



MetaModelAgent

for IBM Rational® Software Architect Designer and RealTime Edition
HCL RealTime Software Tooling and Eclipse Papyrus™

Model Analysis Manual

MetaModelAgent version 4.5.0

AO-MMA-023 • 2020-09-25



© 2020 Adocus AB

This document and its contents are protected by copyright
and must not be copied or distributed, wholly or partially
without prior authorization from Adocus

Contents

1	Introduction	4
1.1	What's New	4
1.2	Support	5
2	Terminology	6
2.1	Modeling terms	6
2.2	Eclipse terms	6
3	Overview of Analysis Views	7
3.1	Invoking model analysis	7
4	Property Table View	8
4.1	Invoking the View	8
4.1.1	From Explorer View, Diagram Editor and Activation View	8
4.1.2	From Guidance View	8
4.1.3	From other Views	9
4.2	View Toolbar	9
4.3	Context Menu	9
5	Trace Matrix View	10
5.1	Invoking the View	11
5.1.1	From Explorer View, Diagram Editor and Activation View	11
5.1.2	From Property Table View	12
5.2	View Toolbar	13
5.3	Context Menu	13
5.4	Matrix Modes and filters	14
5.4.1	Normal Mode	14
5.4.2	Transitive Mode	14
5.4.3	Transitive (Start-End) mode	15
6	Trace Tree View	16
6.1	Invoking the View	16
6.1.1	From Explorer View and Diagram Editor	16
6.1.2	From Trace Matrix View	17
6.2	View Toolbar	17
6.3	View Context Menu	18
7	Chart View	19
7.1	Available Charts	19
7.2	Invoking the View	20
7.2.1	From Explorer View, Diagram Editor and Activation View	20
7.2.2	From Property Table View	20
7.2.3	From Trace Matrix View	20
7.2.4	From Problem View	21

7.3	View Toolbar	21
7.4	View Context Menu	21
8	Static Behavior Analysis	22
8.1	Potential State Machine Transitions	22
8.1.1	Transition Paths to/from a specific State, Entry or Exit point	22
8.1.2	All Transitions in a Hierarchical State Machine	23
8.2	Potential Activity Flows	25
8.2.1	Activity Flows to/from a specific Action	26
8.2.2	All Activity flows in an Activity and referenced Activities	26
8.3	Capsule Connectors	29
8.3.1	Potential Connector paths to/from a Port in a Capsule Part	29
8.3.2	All Connectors in a Capsule Composition Structure	30
8.4	Capsule Connectors for a specific Protocol	32
8.5	Overall Protocol Usage	33

1 Introduction

MetaModelAgent (MMA) is an Eclipse-based modeling tool extension that is installed on top of a host tool and accessed by extended menus and views within the Eclipse workbench. MetaModelAgent supports the following host tools:

- IBM Rational® Software Architect Designer (RSAD)
- IBM Rational® Software Architect RealTime Edition (RSARTE)
- HCL RealTime Software Tooling (RTist).
- Eclipse Papyrus™

This is the user manual for the static model analysis capabilities in MetaModelAgent using the built-in metamodels for standard UML or UML-RT or when using any user-defined domain-specific metamodel. There are two other manuals available in the Help-system after installation:

MetaModelAgent Modeling User Manual	Describes how to activate MetaModelAgent for your models and how to use it for domain-specific modeling using your own metamodel.
MetaModelAgent Metamodeling Manual	Described how to develop your own domain-specific metamodel for usage in MetaModelAgent.
MetaModelAgent License Management	Described how to require and install licenses to be able to use MetaModelAgent.

1.1 What's New

Older version before v4.3.1

The version history of older versions of the analysis features are listed in the *MetaModelAgent Modeling User Manual*.

From v4.3.1 to v4.4.0

- This manual is a new user manual that focus on the model analysis capabilities and features in MetaModelAgent. In the previous versions, the (limited) analysis capabilities were described in the *MetaModelAgent User Manual*, which now have been renamed to *MetaModelAgent Modeling User Manual*.
- Analysis of implicit derived relations based on underlying explicit relationships in *Trace Matrix View* and *Trace Tree View*.
- Analysis of transitive relation chains using *Trace Matrix View* and *Trace Tree View*.
- Colorized grid in *Trace Matrix View* provides information of relationship volumes and relation chain distances as well as occurrences of circular relationship chains.
- Source and target context available in Matrix View.
- Support for populating *Trace Tree View* using several pre-defined queries.
- Support for static analysis of Activity Flows using *Trace Matrix View* and *Trace Tree View*.
- Support for static analysis of complex, hierarchical state machines in *Trace Matrix View* and *Trace Tree View* according to both standard UML and UML-RT semantics.

- Support for analyzing UML-RT Capsule connections in capsule composite structures using *Trace Matrix View* and *Trace Tree View* with respect to UML-RT Semantics.
- UML-RT Protocol usage analysis in *Trace Matrix View*.
- Support for analyzing several models and nested models simultaneously.
- Improved styled tooltips in all views displaying complete paths to elements.

From v4.4.0 to v4.4.1

- Trace Tree View can be used directly for behavioral analysis, without having to populate the Trace Matrix View as an initial step.
 - Potential transitions flow to or from a state or entry/exit point within a hierarchical state machine, see chapter 8.1.1.
 - Potential activity flows to/from an activity node with an activity, see chapter 8.2.1.
 - Explicit or potential incoming or outgoing communication paths to/from a specific capsule part or a capsule part's port, see chapter 8.3.1.
- Property Table View: Tooltips on properties representing valid URLs displays hyperlinks for each URL. This makes it simple to navigate to the URL using the default web browser.
- Property Table View: Related Element Bar Chart context menu provides a separate menu entry for each relation metaclass. Making it possible to filter the Bar chart bases on relation metaclass.
- Property Table View: Columns displaying integer properties are right-adjusted.
- Property Table View: Editing of multi-valued properties supported.

From v4.4.1 to v4.5.0

- New Circular filter in Matrix View. See chapter 5.4.
- Additional predefined filters in Trace Matrix View. These new filters are *All Navigable Associations*, *All Properties* and *All Interface Connections*. See chapter 5.1.1.
- Predefined filter *All Interface Connections* filters in Trace Matrix View. These new filters are All Navigable Associations, All Properties and All Interface Connections See chapter 5.1.1.
- Support for initiating the Trace Matrix View from a diagram, to show all visible and invisible connections within the diagram. See chapter 5.1.1.
- Chart View can be initiated from Trace Matrix View, see chapter 5.2 and chapter 7.2.3.
- Support for incomplete activity flows in Trace Matrix View and Trace Tree View.
- Metaclass filter provided when initiating a related element scatter chart in the Chart View from the context menu in the Property Table View.

1.2 Support

To obtain support, please use the support request form on the Adocus Website (www.adocus.com) or send an email to support@adocus.com.

2 Terminology

The following terms are used frequently in this user manual.

2.1 Modeling terms

<u>Term</u>	<u>Explanation</u>
Item	Item is the generic term for an element, a diagram or a relationship. Examples of items are: classes, dependencies and activity diagrams.
Property	A property is a predefined or user defined feature of an item. Examples of common properties are: name, stereotype, multiplicity, documentation.
Model	A model is a UML-model holding a set of items which represents a model as defined in UML, e.g. a semantically complete abstraction of a system. Examples of common models are Use Case Models, Analysis Models and Design Models.
Metaclass	A metaclass is a classification of items sharing the same characteristics. Examples of standard UML metaclasses are Use Case, Actor and Class.
Metamodel	A metamodel is a UML-model that defines the model guidelines for a specific kind of models in a formal way using metaclasses. A model can therefore be seen as an instance of its metamodel.

2.2 Eclipse terms

Explorer View	The Explorer View is the name used in this manual for the standard view that shows the model structure of loaded models. <ul style="list-style-type: none">• <i>Project Explorer View</i> in RSAD/RSARTE and HCL RTist.• <i>Model Explorer View</i> in Papyrus.
Diagram Editor	The Diagram Editor is the graphical editor view for editing UML-diagrams.
Property View	The Property View is used for viewing and editing item properties.

3 Overview of Analysis Views

MetaModelAgent provides four different views that are used for model analysis:

- The **Property Table View** lets you inspect elements, relationships and diagrams and their significant properties in an editable table layout. The table can be filtered and sorted on any column and the content may also be exported to a CSV-file for further post-processing, for example in MS Excel.
- The **Trace Matrix View** lets you inspect relationships and other kinds of references in a matrix layout. It can also be used to detect hidden relationships and implicit relations between and within models based on underlying explicitly modeled relationships.
- The **Trace Tree View** lets you inspect relationship chains starting from, or finishing at a specific element. You can easily detect unwanted relationship chains and circularities.
- The **Chart View** lets you visualize several aspects of your model in graphical bar charts and scatter charts.

MetaModelAgent provides unique capabilities to navigate between the views to further elaborate the structures, contents and behavior of your models.

The picture below, summarize the different ways each view can be invoked based on the content in other views.

