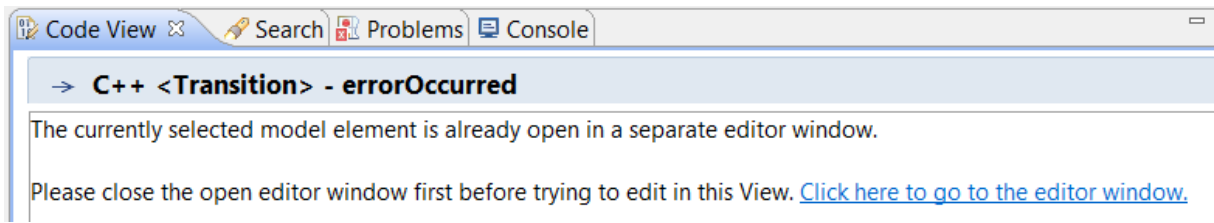
When you are viewing or editing a code snippet using the CDT editor you don't have a hyperlink for navigating to the model element to which the code snippet belongs, like you have when using the Code View or Code Editor (see [Navigate from Code Snippet to Model](#)). But you can still perform this navigation by right-clicking in the CDT editor on a line in the code snippet and then run the command "Goto Source Model Element". It's a commonly used navigation command and therefore has a default keybinding (Ctrl + Alt + M).



Another way to perform this navigation is to use the Properties view. It gets updated when you click inside a code snippet to show properties for the model element to which the code snippet belongs. You can click on the last hyperlink in the "Qualified Name" to navigate to the element in the Project Explorer. See [Navigate to Container Elements](#).

Navigate from Code View to Code Editor

The same code snippet can only be open for editing in one place at the same time. Either it can be edited using the Code View or you can have it open in a Code Editor, but not in both places at the same time. If you have many editors open, and one of them is a Code Editor for a particular code snippet that you wish to edit, it may be hard to find that particular Code Editor among all open editors. And you may have forgotten that you opened the code snippet in a Code Editor already. In this case it's useful to be able to navigate from the Code View to the Code Editor which shows the code snippet you want to edit. If the Code View detects that a code snippet is already open in a Code Editor it will provide a hyperlink which takes you to that Code Editor:

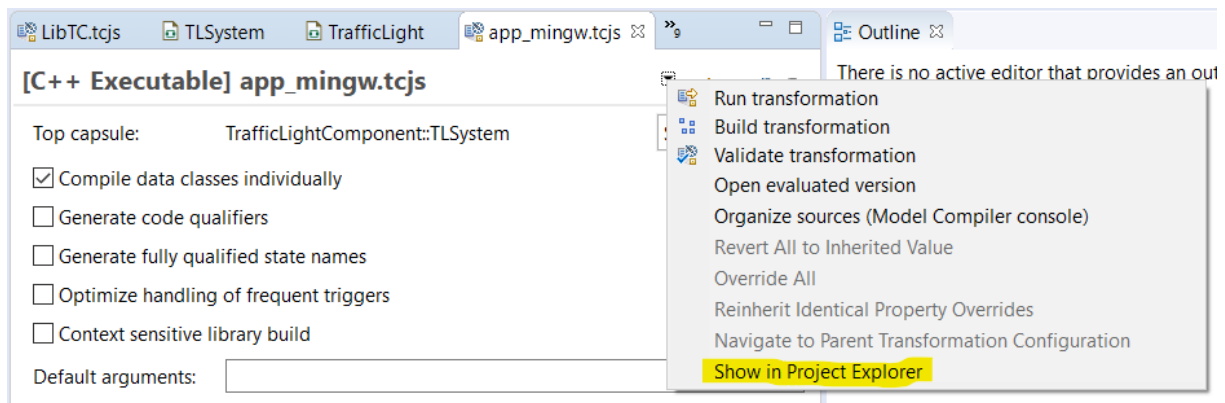


Transformation Configuration Editor Navigations

The transformation configuration editor provides a few navigation commands for finding related transformation configurations, and to locate a transformation configuration in the Project Explorer. You can also navigate to model elements that are referenced from the transformation configuration.

