



IBM Software Group

Searching in RSARTE Models

“seek and you will find”

Mattias Mohlin, January 2014
(updated for RSARTE 10.2 in July 2018)



Documentation

- For full Search documentation refer to the RSARTE wiki

<https://www.ibm.com/developerworks/community/wikis/form/api/wiki/b7da455c-5c51-4706-91c9-dcca9923c303/page/a1cb3f9a-1e85-45aa-8893-ff4354ffe0c9/attachment/0e25cf48-1c5a-4366-b698-3c2e759b6687/media/RSARTE%20Search.pdf>



Searching in RSARTE Models

*Mattias Mohlin
Senior Software Architect
IBM*

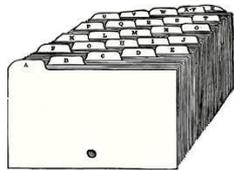
INTRODUCTION	2
DIFFERENT SEARCH STRATEGIES.....	2
SEARCH FIELD	4
SEARCH STRING PROPOSALS POPUP.....	4
VERBATM VERSUS NON-VERBATM SEARCHING.....	5
SEARCH PATTERNS.....	6
SCOPE.....	8
FILTERING.....	8
NAVIGATION.....	9
REPLACE.....	10
INDEX MANAGEMENT AND EXTERNAL PROJECTS.....	11
FIND NAMEELEMENT	13
NAMEDELEMENT FILTER.....	15
SEARCH SCOPE.....	16
NAME PATTERNS.....	17
SHOWING SEARCH RESULT IN SEARCH VIEW.....	17
SEARCHING IN THE SELECT ELEMENT DIALOG	19
FIND/REPLACE	21
FREE TEXT SEARCHING.....	23
SEARCH SCOPE.....	24
SEARCH OPTIONS.....	24
SEARCHING FOR REFERENCES TO AN ELEMENT	26
MEMORY-BASED SEARCH COMMANDS	28
FIND TRIGGERS.....	29
FIND TRIGGERING EVENTS.....	29
FIND CONNECTED PORTS.....	30
INCREMENTAL TEXTUAL SEARCH IN DIAGRAMS	32



Overview

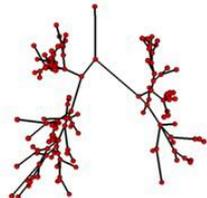
- RSARTE provides multiple search commands
 - ▶ There are usually more than one way to find the element you are looking for, but depending on situation some commands may be more efficient than others
 - ▶ Look upon the multiple search commands as multiple tools in your toolbox (you are more efficient than if you only had one tool for all tasks, but you need to know how to use them!)
- Three categories of search commands

- ▶ **Index-based**



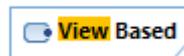
Uses a search index. Fast, once the index is up-to-date. Limited search criteria (can only search on the information that is indexed, such as element name). Index is stored in workspace.
Example: *Search Field*

- ▶ **Memory-based**



Uses the model loaded into memory. Fast, once the model has been loaded (e.g. using Load UML Models). Custom search criteria can be used.
Example: *Find Triggers*