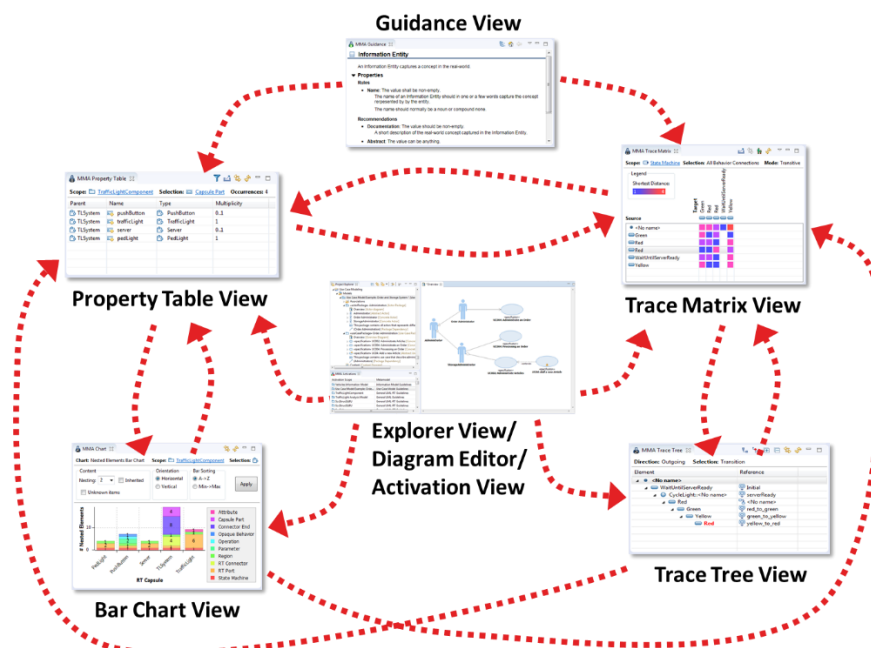


Figure 1: Schematic picture on potential navigation between the different views using each view's context menus

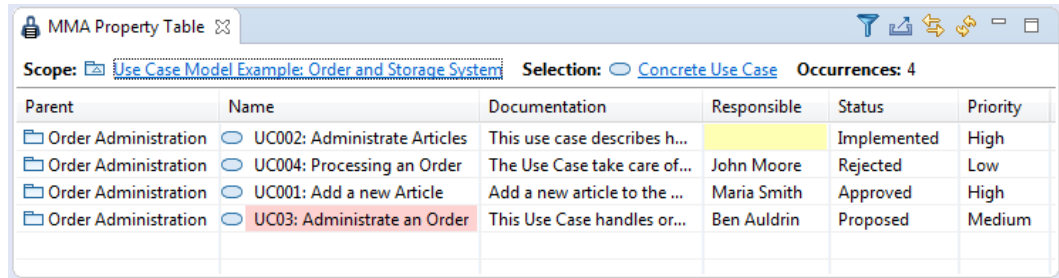
The capabilities and features of each view are described in detail in the following chapters.

3.1 Invoking model analysis

To be able to use model analysis, you must activate MetaModelAgent for all of your models to be analyzed. Please refer to *chapter 4 Start Using MetaModelAgent* in the *MetaModelAgent Modeling Manual* on how to activate MetaModelAgent for your models.

4 Property Table View

The *Property Table View* is a unique spread-sheet table view which primarily shows an overview of all significant properties for all items within a selected scope that belongs to the same metaclass.



The screenshot shows a window titled 'MMA Property Table'. It has a toolbar with icons for search, zoom, and other functions. Below the toolbar, there are fields for 'Scope: Use Case Model Example: Order and Storage System', 'Selection: Concrete Use Case', and 'Occurrences: 4'. The main area is a table with the following data:

Parent	Name	Documentation	Responsible	Status	Priority
Order Administration	UC002: Administrate Articles	This use case describes h...		Implemented	High
Order Administration	UC004: Processing an Order	The Use Case take care of...	John Moore	Rejected	Low
Order Administration	UC001: Add a new Article	Add a new article to the ...	Maria Smith	Approved	High
Order Administration	UC03: Administrate an Order	This Use Case handles or...	Ben Auldrin	Proposed	Medium

Figure 2: *Property Table View showing significant properties for all concrete use cases within the package "Order Administration".*

Any invalid property value, according to the current metamodel is displayed with a background color that indicates the severity of the violence.

Most of the properties are editable. Just click in the corresponding cell and edit the value. Depending on the kind of values that are valid, popup dialogs for selecting or editing a value may occur.

4.1 Invoking the View

4.1.1 From Explorer View, Diagram Editor and Activation View

Instances of a specific metaclass

To show all items that are instances of a specific metaclass:

1. Select one or several elements in your activated models to be the scope.
2. Bring up the context menu and select *MetaModelAgent*→*Show in Property Table View*.
3. There will be one menu entry for each metaclass that is valid within the selected scope. If there are a lot of metaclasses to choose from, they will be organized in consecutive submenus according to the current metamodel.

All items that match the selected metaclass and that are nested to the selected scope will be displayed in the table view.

All instances of a specific metaclass

An alternative way to populate the Property Table View when using your own metamodel available in the workspace:

1. Select a metaclass in the metamodel for which there are at least one current instance,
2. Bring up the context menu and select the context menu entry *MetaModelAgent*→*Show in Property Table View*→*Show all Occurrences*.

This will display all instances of the selected metaclass from all current activations.

4.1.2 From Guidance View

The Property Table View can show all instances of the metaclass that is currently displayed in the Guidance View.

1. Make sure that the Guidance View shows the metaclass for which you are interested in, by selecting an item belonging to that metaclass in some other view or in the diagram editor.
2. Select *Show all occurrences in Property Table View* in the view's pull-down menu in the view's header.





The Property Table View will then display all occurrences of the current metaclass in all activated models.

4.1.3 From other Views

The Property Table View can be populated in different ways from the other analysis views as well. Please refer to the description of those view's content menus for details.





4.2 View Toolbar

The view's toolbar provides the following tools:

-  Brings up a dialog where you can sort and filter out the columns to be displayed in the table view. There are always three potential columns that are initially hidden and can be revealed using this dialog, those are:
 - *Path*; complete path from the model root to the element.
 - *Metaclass*; the item's metaclass. This column is useful if the view is used to display a heterogeneous list of item.
 - *Metamodel*; the metamodel that the model holding the element is using. This column is useful if the view is used to display items in models where different metamodels are applied.
-  Brings up a dialog where you can export the current view content to a comma-separated text file for post-processing in other tools, for example MS Excel.
-  Links the content of the view to the element selected in the Explorer View or in the Diagram Editor. This means that the view will be update with the new selection as scope, but the same selection of metaclass and applied column filter remains.
-  Refreshes the content of the view. This is useful if the items being displayed have been changed in some other view.

4.3 Context Menu

There is a context menu available when selecting one or several rows in the table. The context menu entries provides the following functionality:

-  *Show in Trace Matrix View* Populates the Trace Matrix View in several different ways. Please refer to the chapter 5.1 for details.
-  *Show in Chart View* Populates the Chart View in different ways. Please refer to chapter 0 for details.
-  *Bulk Property Edit* Enables bulk editing of a specific property. All selected rows must represent the same kind of element for this submenu to be available.
This menu entry is not always available, depending on from where and how the Property Table View was invoked.
-  *Delete <n> selected <Metaclass>* Deletes the items that are represented by the selected rows from the model. This menu entry is not always available, depending on from where and how the Property Table View was invoked.

5 Trace Matrix View

The *Trace Matrix View* displays the existence of relationships or relationship chains as well as attribute type references, hereafter referred to as *references* only, which fulfills the chosen criteria in a grid style layout. Source elements are listed vertically and target elements are listed horizontally.

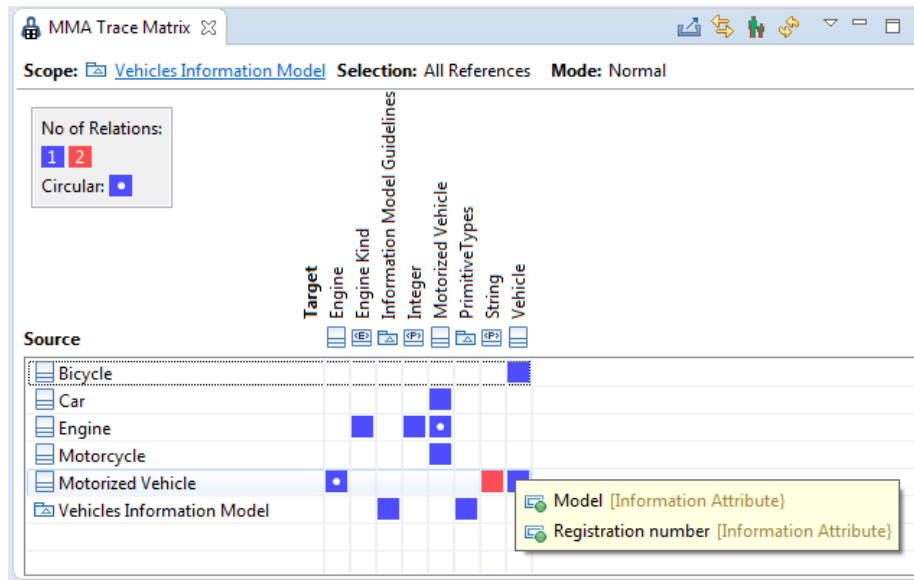


Figure 3: Trace Matrix View showing all relations and attribute types.

The intersection cell between a source and a target indicates if there are direct or indirect references or references chains, depending how the Trace Matrix View was invoked, from the source to the target element.

- The cell color (gradual from blue to red) indicates the relative number of references between source and target or the distance between source and target in terms of consecutive references.
 - Blue represent the longest number/shortest distance.
 - Red represents the highest number/longest distance.
- A white dot in the middle of a colored cell indicates that the source and target element is involved in a circular reference chain.
- A grey cell color indicates that the cell represents an implicit reference between source and target that is relevant in the context of current selection. This is only used in static behavior analysis of state machines, activities and capsule connectors. See chapter 8.
- An empty cell indicates that there are no references or reference chains that fulfills the chosen criteria between the source and target element.

The view does not support horizontal scrolling, if there are more target elements that can fit in the view, numbered “page” buttons will be displayed that let you switch between different “pages” of targets.

You may observe that this view is just a read-only snap-shot view of relationships or association ends. You may not edit the relationships in the view and any changes to the relationships from other parts of the UI will not be automatically reflected.

5.1 Invoking the View

5.1.1 From Explorer View, Diagram Editor and Activation View

Instances of a specific metaclass

To show all relations or attribute type references that are instances of a specific metaclass:

1. Select one or several elements in the Explorer View or Diagram Editor for which you are interested in their nested relationships or attribute type references.
2. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Matrix View*.
There will be one menu entry for each metaclass representing a relationship or attribute that are valid within the selection and for which there are at least one matching item within the selection.