Show In Project Explorer

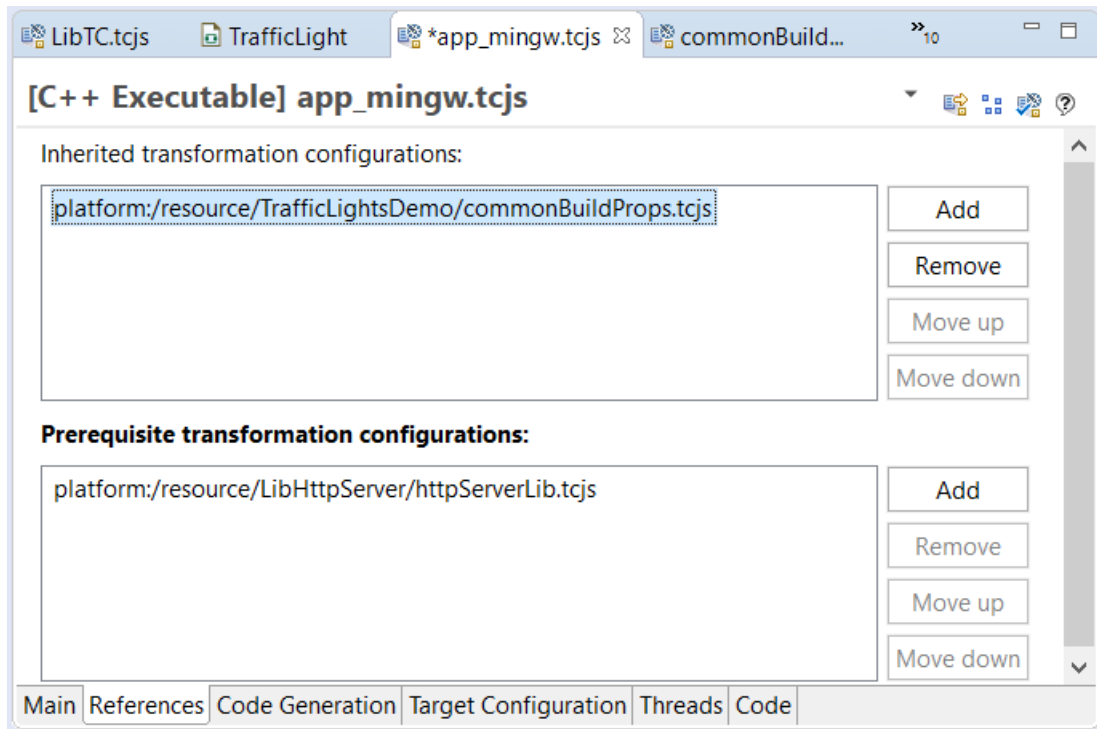
This navigation command is located in the drop down menu in the header of the transformation configuration editor:



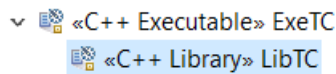
If the Project Explorer has been configured to show *.tcjs files, then the navigation command will select that file. Otherwise it will select the transformation configuration under the “Transformation Configurations” virtual folder.

Navigate to Inherited and Prerequisite Transformation Configurations

In the Properties tab of the transformation configuration editor inherited and prerequisite transformation configurations are listed. You can navigate to these transformation configurations by double-clicking on them. Navigation is done by opening a transformation configuration editor, and you can then use [Show In Project Explorer](#) to find it in the Project Explorer.

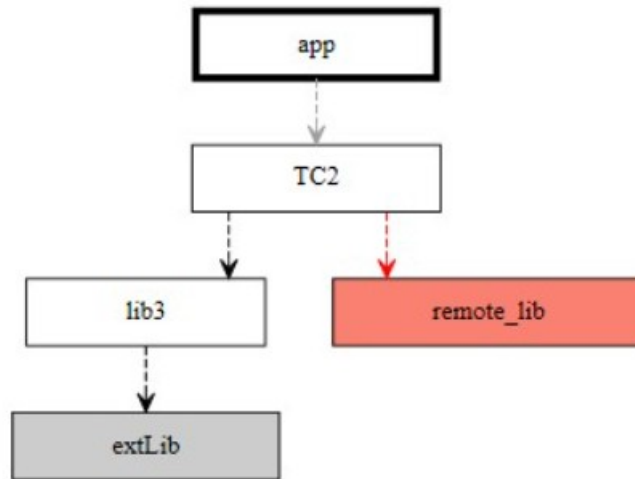


Prerequisite transformation configurations can also be seen in the Project Explorer, and you can double-click to open them in the transformation configuration editor:



If you set the preference *RealTime Development - Project Explorer - Show inherited transformation configurations*, inherited transformation configurations will also be shown in the Project Explorer.

It's possible to generate a graphical diagram that shows the inheritance and prerequisite relationships between transformation configurations. It can also show if any prerequisite transformation configuration cannot be found in the workspace (probably because you forgot to import the project that contains it). Here is an example of what such a diagram can look like:



For details on how to generate such a "transformation configuration visualization diagram" see this article:

[Visualizing TC relationships graphically](#)

Other Navigation Commands

Here we describe some other navigation commands which do not fit in the above categories.

Navigate by URI

All model elements have a textual representation in the form of a URI. Such a URI can for example look like this:

```
platform:/resource/TestProj/CLS.efx#_z_YOgNILEeSShtwgn6g7Ew
```

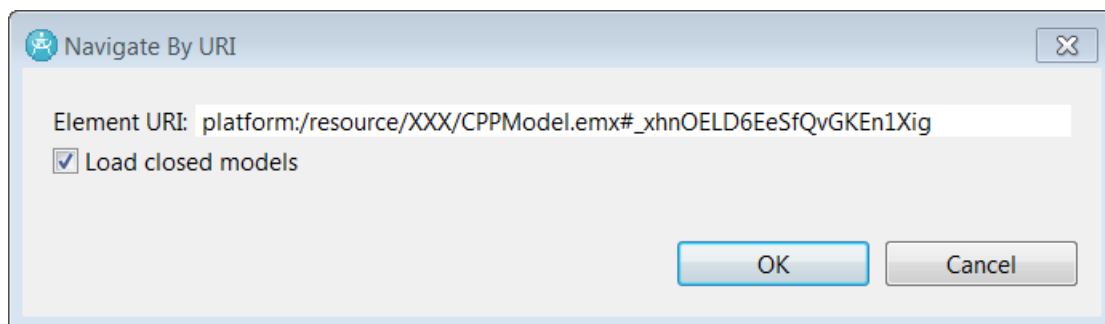
The 'platform' protocol is recognized by Eclipse and 'resource' identifies a file in the workspace. Then follows a path relative to the workspace which consists of the project and a model file within that project. After the '#' sign is the unique ID (UID) of the model element within that file.

When working with Model RealTime you will occasionally notice these model element URIs. For example, inside a generated C++ file there are USR sections that contain user written code snippets. Each such code snippet contains the URI of the model element to which the code snippet belongs. For example:

```
int Capsule1_Actor::foo( int param )
{
//{{{USR platform:/resource/MyProj/CPModel.emx#_xhnOELD6EeSfQvGKE1Xig
// user code here...
//}}}USR
}
```

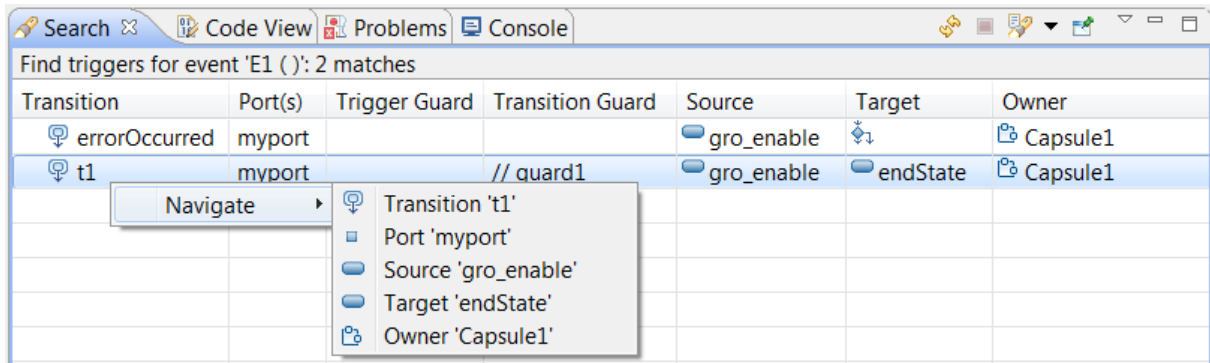
Sometimes you only have the UID part and then you can construct the full URI yourself by prepending it with "platform:/resource" followed by the workspace relative path to the model file.