- ▶ **View-based**

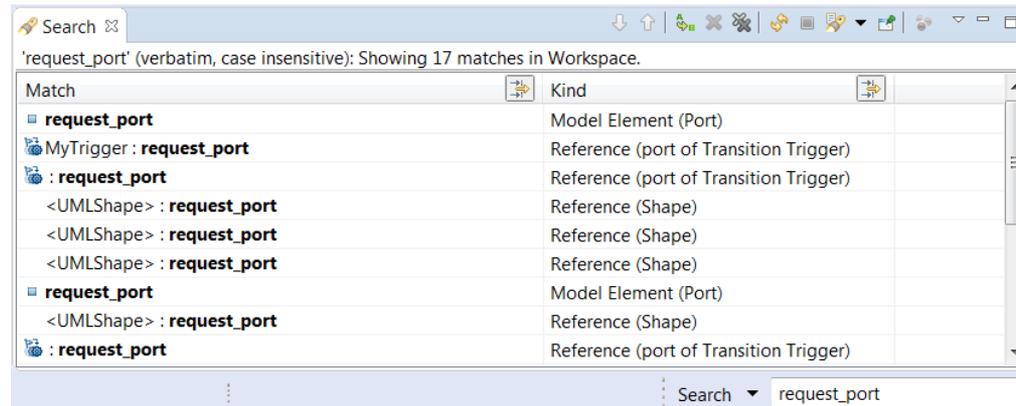


Uses what is shown in an editor.
Example: *Incremental Diagram Find*



Search Field

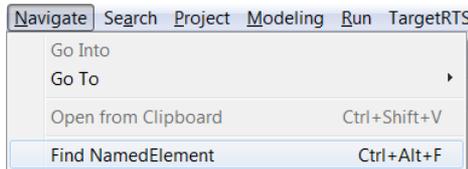
- General purpose searching in models
 - ▶ Index-based search command (index build in background)
 - ▶ Runs a search without use of modal dialogs
 - ▶ Automatic proposals popup for common and recently used search terms
 - ▶ Support for filtering the search result (including negative filters)
 - ▶ Support for invoking Replace on the search result
 - ▶ Can search either verbatimly (default) or on multiple words separately
 - ▶ The most powerful search command – can search in references, TC settings, and projects external to the workspace. **This command should be your default way of searching for something in RSARTE!**



Find NamedElement

- Finds an element with a certain name

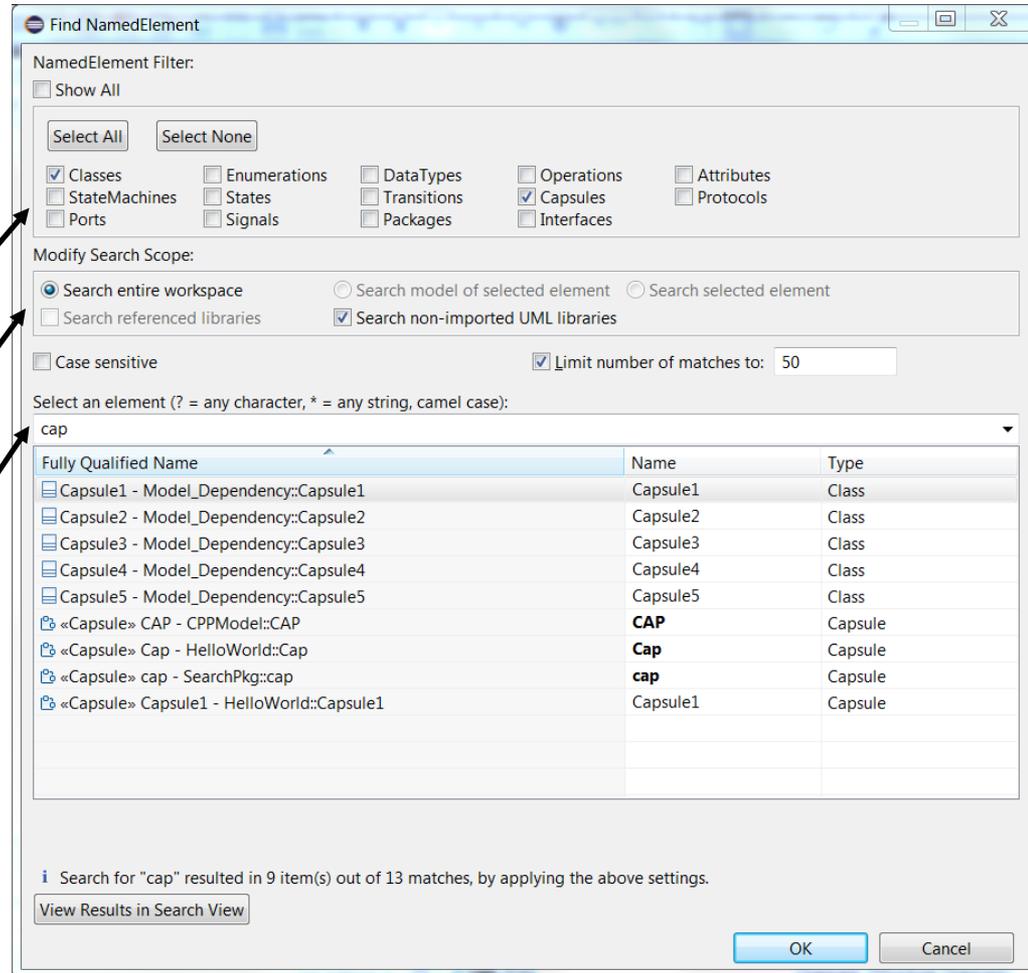
- ▶ Index-based search command
- ▶ Shows matches in dialog as you type
- ▶ Best to use when you are looking for one particular element in the model