IMPORTANT: It is possible to invoke the Trace Matrix View with activity flows, state machine transitions or capsule connectors but they will only display the immediate sources and targets in the matrix. To analyze those kind of connections in a correct way with respect to UML and UML-RT semantics, you should use the static behavior analysis features, see chapter 8 for details.

All instances of a metaclass







An alternative way to populate the Trace Matrix View when using your own metamodel available in the workspace:





1. Select a metaclass in the metamodel representing a relation or attribute for which there are at least one current instance,
2. Bring up the context menu and select the context menu entry
MetaModelAgent→*Show in Trace Matrix View*→*Show all Occurrences*.

This will display all instances of the selected metaclass from all current activations.

Pre-defined selections

There are several pre-defined menu entries in the *MetaModelAgent*→*Show in Trace Matrix View* submenu. Some of those will only appear if they are applicable according to the metamodel. These menu entries are only available when one single element is selected



 <i>All Invisible Relationships</i>	Shows all relationships within the selected scope that are not visible in any structural diagram within the current model.
 <i>All Unknown Relationships</i>	Shows all unidentified relationships within the select scope. An unidentified relationship is a relationship that does not match any metaclass in the metamodel. You may notice that nested relationships to unidentified elements are not reported.
 <i>All Incoming Relationships</i>	Shows all incoming relations to the selected element and its nested elements from elements outside the selected element.
 <i>All Outgoing Relationships</i>	Shows all outgoing relations from the selected element and its nested elements to elements outside the selected element.
 <i>All Relationships</i>	Shows all relationships owned by the selected element or any nested element.
 <i>All Navigable Associations</i>	Shows all navigable associations that are navigable from the selected element or any nested element. Associations that are not navigable in any direction will not be included.

 <i>All Composition Properties</i>	Shows all properties representing compositions owned by the selected element or any nested element. Matrix cells will display the property, the owner of the property will be the source and the type of the property will be displayed as target.
 <i>All Properties</i>	Shows all properties owned by the selected element or any nested element. Matrix cells will display the property, the owner of the property will be the source and the type of the property will be displayed as target.
 <i>All References</i>	Shows all relations or attributes owned by the selected element or a nested element.
 <i>Interface Connections</i>	Shows all interfaces within the selected element and nested elements. Matrix cells will display interfaces and each row will display a interface consumer (that has a usage relationship to the interface) and each column will display an interface provider (that has an realization relationship to the interface).

IMPORTANT: When using one of the predefined menu entries listed above, control flows, object flows and transitions will not be included in the result.

Visible and invisible connections in a diagram

By selecting a diagram in the explorer view or the background canvas of a diagram in the diagram editor. You will have two menu entries within the *MetaModelAgent*→*Show in Trace Matrix View* submenu:

 Visible Connections	Shows all connections (edges) that are visible in the diagram.
 Invisible Connections	Shows all connections between elements in the diagram that do not occur in the diagram.

5.1.2 From Property Table View

From the Property Table View it is possible to populate the Matrix View based on a more precise selection of scope and also to display implicit derived relations, based on underlying explicit relationships.

In this context, an *Implicit Derived Relation* between two elements occurs if the two elements either has a direct relationship two each other or consists of at least one pair of nested elements that have a relationship between each other.

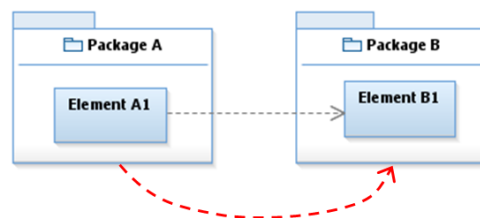







Figure 4: *There is an implicit derived relation between Package A and Package B, manifested by the explicit relationship between Element A1 and Element B1.*

Being able to display implicit derived relations is a very useful feature to be able to find out how larger parts of a model is related to other parts, or how a complete model is related to other models.

Show in Trace Matrix submenu in the *Property Table View* context menu:

 <i>Outgoing Relations →<Metaclass></i>	Shows all direct outgoing relationships of the selected kind of relationship from the selected set of elements.
 <i>All Outgoing Relations</i>	Shows all direct outgoing relationships of any kind from the selected set of elements.
 <i>Derived Outgoing Relations Towards→<Metaclass></i>	Shows all outgoing relationships from the selected elements or from any of their nested elements to elements of the selected metaclass.
 <i>Derived Incoming Relations From→<Metaclass></i>	Shows all incoming relationships to the selected metaclass to the selected elements or to any of their nested elements.
 <i>Derived Interrelationships</i>	Shows all relationships between the selected elements or their nested elements.

5.2 View Toolbar

The view's toolbar provides the following tools:



Shows the context of the source and target elements. This feature may have no effect in some kind of selections.



Refreshes the content of the Trace Matrix View. If the view has been resized, the number of targets displayed are adjusted to fit the new view width. You may observe that the content of the view is normally not updated if it has been changed elsewhere. To make sure the view is updated with changed data, make a new invocation from the context menu you last used.



Link the content of the Trace Matrix View to the element selected in the Explorer View or in the Diagram Editor. This means that the same selection criteria will be used but the scope of the selection will change.



Exports the content of the view to a CSV-file which can be post-processed in some other tool, for example MS Excel. The CSV-file will not contain any details of the references other than the number of references or the reference chain length between each source and targets.



A pull-down menu where you can switch matrix mode, apply matrix filters and initiate MMA Chart View based on the content of the matrix view, as described in chapter 5.2 below. The pull-down menu icon differs between different versions of the Eclipse environment.

The submenu contains the following menu entries

- Normal, Transitive and Transitive (Start-End) mode switches (see chapter 5.2 below)
- Circular filter that which will focusing on circular relationships and references by eliminating all other relationships and references from the matrix.
- Show in Chart View, which will initiate the Chart View with related elements based on the context in the matrix view in a bar chart or scatter chart style.

5.3 Context Menu

There are context menus available for all source and targets elements, as well as for the filled cells in a matrix:

- For a source element it is possible to show all directly and indirectly outgoing relations in the *Trace Tree View* or all related elements in the *Trace Tree View*.
- For a target element it is possible to show all directly and indirectly incoming references in the *Trace Tree View* or all elements that has references to the target element in the *Property Table view*.
- For a filled cell you will be able to either populate the *Property Table View* with the underlying relationships or populate the *Trace Tree View* with reference chains between source and target, depending on how the Matrix View was invoked.

5.4 Matrix Modes and filters



MetaModelAgent provides several matrix modes which can be altered using the entries in the view's pull-down menu. The modes are described in detail below.

Besides matrix modes switches, the view's menu also provides a circularity filter that can be applied to any mode and which will focusing on circular relationships and references by eliminating all other relationships and references from the view.

5.4.1 Normal Mode

When the matrix mode is initially populated, it is in the *Normal Mode* with contents based on how the view was invoked.

Thereafter it is possible to switch to *Transitive mode* and finally *Transitive (start-end) Mode* using the view's dropdown menu. To return to a previous mode, you will have to invoke the Trace Matrix View again.

Mode switching is available in the view's drop down menu, indicated by the  button in the view's button menu. From Eclipse release 2019-12, this icon is changed to .

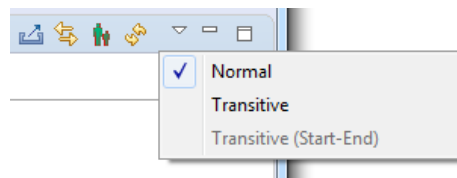


Figure 5: Trace Matrix View dropdown menu for mode switching.

5.4.2 Transitive Mode

Transitive mode shows if two elements are related to each either directly or indirectly by following one or several consecutive relationships. Each cell representing such an *implicit transitive relation*. The cell will have a color from blue to red that indicating the shortest distance between source and target.

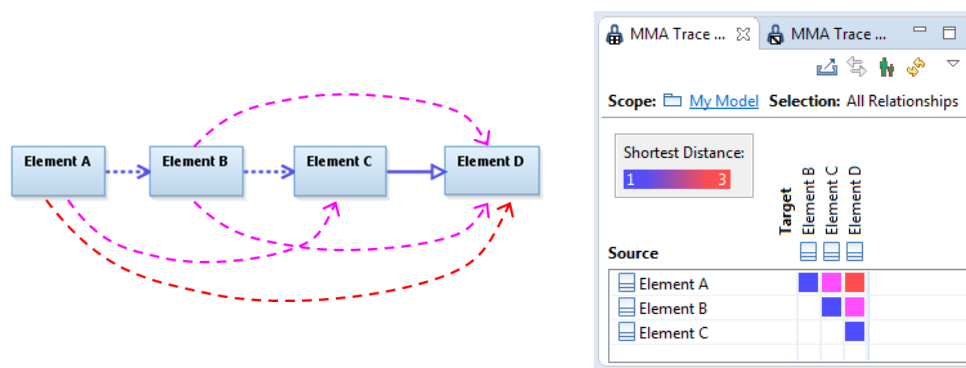


Figure 6: The blue cell indicates a direct relationship.
The magenta and red cells indicate implicit transitive relations between the element with the shortest distance of two and three.

Examine the path(s) of transitive relations

The relationship path(s) between source and target elements can be examined in details in the *Trace Tree View* by selecting “*Show all path(s) in Trace Tree View*” from the cell’s context menu.

IMPORTANT: If there are a lot of unique paths between source and target, it may take several seconds or even minutes to calculate all paths.

5.4.3 Transitive (Start-End) mode

Transitive (Start-End) Mode is a filtered variant of the *Transitive Mode* where only complete paths between elements are shown.

There will be a color indicating the shortest distance from a source element to a target element only if there are no incoming relationships to the source element and no outgoing relationship to the target element.

Paths that contains loops and therefore do not have final element will not be displayed.

In the example above, only the red arrow between Element A and Element B would be displayed in the *Transitive (Start-End) Mode*.

6 Trace Tree View

The *Trace Tree View* is used to display reference chains starting from, and/or ending at, a specific element. Within the view, several features support the analysis of complex reference chains.