Occasionally it's useful to navigate to an element within a model based on its URI. To do this copy the URI, and then perform the command "Navigate by URI" which is available in the Navigate menu. If your workspace contains closed models you need to mark the checkbox if you want these models also to be searched.



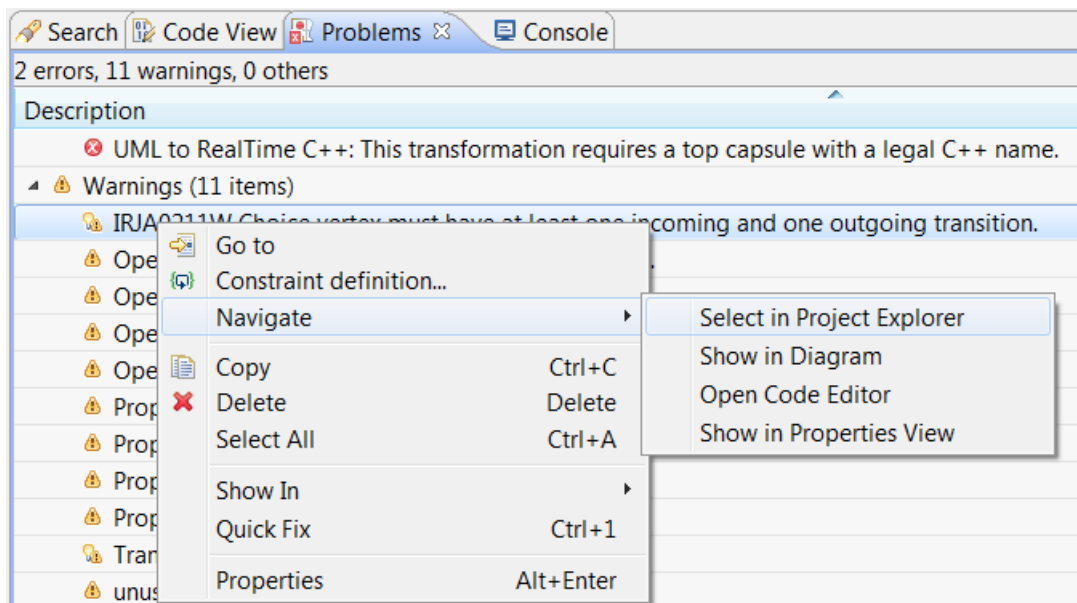
Search View Navigations

The Search view is used for showing the result of running a search command, but it is also used by some navigation commands where more than one possible navigation target exists. Often there is just a single element associated with each item in the search result, and you navigate to it by double-clicking on the search result item. However, in some cases there could be more than one element associated with each item. For example, when you run the search command "Find Triggers" the search result is a list of transition triggers. For each such trigger there are a number of related elements which could be interesting to navigate to. Navigations to those elements are provided by commands in a "Navigate" sub context menu:



Problem View Navigations

Usually a problem is associated with a single element, and then double-clicking on the problem will select that element in the Project Explorer. However, not all problems are best resolved using the Project Explorer. In some cases it would be better to navigate to somewhere else, for example a code editor, a diagram or the Properties view. If you feel the default navigation (which actually is different depending on the kind of problem), that you get when double-clicking the problem, is not appropriate, you can right-click on a problem and see if there are other navigation possibilities in the "Navigate" sub context menu.