Filters for what is commonly used in RT models

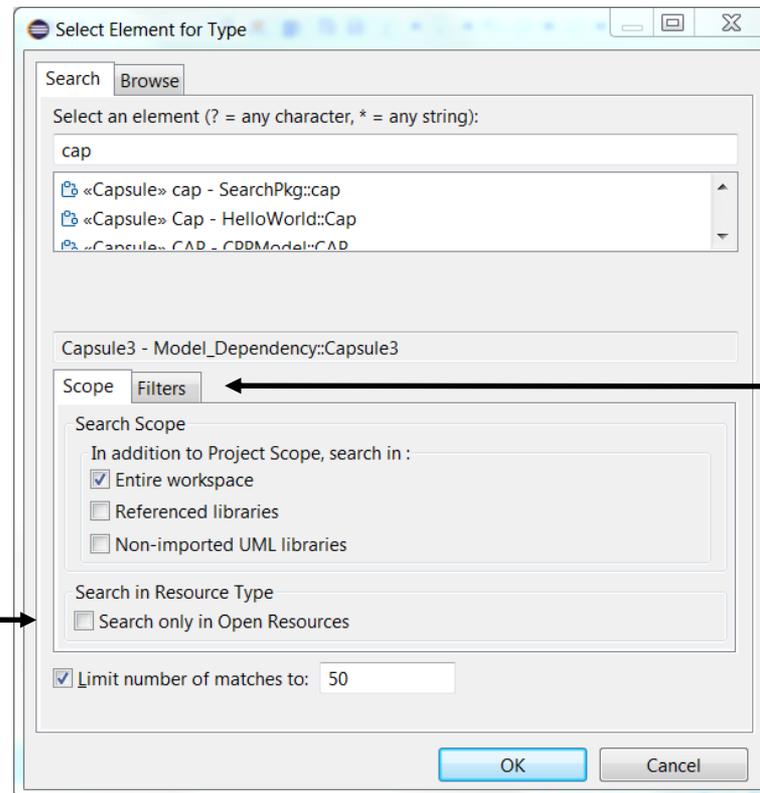
Use as narrow search scope as possible

Search string may use wildcards for pattern searches



Select Element Dialog

- Finds an element to use in some context other than searching
 - ▶ Example: Press the Set button to set-up the type of an attribute
 - ▶ Think about this as a simplified Find NamedElement dialog



Mark to not search in the entire index (skip closed models)