All kind of UML relationships can be traced in this view including navigable associations, object flows, control flows and transitions, as well as attribute type references.

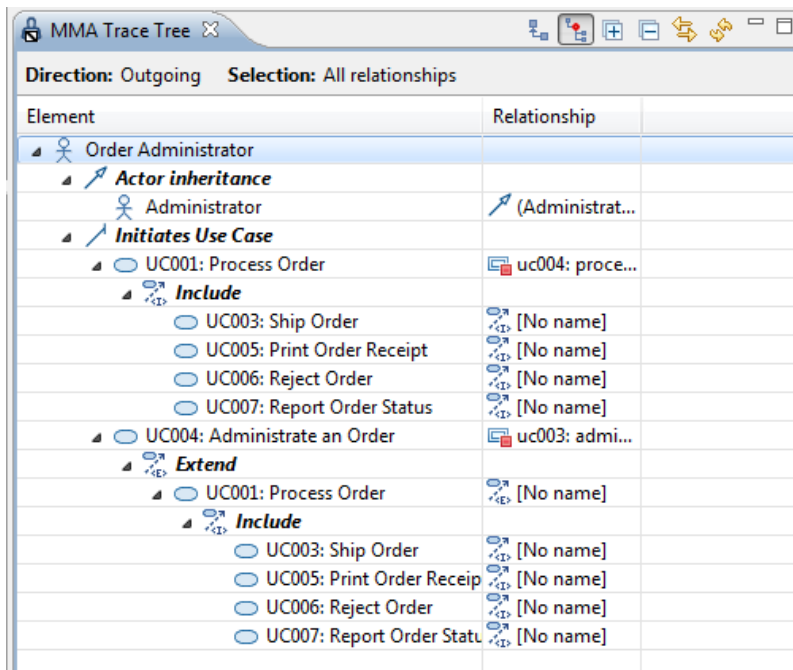


Figure 7: Trace Tree View displaying all relationships starting from the actor Order Administrator.

If an existing element already displayed occurs again then it will be highlighted in red, indicating a circular relationship.

By selecting an element in the tree, all other occurrences of the same element in the tree are highlighted in blue and a popup-dialog will display the complete relationship chain from the root element to the selected element. This make it easy to inspect the details of a specific relationship chain.

Double-clicking an element will highlight the element in the Explorer View.

6.1 Invoking the View






6.1.1 From Explorer View and Diagram Editor

To display the outgoing or incoming relationship chain from a specific element in the Explorer View or Diagram Editor.

1. Select one or several items in the Explorer View or in the Diagram Editor for which you are interested in their nested relationships or association ends.
2. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Matrix View*.
3. Continue by selecting *Outgoing Relationships* or *Incoming Relationships* in the nested submenu.

4. Select the metaclass representing the kind of relationship or reference to be traced.

You will also have a set of predefined selections available, depending on the selected element:

 <i>All Generalization</i>	Shows the path of generalizations starting from or ending at the selected element.
 <i>All Relations</i>	Shows the path of all directed relations starting from or ending at the selected element.
 <i>All Compositions</i>	Shows the path of all composition properties starting from the selected element (this alternative is not available in the incoming references submenu).
 <i>All References</i>	Shows the path of all directed relationships and properties starting from or ending at the selected element.
 <i>All Interface Connection</i>	Shows the path of usage of interfaces realized by other elements or the path or realized interfaces used by other elements.

IMPORTANT: It is possible to select metaclasses representing state machine transitions, control flows and object flows using the steps described above. But those selections will not respect UML semantics of state machines and activities. To analyze those kind of relationships with respect to semantics, start by populating the *Trace Matrix View* using *Static Behavior Analysis*, see chapter 8, from which the Trace Tree View then could be populated as described below.



6.1.2 From Trace Matrix View





Based on a populated Trace Matrix View, the Trace Tree View can be used to further elaborate the content starting from a source element, a target element or a filled matrix cell.

Selection	Context menu entry	Description
Source element	<i>Show outgoing(s) in Trace Tree View</i>	Displays a tree of all outgoing references represented in the Matrix View starting from the selected source element.
Target element	<i>Show incoming(s) in Trace Tree View</i>	Displays a tree of all incoming references represented in the Matrix View ending at the selected target element.
Filled matrix cell	<i>Show shortest path(s) in Trace Tree view</i>	Displays a tree of all minimal paths from source to target.
	<i>Show all path(s) in Trace Tree view</i>	Displays a tree of all potential paths from source to target.

6.2 View Toolbar










The view's toolbar provides the following tool:

-  Switches the tree to show incoming references instead of outgoing references.
-  Switches from grouping related elements, based on the relationship metaclass, to omit that node level, making the tree more compact.

-  Expand the tree to the bottom. E.g. all indirectly related elements will be exposed.
IMPORTANT: To avoid long execution time. The expansion will stop after a certain number of paths have been expanded. Very deep paths will also be truncated.
-  Compresses the tree, only the root will be visible.
-  Refreshes the content of the view.
IMPORTANT: As the content of the view in many cases are pre-calculated, refreshing the view does not always reflect changes in the underlying model made in other views and editors. To make sure that the view shows correct data after changes in the model, invoke the view once again.
-  Link the content of the view to the element selected in the Explorer View or in the Diagram Editor. This means that the same selection criteria will be used but the scope of the selection will change. This will only work if the Trace Tree view was invoked from the Explorer View.

6.3 View Context Menu

The context menu of the trace tree has the following menu entries:

- | | |
|---|---|
|  <i>Show Path Details</i> | Brings up a popup-dialog where the path from the root to the selected element is displayed in a table layout. |
|  <i>Expand All</i> | Expands the tree starting from the selected element.
IMPORTANT: To avoid long execution time. The expansion will stop after a certain number of paths have been expanded. Very deep paths will also be truncated. |
|  <i>Collapse All</i> | Collapses the tree until the selected element. |
|  <i>Navigate to relationship in Model Explorer</i> | Selects the relationship in the Model Explorer View. |
|  <i>Set as Root</i> | Makes the selected element the new root of the tree. |
|  <i>Filter tree on selected element</i> | Filters the tree so that only paths that leads from the root to the selected element will remain. The tree will be automatically expanded. |
|  <i>Show in Property Table View</i> | Shows all instances of a specific metaclass below the selected item in the relationship chain in the Property Table View. |
|  <i>Show in Trace Matrix View</i> | Shows all relationships beneath the selected element in the relationship chain in the Trace Matrix View. |
|  <i>Show in Trace Diagram</i> | Creates a new class diagram within the selected element that will visualize the complete relationship chain starting from the selected element.
IMPORTANT: If the selected element is not an element that can hold a class diagram, no diagram is created. If the elements and relationships to be displayed cannot exist in a class diagram, the resulting diagram will be incomplete. |

7 Chart View

The *Chart View* will provide you with different kind of bar charts and scatter charts that will help you analyze your models.



Figure 8: Chart View displaying an Enumerated Properties Bar Chart of Use Cases organized around status and priority.

All bar charts can be sorted in alphabetic order or by size and you can switch from vertical bars to horizontal bars. Select a bar and open the context menu to populate the Property Table View or the Problem View with the content of the bar.

In the upper right area of a bar chart you will see some statistics; Max and min height of the bars, the median bar height and the average bar height.

By hovering the dots in a scatter chart you can see the list of elements that each dot represents. In the context menu you will be able to populate the Property Table View with those elements.

7.1 Available Charts

The following kinds of charts are available:

<i>Properties Bar Chart</i>	Displays the distribution of the selected kind of items based on the value(s) of one or two selected properties. One of the properties values will be represented as bars and the other one as groups on the bars. Only properties with an upper multiplicity of 1 will be available for selection. If the number of property values to be displayed as bars exceeds 100 or the number property values to represent groups exceeds 30, the chart will not be displayed.
<i>Nested Elements Bar Chart</i>	Displays the total number of nested elements for the selected kind of elements, organized in groups based on their element kinds. The nesting depth to be included can be set in the settings area.

<i>Related Elements Bar Chart</i>	<p>Displays the number of related elements for each element of the selected metaclass, grouped based on their metaclass.</p> <p>You can set the depth of relations, as well as whether incoming or outgoing relations should be displayed in the settings area.</p> <p>IMPORTANT: Associations, control flows, object flows and transitions are not regarded as relations in this chart.</p>
<i>Problems Bar Chart</i>	<p>Displays the number of problems for each item of the selected metaclass and its nested items, grouped based on the severity of the problems.</p> <p>You can set the depth of nested items to be included in the settings area.</p>
<i>Problem Distribution Bar Chart</i>	<p>Displays the distribution of problems based on a selection of two of the following three criteria: <i>Severity</i>, <i>Problem Kind</i> and <i>Metaclass</i>. One of the selected criteria will make up the bars and the other one will be represented by groups on each bar.</p> <p>IMPORTANT: This chart is only available from the Problem View.</p>
<i>Related Elements Scatter Chart</i>	<p>Displays a scatter chart of all nested elements of the specific kind, where one axis represents number of incoming relationships and the other axis represents the number of outgoing relationships.</p> <p>The color (from blue to red) of the “dots” in chart indicates how many elements that the dot represent.</p> <p>You can set the depth of relationships to be included in the settings area.</p>

7.2 Invoking the View

7.2.1 From Explorer View, Diagram Editor and Activation View

The bar charts and scatter chart are available from the context menu in Explorer View, Diagram Editor and Activation View.

To invoke one of the charts:

1. Select one or several elements as the scope.
2. Bring up the context menu and open submenu *MetaModelAgent* → *Show in Chart View*.
3. Select the kind of chart among the submenu entries. For each kind you will finally select the metaclass of elements to be displayed.

For each kind of chart, the available settings are displayed in the upper left area. After the settings have been set, click the **Apply**-button to bring up the corresponding chart.

7.2.2 From Property Table View

The same context submenu with different charts as above is available when selecting one or several element rows in the Property Table View.