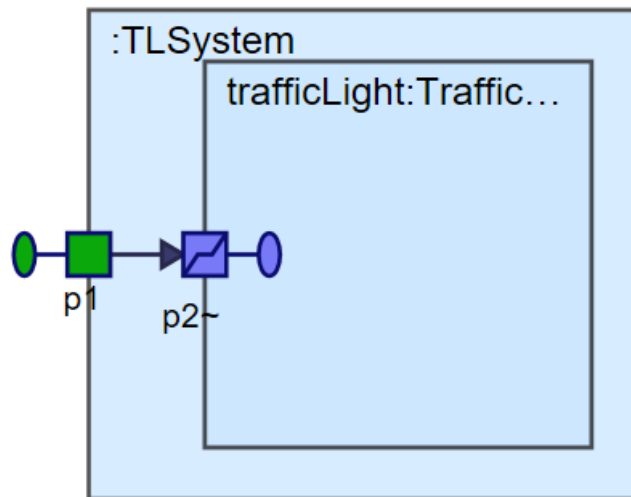
Navigating from Code Sending an Event to Where it is Received

A specific, but very common scenario, is that you have a line of code that sends an event. For example:

```
pedLightControl.walk().send();
```

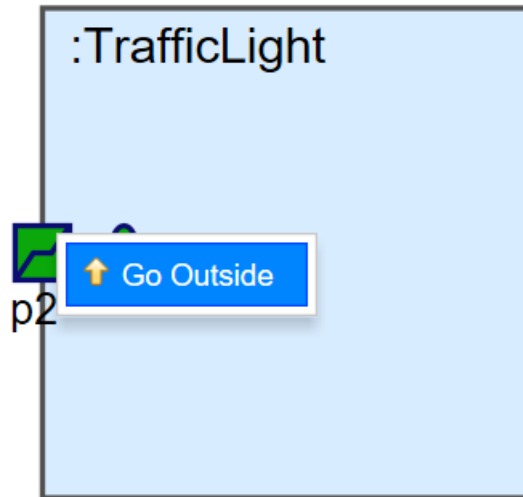
Now you want to navigate to the transition which may receive the event that is sent by this line of code. You can do this by right-clicking on the line in the Code or CDT editor and invoke the command *Visualize - Event Routing* (the command is not available in the Code view).

The command brings up a diagram that shows the composite structure of the capsule that owns the port on which the event is sent. In the diagram you can see how the event can be routed in the composite structure, and hence you can find out the behavior ports which may receive it. If the event is sent "inwards" in the composite structure of the capsule, the possible communication paths will be immediately visible. Below is a simple example where the sender and receiver ports are directly connected without intermediate relay ports:



Here we can see that an out event sent to port "p1" will be received by port "p2".

If, however, the event is sent "outwards" in the composite structure, you first need to invoke the command "Go Outside" on the port to see the next enclosing level in the composite structure. For example:



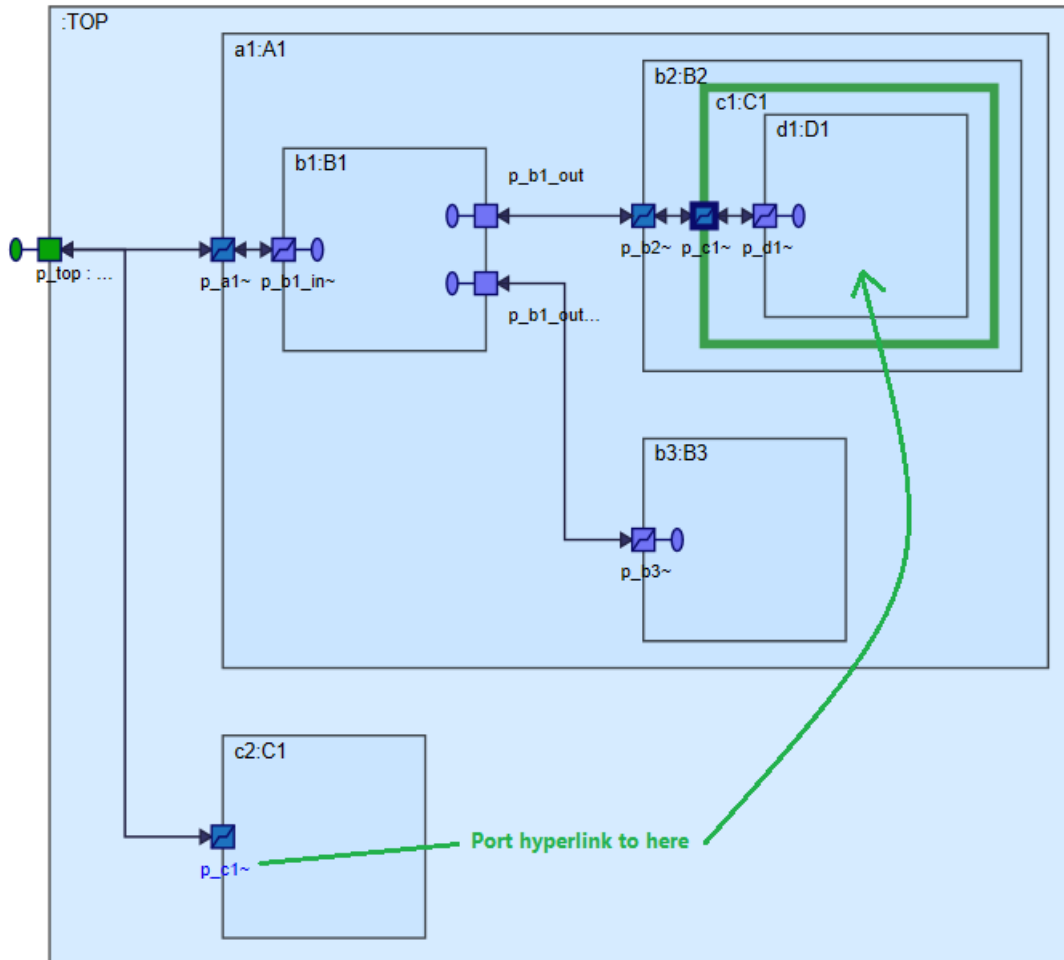
You may have to repeat invoking "Go Outside" until you have found the ports to which the sender port is connected. The "Go Outside" command (and the similar "Go Inside" command) works like in the composite structure diagram editor, except that "Go Outside" will only be available when the selected port is connected to a port with a connector in the enclosing composite structure. See [Go Outside/Inside](#) for more information.

Once you have found the port (or ports) that are the possible receiver(s) of the sent event, you can right-click on it and invoke the command "Find Triggers" in order to find the transitions that may trigger when receiving the event on that port.

Note that this navigation procedure also works on code lines that invoke or reply to an event.

Also note that it's possible to use the "Event Routing Visualization" feature as a means to understand a composite structure of a capsule in an easier way than to open several composite structure diagrams. You can right-click on a port shown in the Project Explorer or in a diagram and invoke the command *Visualize - Event Routing*. The only difference compared to when opening it from the Code or CDT editor is that the diagram will show communication paths for all events of the protocol that types the port. When it's opened from a line of code that sends an event, then only the communication paths for that specific event is shown. The difference is the direction of the arrows in the diagram (out events travel in the opposite direction to in events).

In complex composite structures the same capsule may type multiple capsule parts. To avoid very big Event Routing Visualization diagrams, the composite structure of such capsules will only be shown for one of the capsule parts. For the other capsule parts there will be hyperlinks on the ports that allow you to navigate to the composite structure shown elsewhere in the diagram. Here is an example where a capsule "C1" types two capsule parts ("c1" and "c2"):



The capsule part "d1" contained in "C1" is only shown once.

Finally, let's explain the colors of the ports shown in the above diagram:

- **Green:** This is the port on which you opened the diagram (either using the *Visualize - Event Routing* command, or *Go Outside/Inside* commands).
- **Blue:** Relay ports that just forward the event further
- **Purple:** Behavior ports that may receive the sent event. Also used for behavior ports, other than the green port, that may re-send the event. For example, the capsule B1 in the above diagram has two such ports. If the event arrives on port "p_b1_in" B1's state machine may re-send the event out on either of those two ports (thereby splitting one communication path into two).