This dialog is not RT specific so there are many more filters here

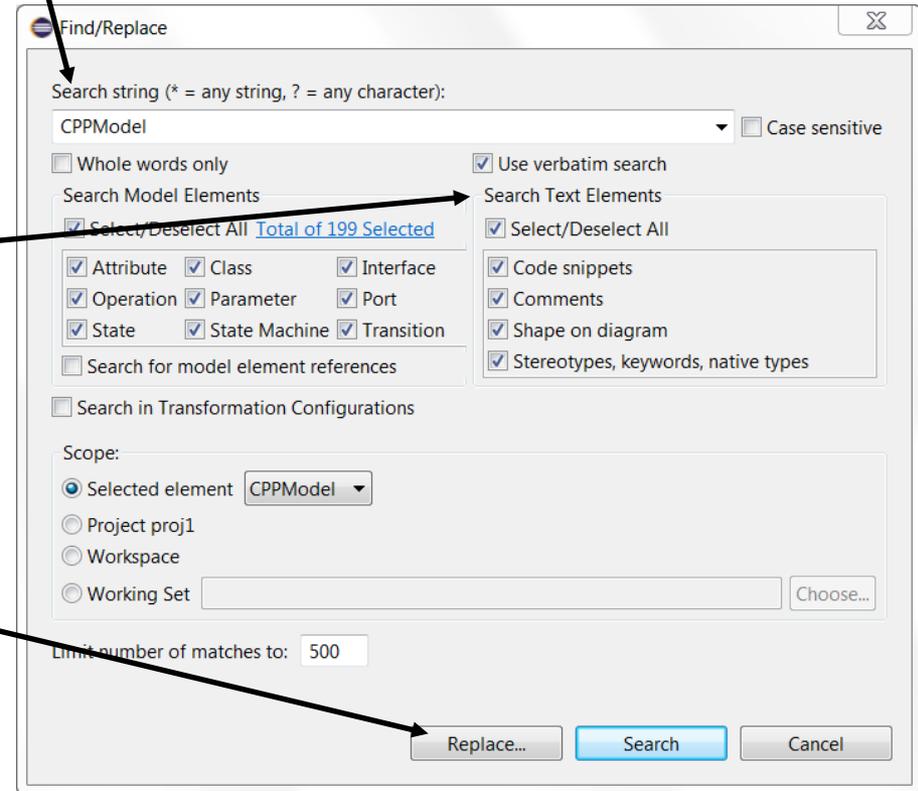


Find/Replace & Model Search Dialogs

- Finds elements with a certain name
 - ▶ Index-based search command
 - ▶ Matches are shown in the Search view when the dialog is closed
 - ▶ Similar to use of Search field, but requires settings to be made before searching

- Finds texts in the model
 - ▶ Also index based
 - ▶ In code snippets, comments etc.
 - ▶ Better to use the Search field instead!

- Replacing
 - ▶ Click “Replace” instead of “Search” to iterate the search result and replace the found name or text with some other string



Modeling References

- Finds model references to an element
 - ▶ Index-based search command
 - ▶ Available in the context menu of a model element
 - ▶ Answers the question “Where in the model is this element used?”
 - ▶ Note: Only model references are found – not textual references (for example from code snippets). The command Code Snippet References can be used for that. It is based on CDT:s References command (and hence requires presence of generated C++ code).
 - ▶ For elements that have a name, you can instead use the Search field to find the references (filter the Kind column to only show reference matches)

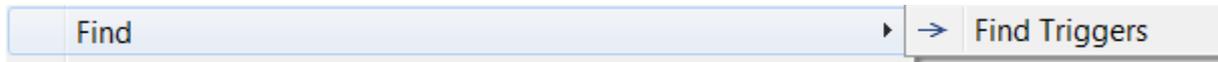


Select the scope where to look for elements that reference the selected element.



Find Triggers

- Finds transition triggers for a selected element
 - ▶ Memory-based search command
 - ▶ Available in the context menu (Find) of a port, protocol event or trigger operation



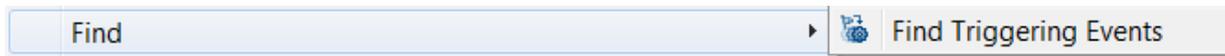
- **Protocol event**
Finds the transition triggers which trigger on the protocol event.
- **Behavior port**
Finds the transition triggers which trigger when an event arrives at the port.
- **Trigger operation**
Finds the transition triggers which trigger when the trigger operation is called.
- ▶ Found triggers are listed in the Search view together with related information. Navigate to it by double-click (or in Navigate context menu).