7.2.3 From Trace Matrix View

Related Elements Bar Chart and *Related Element Scatter Chart* are available from the Matric View's view menu. These menu entries makes it possible to get a graphical overview of the current content in the Trace Matrix View.

7.2.4 From Problem View

The *Problem Distribution Bar Chart* could be invoked from the Problem View.

1. Select a set of problems in the Problem View.
2. Bring up the context menu and select *Show in Chart View* → *Problem Distribution Bar Chart* to open the *MetaModelAgent Chart View*.
3. In the chart view header you can switch between letting the bars and groups represents metaclasses, problem kinds and severities.

7.3 View Toolbar

The view's main menu have three icon buttons:



Shows the context of the elements.



Refreshes the content of the view. However, you will need to click Apply once again to display the chart.



Link the content of the view to the element selected in the Explorer View or in the Diagram Editor. This means that the same selection criteria will be used but the scope of the selection will change.

7.4 View Context Menu

By bringing up the context menu a in a bar chart or scatter chart you will have the following context menu entries:

<i>Adjust Axis Range</i>	Submenu with entries for adjusting the range of the X and Y axis.
<i>Zoom In</i>	Submenu with entries for zooming in on X and/or Y-axis.
<i>Zoom Out</i>	Submenu with entries for zooming out on X and/or Y-axis.
<i>Save As...</i>	Opens a dialog where you can save the chart as a picture in png or jpeg format.
<i>Properties...</i>	Provides a dialog where you can customize the layout of the chart in a numerus ways.
<i>Show in Property Table View</i>	Only available when a dot in a scatter chart or a bar segment in a bar chart is selected. Populates the Property Table View with all elements that are represented by the selected dot or bar segment.

8 Static Behavior Analysis

Trace Matrix View in combination with Trace Tree View provides excellent capabilities for static analysis of activities, state machines and UML-RT capsules.

Those analysis are based on the default semantics in UML and UML-RT and are independent on which metamodel is applied.

This chapter describes how to perform static behavior analysis of State Machines (UML and UML-RT), Activity Flows (UML) and Capsule compositions (UML-RT) as well as analysis of protocol usage (UML-RT).

IMPORTANT: For correct behavior when using UML-RT (Capsule modeling) in RSARTE or HCL RTist, make sure that the MetaModelAgent preference setting *Enable real-time modeling support* is checked.

8.1 Potential State Machine Transitions

Static analysis of state machine transitions involves finding all potential transitions between states and their entry and exit points. As no simulation or evaluation is made, all transitions found are potential. E.g. they may occur depending on their guards.

Potential transitions between states that are not explicitly modeled are reported as implicit references in the analysis result. For example exiting from the CycleLight composite state in the example below, which is possible from all three nested states.

Implicit references are also used to denote how entry/exit points are connected to states.

The state machine is flattened in the analysis. That means that only the atomic states will be displayed in the analysis result.

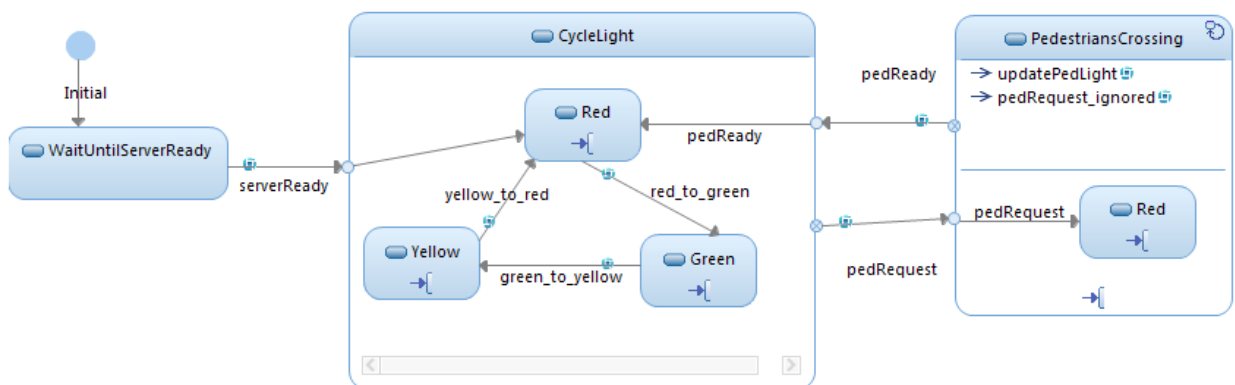


Figure 9: State machine of Traffic Light capsule used as an example in the state machine transition analysis below.

8.1.1 Transition Paths to/from a specific State, Entry or Exit point

To analyze all *potential forward transition paths* from a specific atomic state or entry/exit point in the Trace Tree view.

1. Select the atomic state or entry/exit point that should be the starting point in the Explorer View or in the Diagram Editor.
2. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Tree View*→*Outgoing References*→*Potential Transition Paths*.

To analyze all potential transition paths backwards from a specific atomic state or entry/exit point in the Trace Tree View.

1. Select the atomic state or entry/exit point that should be the end point in the Explorer View or in the Diagram Editor.
2. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Tree View*→*Incoming References*→*Potential Transition Paths*.

As the state machine is flattened, composite states containing nested states can't be the starting point for these analysis.

Element	Reference
▲ CycleLight::Red	
▲ CycleLight::<No name>	[Implicit reference]
▲ PedestriansCrossing::<No name>	pedRequest
▲ PedestriansCrossing::Red	pedRequest
▲ CycleLight::<No name>	[Implicit reference]
▲ CycleLight::Red (circular)	pedReady
CycleLight::Red (circular)	pedReady
▲ CycleLight::Green	red_to_green
▲ CycleLight::<No name>	[Implicit reference]
▲ PedestriansCrossing::<No name>	pedRequest
▲ PedestriansCrossing::Red	pedRequest
▲ PedestriansCrossing::<No name>	[Implicit reference]
▲ CycleLight::<No name>	pedReady
CycleLight::Red (circular)	pedReady
▲ CycleLight::Yellow	green_to_yellow
▲ CycleLight::<No name>	[Implicit reference]
▲ PedestriansCrossing::<No name>	pedRequest
▲ PedestriansCrossing::Red	pedRequest
▲ PedestriansCrossing::<No name>	[Implicit reference]
▲ CycleLight::<No name>	pedReady
CycleLight::Red (circular)	pedReady
CycleLight::Red (circular)	yellow_to_red

Figure 10: Potential transition paths from the *CycleLight::Red* atomic state in the state machine example above.

Only the state machine that owns the selected state, entry or exit point will form the scope for this analysis. Any state machine that inherits from the state machine will not be considered.

If you need to analyze the transition paths starting from or ending at an element that is reused in an inheriting state machine, please populate the Trace Matrix View with that inheriting state machine and select a state or entry/exit point from the matrix to populate the Trace Tree View. See 8.1.2 below.

8.1.2 All Transitions in a Hierarchical State Machine

All transitions in a state machine and its composite states can be displayed in the *Trace Matrix View*.

3. Select a state machine or a composite state in Explorer View or in the Diagram Editor.
(In an UML-RT model you could also select a Capsule, as a capsule is supposed to have a companion state machine).
4. Bring up the context menu and open submenu *MetaModelAgent*→*Show in Trace Matrix View*→*State Machine Transitions*.

Nested state machines will be flattened, e.g. only the most nested states will be displayed. If a nested state machine contains a history state. This history state will be substituted with all states that potentially could be the next state.

Pseudo states such as entry/exit points, fork, join and junctions are excluded in the resulting matrix. Initial states are always displayed as well as entry/exit states if the selected scope is a composite state.

Transition paths that passes excluded pseudo states are displayed as transitive relations in the matrix cells.

The analysis does not evaluate any guards or other expressions that controls the transitions being traversed. The resulting matrix will therefore show “potential” transitions.

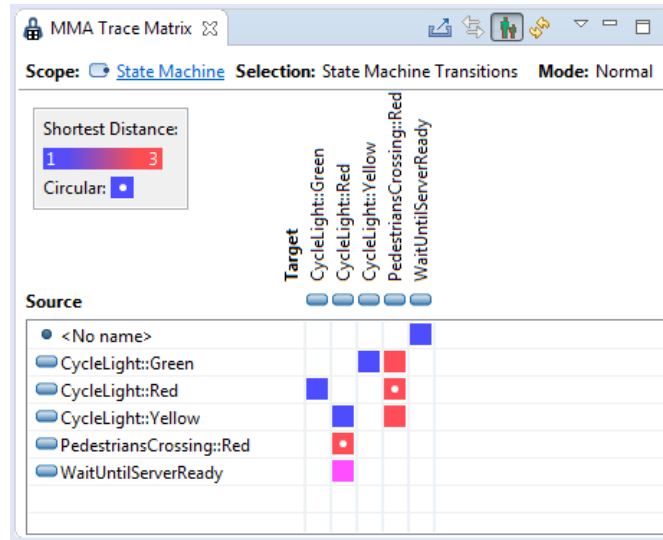


Figure 11: All potential transitions in a state machine excluding entry/exit points. Transitive relationships in red indicates transition chains where entry/exit points are traversed.

Switching to Transitive Mode in the view's drop down menu highlights which states that can be reached from other states, directly or indirectly. The color indicates the number of consecutive transitions (explicit and implicit) from source to target state.

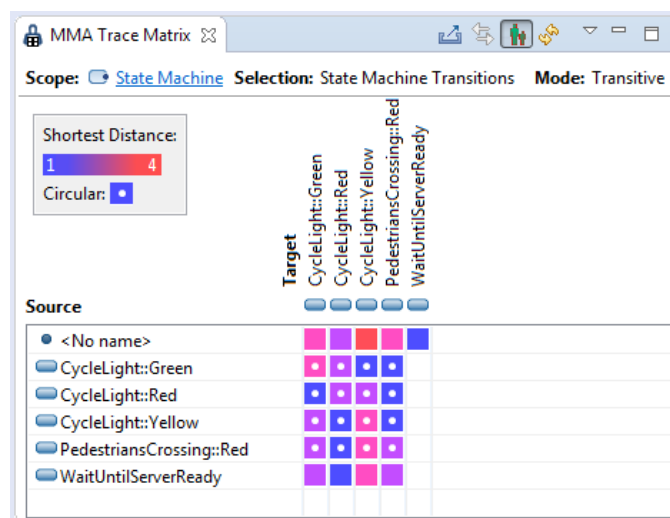


Figure 12: Turning the matrix view into Transitive Mode to show potential transition paths between all states.

The chain of transitions making up a transitive relation can be further examined in the Trace Tree View. Select a cell representing a transitive relation, bring up the context menu and select Show all path(s) in Trace Tree View, see next chapter for details about the Trace Tree View.

Element	Reference
<No name>	
WaitUntilServerReady	Initial
CycleLight::Red	Transitive Relation
CycleLight::Green	red_to_green
CycleLight::Yellow	green_to_yellow

Figure 13: The potential transition path from the initial state to the Yellow state by selecting the intersection cell (red) in the Transitive Model of the Trace Matrix View.

8.2 Potential Activity Flows

Static analysis of activities involves finding all potential control and object flows between actions and other activity nodes. As no simulation or evaluation is made, all flows found are potential. E.g. they may occur depending on their guards.

Implicit references are used to denote how pins are connected to actions.

Analysis of an activity will include analysis of any activity referred from call behavior actions.

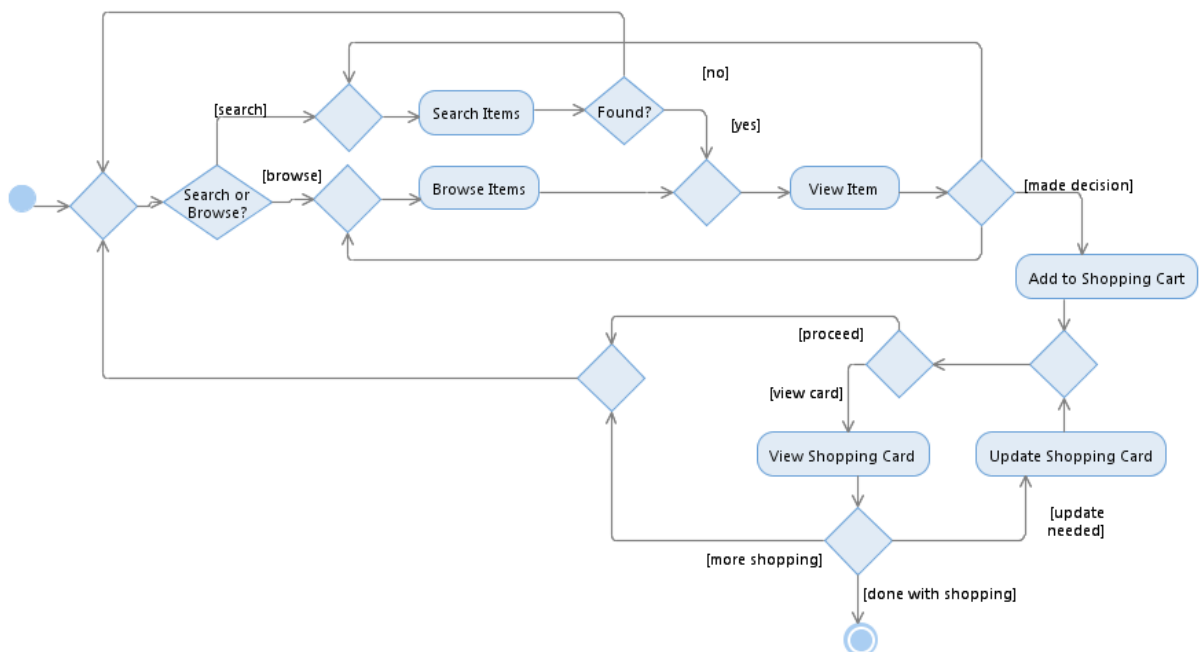


Figure 14: Activity Flow of an Online Shopping process used as an example in the analysis below.

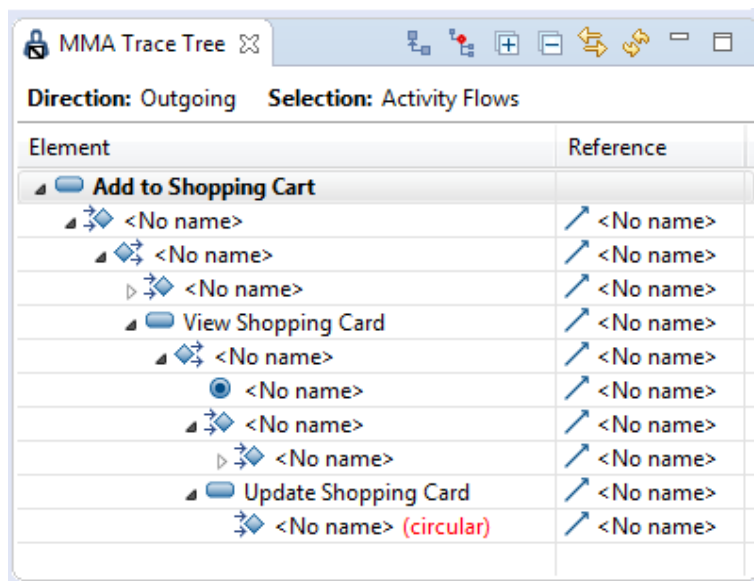
8.2.1 Activity Flows to/from a specific Action

To analyze all *potential flow paths* from a specific activity node (e.g. actions, control nodes, object nodes or pins) in the Trace Tree view:

1. Select the activity node that should be the starting point in the Explorer View or in the Diagram Editor.
2. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Tree View*→*Outgoing References*→*Potential Flow Paths*.

To analyze all *potential flow paths* towards a specific activity node in the Trace Tree view:

1. Select the activity node that should be the end point in the Explorer View or in the Diagram Editor.
2. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Tree View*→*Incoming References*→*Potential Flow Paths*.



Element	Reference
▲ Add to Shopping Cart	
▶ <No name>	▶ <No name>
▶ <No name>	▶ <No name>
▶ <No name>	▶ <No name>
▶ View Shopping Card	▶ <No name>
▶ <No name>	▶ <No name>
● <No name>	▶ <No name>
▶ <No name>	▶ <No name>
▶ <No name>	▶ <No name>
▶ Update Shopping Card	▶ <No name>
▶ <No name> (circular)	▶ <No name>

Figure 15: Potential flow paths from the action Add to Shopping Cart in the example above. The paths are not fully expanded.

8.2.2 All Activity flows in an Activity and referenced Activities

All Control and Object Flows within an Activity with nested structured activity nodes and referenced activities (through call behavior actions) can be analyzed at different levels of abstraction in Trace Matrix View.

1. Select an Activity in the Explorer View or in the Diagram Editor.
2. Bring up the context menu and open submenu *MetaModelAgent*→*Show in Trace Matrix View*→*Activity Flows*.
3. Select the level of details among the submenu items to bring up the *Trace Matrix View*.

The different submenu items represents different level of abstractions and details:

<i>All Nodes</i>	Shows all control and object flows between actions, control and object nodes including input/output pins. The “connection” between input/output pins and their actions are shown as implicit relations in the matrix.
------------------	--

<i>Excluding Control Nodes</i>	Same as above but control nodes, except initial and final node, are excluded.
<i>Excluding Pins</i>	Shows all control and object flows between actions, control and object nodes but not pins.
<i>Excluding Control Nodes and Pins</i>	Shows all control and object flows between actions, object nodes. Control Nodes, except initial and final node, and Pins are excluded.

For all about the “All nodes” selection, a filled cell may represent a flow path that passes one or several control nodes and/or pins that have been excluded. The color of the cell indicates the shortest path between the source and target action. The details of such a flow path can be examined in the Trace Tree View using the cell’s context menu.

The analysis does not evaluate any conditions or other expressions that controls the flows being traversed. The resulting matrix will therefore show *potential* flows.

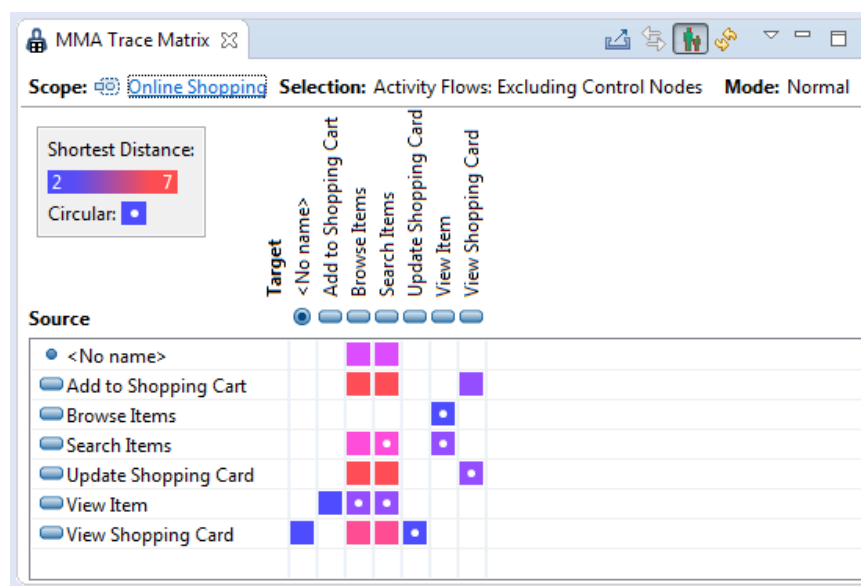


Figure 16: Potential flows between actions in the Online Shopping process above where control nodes have been excluded.

As with State Machine Transitions in Trace Matrix View, the Transitive mode can be used for Activity Flows to get an overview of which actions that can be reached from another action.