 A screenshot of the Search view in a software application. The title bar says 'Search'. Below it, the text reads 'Find triggers for event 'InEvent2 (data : int): 1 match'. A table displays the search results. The first row is selected and has a context menu open over it. The context menu has a 'Navigate' option, which is highlighted. The table has columns for Transition, Port(s), Trigger Guard, Transition Guard, Source, Target, and Owner.

Transition	Port(s)	Trigger Guard	Transition Guard	Source	Target	Owner
t1	pp			State1	State2	cap

Find Triggering Events

- Finds the events which may trigger a transition
 - ▶ Memory-based search command
 - ▶ Available in the context menu (Find) of a state, state machine or capsule/class



- **State**
Finds the events which may trigger a transition when the state is active.
- **State machine**
Finds all events which the state machine can handle.
- **Capsule or class with state machine**
Finds all events which the state machine of the capsule/class may handle.

Search Find triggering events for state 'Nested': 3 matches

Order	Event	Transition	Port(s)	Trigger Guard	Transition Guard	Source	Target
-1.1	reply ()	doReply	basePort		size <= capacity	State1	State2
0.1	handle (Parameter1 : bool)	handleReq	basePort		size <= capacity	Nested	Nested
1.1	abort ()	exceptionOccurr...	basePort			S	State1

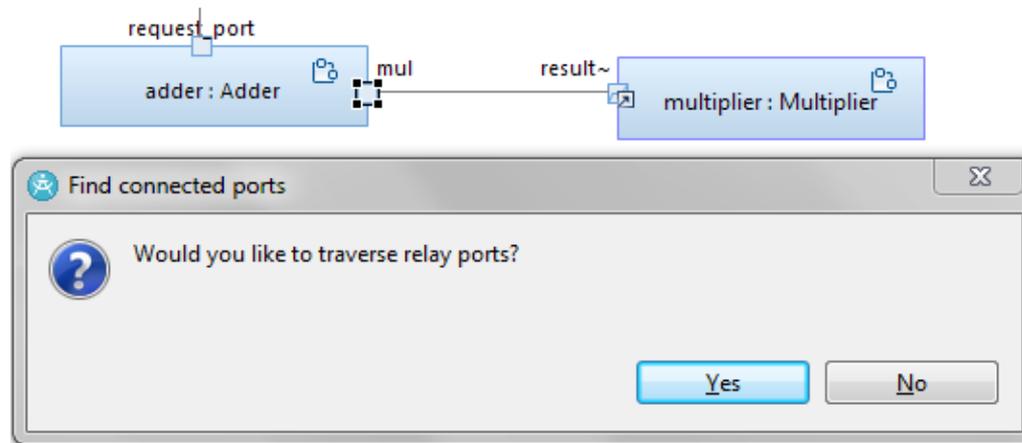
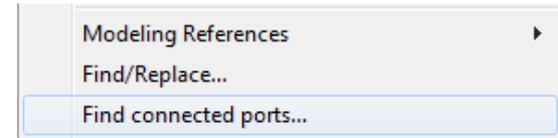
Navigate

- Transition 'exceptionOccurred'
- Event 'abort ()'
- Port 'basePort'
- Source 'S'
- Target 'State1'



Find Connected Ports

- Finds the ports which a selected port may communicate with
 - ▶ Memory-based search command
 - ▶ Available in the context menu of a port
 - ▶ You can choose to traverse relay ports



- ▶ Found ports are listed in the Search view

Search			
Found 1 port(s) connected to mul : INC_REQ			
Port	Part with Part	Protocol	Diagram Context
result	: Multiplier	INC_REQ	PiComputer



Incremental Diagram Search

- Finds texts shown in a diagram
 - ▶ View-based search command
 - ▶ Same keyboard accelerators as for text editors (Ctrl+J for forward search and Ctrl+Shift+J for backwards search)
 - ▶ In particular useful for finding texts in big and cluttered diagrams
 - ▶ Works also in derived diagrams such as browse diagrams