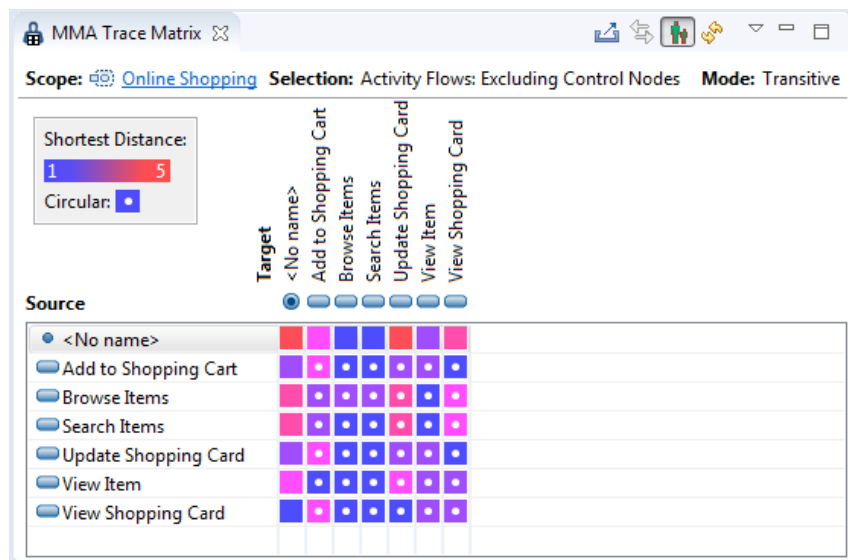


Figure 17: Distance between all pair of actions in terms of intermediate actions.

By bringing up a cell's context menu in Transitive Model, you are able to populate the Trace Tree View with all potential paths between the source and target node.

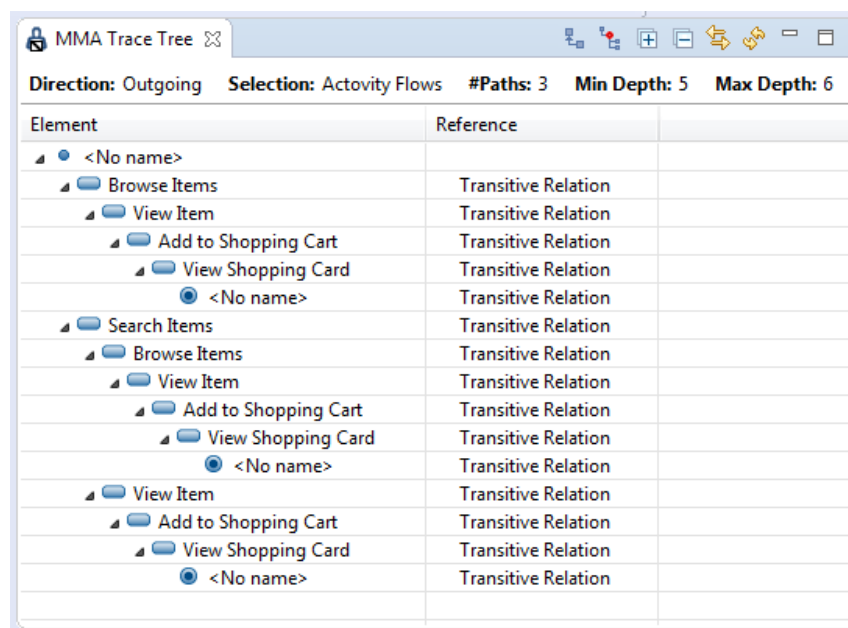


Figure 18: All potential paths from initial node to activity final node in the Activity Flow above.

IMPORTANT: Activity partitions will not be considered. All variants of Structure Activity Nodes (Expansion Region, Conditional Node, Loop Node and Sequence Node) will be reported as Structure Activity Nodes. The potential loop within a Loop Node will there for not be shown.

8.3 Capsule Connectors

Capsule is a concept within UML for Real-time (UML-RT) supported in RSARTE and HCL RTist.

Connectors within a Capsule and its composited Capsule Parts' capsules can be analyzed at different levels of abstraction in Trace Tree View and in Trace Matrix View.

Dynamic Connections for non-wired ports will however not be included as they are impossible to analyze without interpretation/simulation.

As no simulation or evaluation is made the following constraints holds:

- All connections are said to be potential. E.g. they may occur depending on the state machine behavior in the involved capsules.
- Dynamic Connections for non-wired ports will not be included.

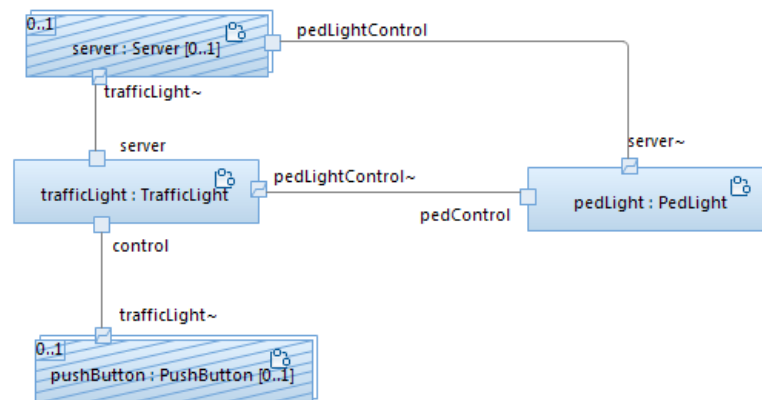


Figure 19: Composite Structure of Capsule TLSystem capsule used as the example in the analysis below.

8.3.1 Potential Connector paths to/from a Port in a Capsule Part

To analyze all potential connection paths starting from a specific port:

1. Select the starting port in a composite structure diagram
2. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Tree View*→*Outgoing References*→*Potential Port Connection Paths*.

To analyze all potential connection paths towards a specific port:

3. Select the final port in a composite structure diagram
4. Bring up the context menu and select *MetaModelAgent*→*Show in Trace Tree View*→*Incoming References*→*Potential Port Connection Paths*

Both this operations will display a path of potential connected ports in the Trace Tree View.

Alternatively, you can select a starting or end capsule part and display the connection path on a capsule part level instead, leaving out the involved ports.

The context menu does also provides option to only display explicit connection paths.

Element	Reference
TLSystem.pushButton.trafficLight~	
└─ [RT Connector]	
└─ TLSystem.trafficLight.control	LightControl, LightControl~
└─ [Implicit Reference]	
└─ TLSystem.trafficLight.pedLightControl~	[Implicit reference]
└─ [RT Connector]	
└─ TLSystem.pedLight.pedControl	PedLightControl, PedLightControl~
└─ [Implicit Reference]	
└─ TLSystem.pedLight.server~	[Implicit reference]
└─ [RT Connector]	
└─ TLSystem.server.pedLight	PedLightControl~, PedLightControl
└─ [Implicit Reference]	
└─ TLSystem.trafficLight.server	[Implicit reference]
└─ [RT Connector]	
└─ TLSystem.server.trafficLight~	TrafficLightSignals~, TrafficLightSignals

Figure 20: Trace Tree View of potential communication paths between ports starting from the traffic light port in the capsule part pushButton.

8.3.2 All Connectors in a Capsule Composition Structure

Invocation

1. Select a Capsule in the Explorer View or in the Diagram Editor.
2. Bring up the context menu and open the submenu *MetaModelAgent*→*Show in Trace Matrix View*→*Capsule Connectors*.
3. Select the level of abstraction among the submenu items to bring up the *Trace Matrix View*.

The different levels of abstraction available in the context submenu are:

<i>All Explicitly Connected Ports</i>	Shows all explicit connectors between ports within the selected Capsule and all its composited capsules.
<i>All Potential Connected Ports</i>	Same as above but also including potential implicit connections between behavior ports handled by a capsule's state machine.
<i>All Potential Connected Parts</i>	Shows all connectors between capsule parts, within the selected Capsule and all its composited capsules, including potential implicit connections handled by a capsule's state machine.

IMPORTANT: For correct behavior, make sure that the MetaModelAgent preference setting *Enable real-time modeling support* is checked.

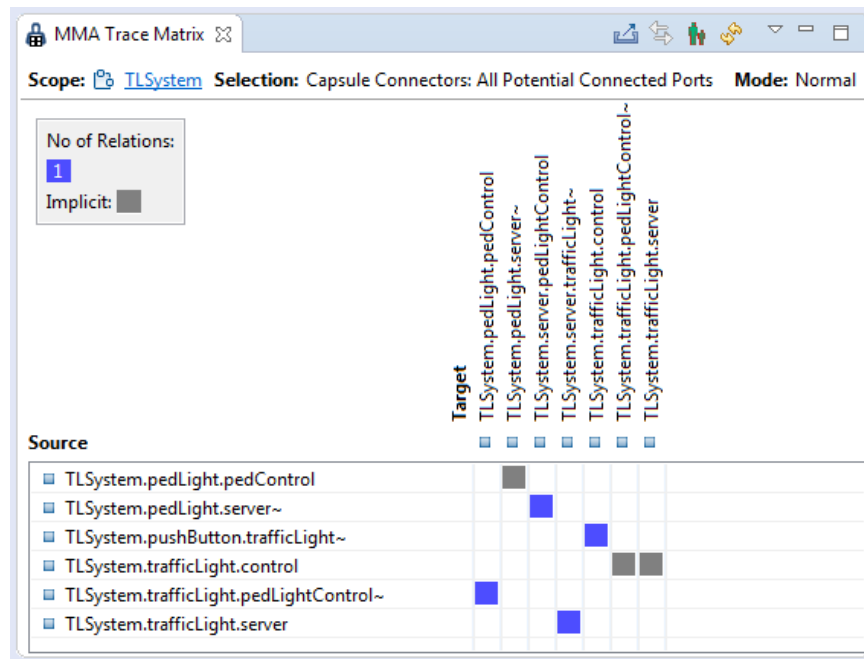


Figure 21: Trace Matrix View of all potential connected ports. The grey cells indicate potential non-explicit connectors between behavior ports handled by the capsule's state machines.

To further analyze the potential communication ways between ports, switch to Transitive Mode using the view's menu and to show all ports that potentially can communicate with each other.

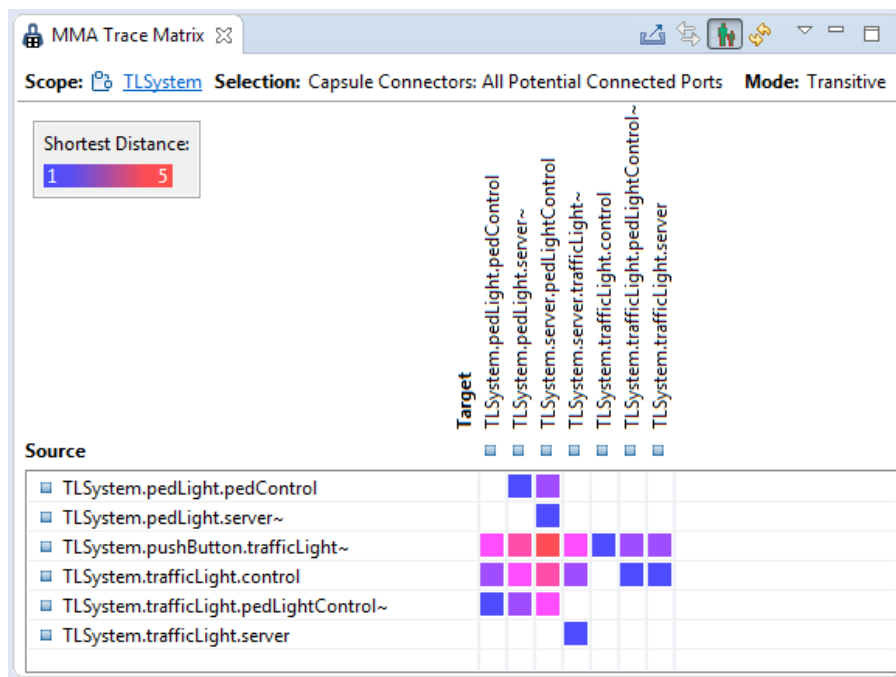
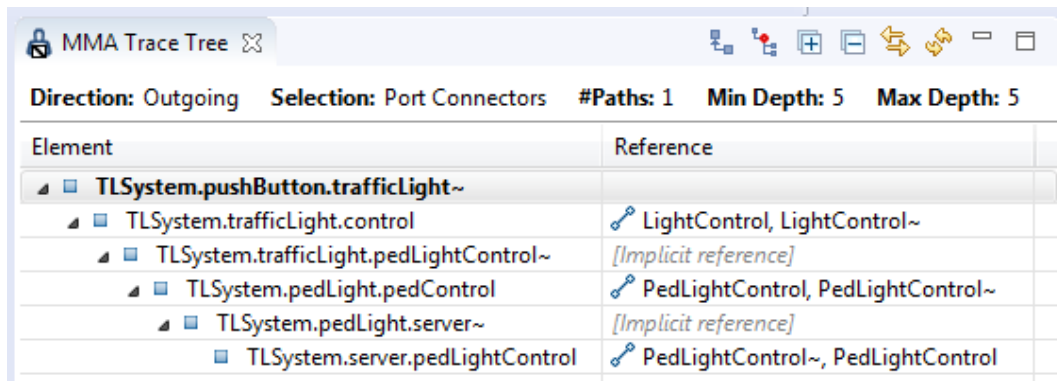


Figure 22: The Trace Matrix View in transitive mode showing which ports could be connected directly or indirectly using explicit connectors and potentially through behavior ports according to the capsule's state machines behavior.

Select a filled cell representing potential communication path(s) and explore the path(s) further by invoking the Trace Tree View using the cell's context menu.



The screenshot shows the MMA Trace Tree window with the following settings: Direction: Outgoing, Selection: Port Connectors, #Paths: 1, Min Depth: 5, Max Depth: 5. The table below represents the data shown in the tree.

Element	Reference
TLSystem.pushButton.trafficLight~	
TLSystem.trafficLight.control	LightControl, LightControl~
TLSystem.trafficLight.pedLightControl~	[Implicit reference]
TLSystem.pedLight.pedControl	PedLightControl, PedLightControl~
TLSystem.pedLight.server~	[Implicit reference]
TLSystem.server.pedLightControl	PedLightControl~, PedLightControl

Figure 23: The potential communication paths between two of the ports. Implicit reference represent potential communication dependent on state machine behavior.

IMPORTANT: When analyzing capsule structures, the label of a port will consist of the path of parts starting from the initial capsule. If the composite structure of capsules are very deep, the corresponding path to the port will also be deep. In the Trace Matrix View, this can lead to the labels being truncated. Enlarge the view to minimize the problem with truncated labels.

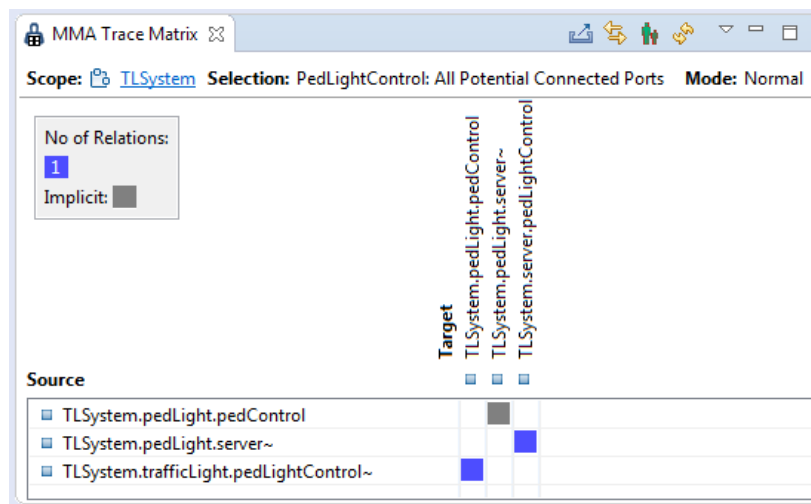
8.4 Capsule Connectors for a specific Protocol

The Matrix view can also be filtered to only display connectors that connects ports typed by a specific protocol. This is useful for analyzing usage of a specific protocol within the composite structure of a capsule.

Invocation

1. Select a connector within a Capsule in the Explorer View or in the Diagram Editor.
2. Bring up the context menu and open the submenu *MetaModelAgent* → *Show in Trace Matrix View* → *Protocol <Protocol name> Connectors*.

The protocol usage analyzed will be the protocol used by the ports connected by the connector. If the two ports of the connector are typed by different protocols, the most specific of those protocols are used.



The screenshot shows the MMA Trace Matrix window with the following settings: Scope: TLSystem, Selection: PedLightControl: All Potential Connected Ports, Mode: Normal. The matrix shows the following data:

Source	TLSystem.pedLight.pedControl	TLSystem.pedLight.server~	TLSystem.server.pedLightControl
TLSystem.pedLight.pedControl			
TLSystem.pedLight.server~			
TLSystem.trafficLight.pedLightControl~			

Figure 24: Trace Matrix View of all potential connected ports using the protocol *PedLightControl* within the capsule *TLSystems* composite structure.

Switch to Transitive (Start-End) Mode to see all complete paths of the specific protocol in the capsule structure.

Select a filled cell and select “Show all path(s) in Trace Tree View to display a specific complete connector path using that protocol.

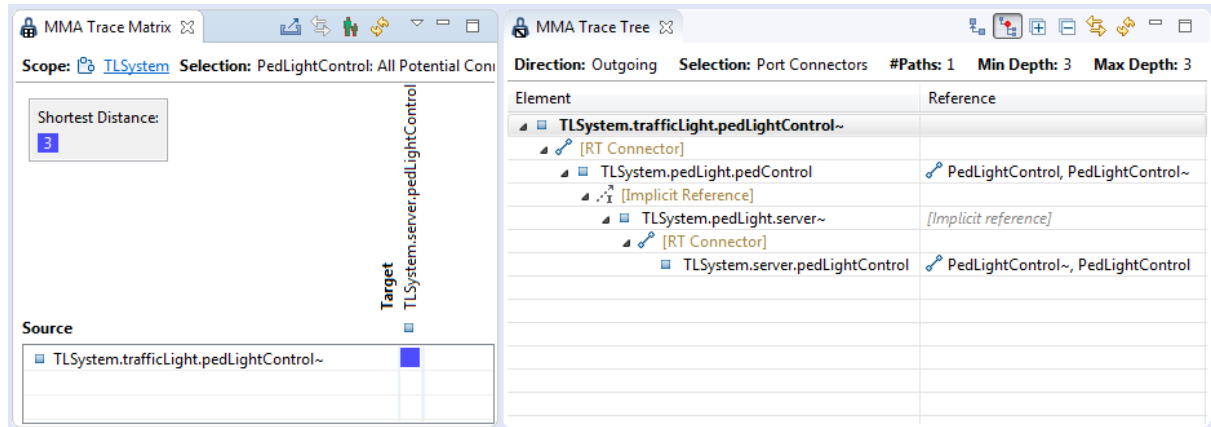


Figure 25: Trace Matrix View of the only complete path of the protocol *PedLightControl* and the details of the same path expanded in the Trace Tree View.

8.5 Overall Protocol Usage

All usage of a specific protocol can also be displayed in the Trace Matrix View

1. Select the protocol in Project Explorer or in the Diagram Editor.
2. Select *MetaModelAgent* → *Show in Trace Matrix View* → *All Connectors using Protocol* to bring up the *Trace Matrix View*.

The Matrix View will display all connectors in all activated models that are using the selected protocol or inherited protocols.

IMPORTANT: This gives an overview of the protocol usage, but there is no idea to turn to Transitive Mode for analyzing the potential communication paths. Instead you should use analysis described in chapter 0 if you would like to analyze the potential communication path for a specific protocol.